CS281A/Stat241A: Statistical Learning Theory

The Junction Tree Algorithm (11/16/04)

Lecturer: Michael I. Jordan *Scribes: Wei Xu and Qi Zhu*

1 Overall Procedure

The *junction tree algorithm* is a general algorithmic framework, which provides an understanding of the general concepts that underly inference. The general problem here is to calculate the conditional probability of a node or a set of nodes, given the observed values of another set of nodes. We have treated a number of inferential calculations in graphical models, but all of them are special cases. The idea of junction tree algorithm is to find ways to decompose a global calculation on a joint probability into a linked set of local computations. The key point of this approach is the concept of locality. A particular data structure – the junction tree – is introduced to make explicit the important relationship between graph-theoretic locality and efficient probabilistic inference.

The overall procedure of the junction tree algorithm is as follows:

1. Moralization

   In the moralization step, a directed graph is converted into an undirected graph, so a uniform treatment of directed and undirected graphs is possible. Recall that the moral graph $\mathcal{G}^m$ is obtained by linking the parents of each node and dropping the directionality of the edges in directed graph $\mathcal{G}$.

2. Triangulation

   The triangulation step determines the "elimination" order of the graph. The elimination algorithm we introduced before is a special case of the junction tree algorithm, and *UndirectedGraphEliminate* can be viewed as a procedure for creating a triangulated graph, by selecting the order of elimination of the graph's nodes. The triangulation step will be discussed further later.

3. Construct the junction tree

   Given a triangulated graph, a junction tree is constructed by forming a maximal spanning tree from the cliques in that graph. A clique tree will be constructed with separators. Note that not every clique tree is a junction tree. A clique tree is a junction tree if and only if it has the *running intersection property* or the *junction tree property*. This will be discussed further.

4. Transfer the potentials

   Associated with each clique we define a potential $\psi_C(x_C)$, a nonnegative function on the realizations of clique X_C. For this algorithm, the potentials on the hypergraph are initialized from those of an underlying graph. The potentials of separators are initialized to unity.

5. Propagate

   Computation proceeds in the junction tree as follows. Suppose we have two cliques V and W and suppose that V and W have a non-empty intersection S, as shown in Fig 1.

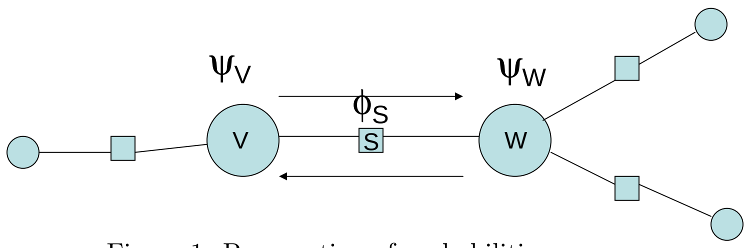

Figure 1: Propagation of probabilities

The cliques V and W have potentials ψ_V and ψ_W, and we also endow S with a potential ϕ_S that we initialize to unity. The basic operation of the junction tree algorithm is a propagation of information between V and W, with S serving as a conduit for the flow of information. The update equations (one direction) are as follows:

$$\phi_S^* = \sum_{V \setminus S} \psi_V$$

$$\psi_W^* = \frac{\phi_S^*}{\phi_S} \psi_W$$

The updates must respect the Message-Passing Protocol. Once the algorithm terminates, the clique potentials and separator potentials are proportional to marginal probabilities. Further marginalization can be performed to obtain the probabilities of singleton nodes or other subsets.

We proved in class that after two iterations of the update equations, following the Message-Passing protocol, the potentials are *consistent*:

$$\sum_{V \setminus S} \phi_V = \sum_{W \setminus S} \phi_W$$

2 Junction Tree Property

The idea of the junction tree algorithm is to utilize graph-theoretic locality. However, for general clique trees, local consistency does not imply global consistency. Fig 2 is an example, where the clique tree for that graphical model has a problematic feature. Node D appears in two cliques in the tree and these two cliques are not neighbors. Given that our algorithm only enforces local consistency, there is no guarantee that the two cliques containing D will be consistent (i.e., there is no requirement that the marginal probability of node D is the same for both sets of cliques).

In Fig 3, the clique tree corresponding to the graphic model does not have this problematic feature.

We articulate a property that rules out the problematic configurations of the kind that we saw in Fig 2.

The junction tree property: A clique tree possesses the junction tree property if for every pair of cliques V and W, all cliques on the (unique) path between V and W contain $V \cap W$.

For any junction tree, we have following theorem:

Theorem 1 In a junction tree, local consistency implies global consistency.

Theorem 2 Global consistency can be achieved if and only if, when the junction tree algorithm terminates, the clique potentials and separator potentials are proportional to local marginal probabilities. In particular, $\psi_C(x_C) \propto p(x_C)$.

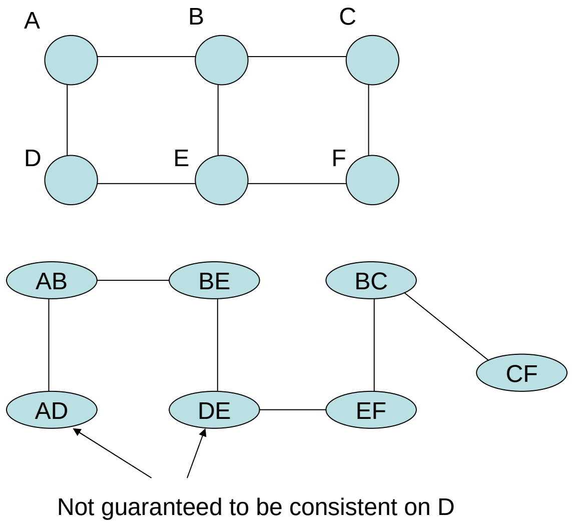

Figure 2: An undirected graphical model and a corresponding clique tree

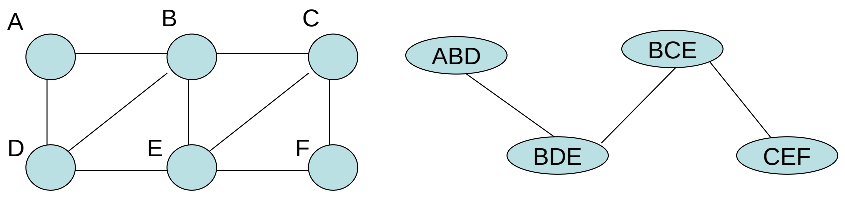

Figure 3: Another undirected graphical model and its corresponding clique tree. This clique tree is a junction tree.

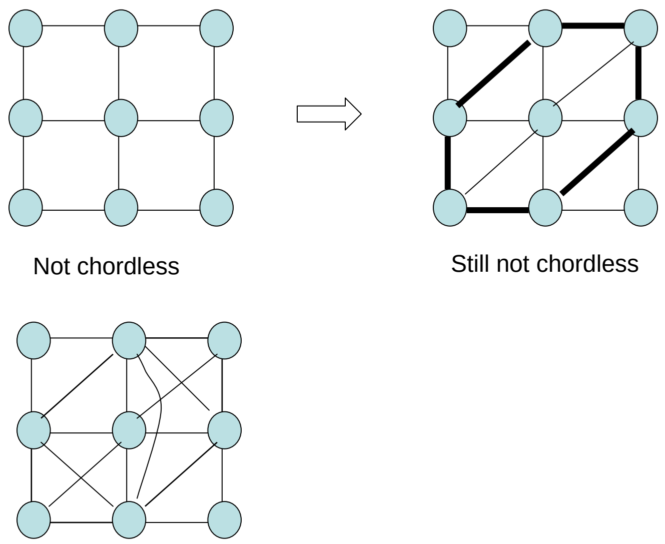

Figure 4: Non-triangulated graph and one possible triangulation

From above theorems, we know that if the clique tree is a junction tree, and if we run the message-passing procedure as described before, we achieve not only local consistency but also global consistency. Which graphs have a junction tree? That is the topic of next section.

3 Triangulation Graph and Junction Tree

We present a sufficient condition for a graph to have a junction tree – the graph must be *triangulated*. It turns out that triangulation is also a necessary condition for a graph to have a junction tree.

The definition of triangulated graph is as follows. First, we define a cycle in a graph as *chordless* if there are no edges between nodes that are not successors in the cycle (for cycles of four or more). Then we have following definition of triangulated graph:

Triangulated graph A graph is triangulated if it has no chordless cycles.

Fig 4 is an example of non-triangulated graph, which does have chordless cycle. And by adding more edges, it can be triangulated.

Triangulation implies the existence of a clique tree with the junction tree property, although we do not prove this theorem here.

Theorem 3 All triangulated graphs have a junction tree.

How do we find the junction tree with a small maximal clique?

It turns out that the elimination algorithm always produces a triangulated graph (proof by induction traced). But there are $n!$ orderings of the elimination algorithm, so one can prove the finding an optimal elimination ordering is NP-hard.

Given an elimination ordering (hence a triangulation), how do you find a junction tree? Fig 5 shows that not all the clique trees obtained from a triangulated graph are junction trees.

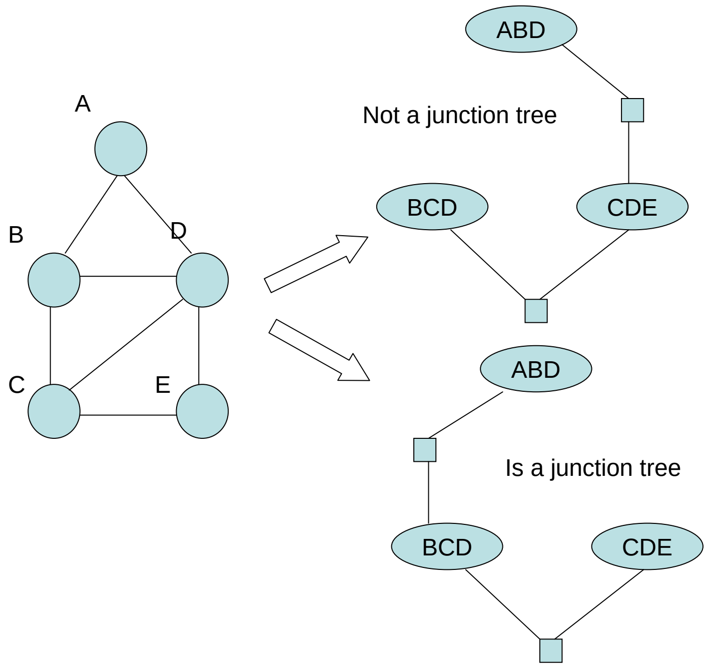

Figure 5: A graphic model and two corresponding clique trees

We define a weight $w(T)$ as the sum of the cardinalities of the separator sets in clique tree T , then we have following theorem.

Theorem 4 A clique tree T is a junction tree if and only if it is a maximal spanning tree.

Proof The total weight of a clique tree is equal to the sum of the cardinalities of its separators. Therefore,

$$\begin{aligned} w(T) &= \sum_{j=1}^{M-1} |S_j| \\ &= \sum_{j=1}^{M-1} \sum_{k=1}^{N} 1(X_k \in S_j) \\ &= \sum_{k=1}^{N} \sum_{j=1}^{M-1} 1(X_k \in S_j) \\ &\leq \sum_{k=1}^{N} \left[\sum_{i=1}^{M} 1(X_k \in C_i) - 1 \right] \\ &= \sum_{i=1}^{M} \sum_{k=1}^{N} 1(X_k \in C_i) - M \\ &= \sum_{i=1}^{M} |C_i| - M \end{aligned}$$

Note that the right-hand side is independent of T. The inequality is an equality if and only if T is a junction tree. So we obtain an upper bound on the weight of a junction tree.