STAT 241: HOMEWORK #2

MIKE SCHACHTER

Problem 1. Consider an undirected cycle, where each node can take on K potential states.

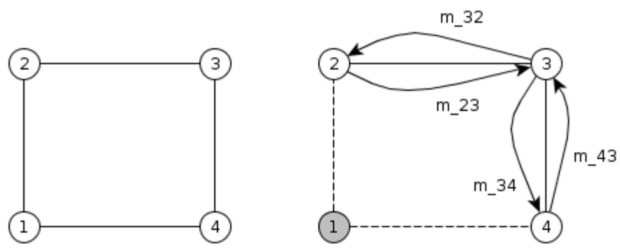

Figure 1: An undirected 4-cycle on the left. When node 1 is conditioned on, we get the graph on the right, the dotted lines indicate the cycle being disconnected and turned into a chain. The arrowed edges represent messages sent between nodes in the SUM-PRODUCT algorithm.

a) Devise an algorithm for computing all single node marginals using SUM-PRODUCT and conditioning.

Consider a 4-cycle G with cliques $\mathcal{C} = \{(x_1, x_2), (x_2, x_3), (x_3, x_4), (x_4, x_1)\}$, and clique potentials ψ_{12},ψ_{23},ψ_{34},ψ_{41}, where each potential is parameterized by it's subscripts. The graph G has the joint distribution $p(x_1 x_2 x_3 x_4) = \frac{1}{Z}\psi_{12}\psi_{23}\psi_{34}\psi_{41}$. Suppose we condition on $x_1 = \bar{x}_1$, this breaks the 4-cycle into a chain by making clique potentials ψ_{12} and ψ_{41} only dependent on x_2 and x_4, respectively.

The evidence probability $p(x_1 = \bar{x}_1)$ can be computed as:

$$p(\bar{x}_1) = \frac{1}{Z} \sum_{x_2,x_3,x_4} \psi_{12}\psi_{23}\psi_{34}\psi_{41}$$

$$= \frac{1}{Z} \sum_{x_3,x_4} \psi_{34}\psi_{41} \sum_{x_2} \psi_{12}\psi_{23}$$

$$= \frac{1}{Z} \sum_{x_3,x_4} \psi_{34}\psi_{41} m_{23}(x_3, \bar{x}_1)$$

$$= \frac{1}{Z} \sum_{x_4} \psi_{41} m_{34}(x_2, \bar{x}_1)$$

$$= \frac{1}{Z} m_{41}(\bar{x}_1)$$

where message values are given as:

$$m_{23}(x_3, \bar{x}_1) = \sum_{x_2} \psi_{12}\psi_{23}$$

$$m_{34}(x_4, \bar{x}_1) = \sum_{x_3} \psi_{34} m_{23}(x_3, \bar{x}_1)$$

$$m_{41}(\bar{x}_1) = \sum_{x_4} \psi_{41} m_{32}(x_2, \bar{x}_1)$$

where m_{41} is a message "sent" to conditioned node $\bar{x}_1$. We can compute this quantity for each of K values of x_1 to get the marginal probability $p(x_1)$. Given a specific value $x_1 = \bar{x}_1$, we can compute conditional marginal probabilities for all other nodes in the cycle:

$$p(x_2|\bar{x}_1) = \frac{1}{Z} \sum_{x_3, x_4} \psi_{12}\psi_{23}\psi_{34}\psi_{41}$$

$$= \frac{1}{Z} \sum_{x_3} \psi_{12}\psi_{23} \sum_{x_4} \psi_{34}\psi_{41}$$

$$= \frac{1}{Z} \sum_{x_3} \psi_{12}\psi_{23} m_{43}(x_3, \bar{x}_1)$$

$$= \frac{1}{Z} \psi_{12} m_{32}(x_2, \bar{x}_1)$$

$$p(x_3|\bar{x}_1) = \frac{1}{Z} \sum_{x_2, x_4} \psi_{12}\psi_{23}\psi_{34}\psi_{41}$$

$$= \frac{1}{Z} \sum_{x_2} \psi_{12}\psi_{23} m_{43}(x_3, \bar{x}_1)$$

$$= \frac{1}{Z} m_{23}(x_3, \bar{x}_1) m_{43}(x_3, \bar{x}_1)$$

$$p(x_4|\bar{x}_1) = \frac{1}{Z} \sum_{x_2, x_3} \psi_{12}\psi_{23}\psi_{34}\psi_{41}$$

$$= \frac{1}{Z} \sum_{x_3} \psi_{34}\psi_{41} \sum_{x_2} \psi_{12}\psi_{23}$$

$$= \frac{1}{Z} \sum_{x_3} \psi_{34} \psi_{41} m_{23}(x_3, \bar{x}_1)$$

$$= \frac{1}{Z} \psi_{41} \sum_{x_3} \psi_{34} m_{23}(x_3, \bar{x}_1)$$

$$= \frac{1}{Z} \psi_{41} m_{34}(x_3, \bar{x}_1)$$

By writing these out we can see that messages can be re-used across conditioned-marginal computations. Since we can compute the marginal $p(x_1)$ from repeated application of the first equation for all x_1, the law of total probability can be used to compute the unconditioned marginals for x_2, x_3, x_4:

$$p(x_{i \neq 1}) = \sum_{j=1}^{K} p(x_i | x_1 = j) p(x_1 = j)$$

b) What is the computational complexity of using the conditioned-cutset approach vs. using the junction tree algorithm?

Let's start by computing the complexity for (a). Computing the conditioned-marginal $p(x_1 = \bar{x}_1)$ requires 3 nested summations of K terms, with a complexity of $O(K^3)$. To compute $p(x_1)$, we must compute K conditioned-marginals, increasing complexity to $O(K^4)$. Computing $p(x_2|\bar{x}_1)$ requires $O(K^2)$ operations, and these are nested within the summation that uses total probability to compute $p(x_2)$, bringing the total order of complexity of computing $p(x_2)$ to K^3. The same holds for $p(x_3)$ and $p(x_4)$. So the total number of additions performed for all 4 marginals is something like $K^4 + 3K^3$. In the general case of an N-cycle, the marginal for the node conditioned on is $O(K^N)$, but every other marginal will still be $O(K^3)$, giving an overall complexity in the general case of $O(K^N + NK^3)$, exponential in N.

For the junction tree algorithm: intuitively from looking at small 4, and 5 cycles, the biggest clique that arises during "sensible" triangulations is of size 3. From wild speculation and extrapolation on these small cycles, the number of maximal cliques is $N - 2$ for an N-cycle. During the running of the junction tree algorithm, each clique node contains 3 or less elements, each separator contains 2 or less elements. Each separator is updated at most twice, so each separator update is less than or equal to $O(K^2)$. Each clique node update is neglible in the context of separator updates, something like $O(2(N-2))$. So, excluding the triangulation step, the total complexity of the junction tree algorithm for an N-cycle could be at most something like $O(2(K^2 - N - 2))$, based on aforementioned speculation.

Problem 2. Outline the junction tree construction for an undirected tree $G = (V, E)$ parameterized with pairwise potentials $\psi_{st}(x_s, x_t)$ for $(s,t) \in E$. Derive the SUM-PRODUCT algorithm from the junction tree propagation rules.

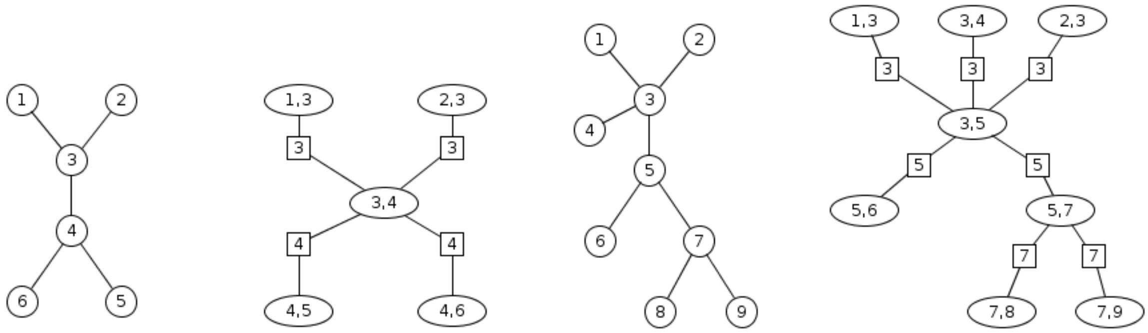

Figure 2: Two trees and their corresponding junction trees.

Construction of Junction Tree:

(1) Create unconnected clique nodes from all pairwise cliques.
(2) Connect clique nodes containing leaves to the clique nodes that contain their parent, adding a separator set on the edge that contains the parent.
(3) Connect all clique nodes that contain only non-leaf nodes to eachother, so as long as they have a non-empty intersection. Create separators between them that contain the intersection.

Note that all clique nodes contain two nodes of the original graph, and at least one of them is a non-leaf node.

Propagation Rules:

(1) Initialize all separators to 1, i.e. set $\phi_j = 1$
(2) Have each leaf node send a message to it's parent. Let $C = (a, b)$ be the leaf node and $P = (a, c)$ the parent node. The separator contains $S = C \cap P = (a)$. The first separator update is accomplished as follows:

$$\phi_S^{(1)} = \sum_b \psi_{ab}$$

$$\psi_{ac}^{(1)} = \phi_S^{(1)} \psi_{ac} = (\sum_b \psi_{ab})\psi_{ac}$$

After this, all non-leaf clique nodes will be marginalized with respect to their leaf-node-containing children. To be more formal, for a given clique non-leaf clique node K, let $\mathcal{C}_K$ be the set of K's neighbors that contain leaf nodes of the original tree. After this step, the potential function of a non-clique node will be given as:

$$\psi_K^{(1)} = (\prod_{J \in \mathcal{C}_I} \sum_{J \setminus K} \psi_J)\psi_K$$

(3) Have each non-leaf node in the junction tree send a message to another non-leaf node once it's received messages from all it's neighbors, using the same propagation rules as described above. If K and L are two non-leaf nodes sharing a separator $M = K \cap L$, K sends a message to L with the following update:

$$\phi_M^{(2)} = \sum_{J \setminus M} \psi_K^{(1)}$$

$$\psi_J^{(2)} = \phi_M^{(2)} \psi_L^{(1)}$$

Make all non-leaf clique nodes in the junction tree exchange messages with eachother in this manner. Because each non-leaf clique node acts as a cutset between subtrees in the junction tree, this process will exchange marginals across subtrees, and these marginals are now contained in the non-leaf nodes. The non-leaf clique nodes are now completely marginalized.

(4) Propagate messages from all non-leaf clique nodes to their children that contain leaf nodes in the original graph. This process will propagate all marginals from subtrees separated by the non-leaf clique nodes, and thus marginalize all the leaf clique nodes. Each separator has been updated twice.

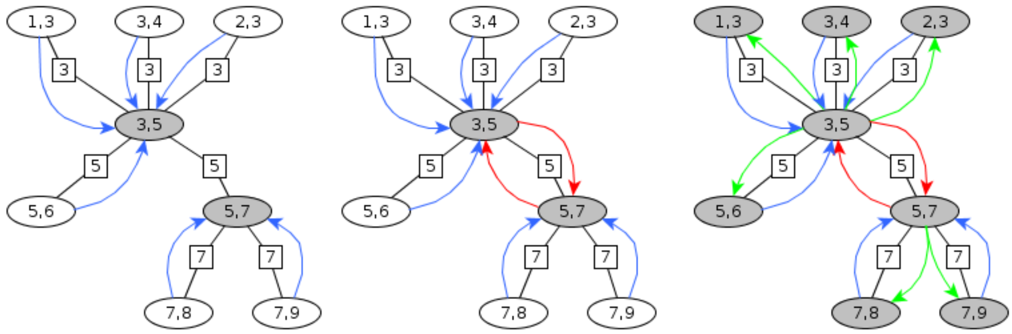

Figure 3: Propagation steps for a tree's junction tree. Blue corresponds to step (2), red to step (3), green to step (4). Filled clique nodes correspond to when the node's contents have changed.

Derivation of SUM-PRODUCT Algorithm:

By the end of step (4), each clique node contains the joint marginal potential for clique (s,t),such that

$$p(s,t) = \frac{1}{Z}\psi_{st}^{(final)}$$

To get single node marginals we marginalize the clique potentials:

$$p(s) = \frac{1}{Z}\sum_{t}\psi_{st}^{(final)}$$

If we expand out the ψ for a given clique node in Figure (3), such as $(7,8)$:

$$p(x_8) = \frac{1}{Z}\sum_{x_7}\psi_{78}^{(final)} = \frac{1}{Z}\sum_{x_7}\psi_{78}\sum_{x_5}\psi_{57}\sum_{x_3}\psi_{35}\sum_{x_6}\psi_{56}\sum_{x_1}\psi_{13}\sum_{x_4}\psi_{34}\sum_{x_2}\psi_{23}$$

Each summation term corresponds to a message in the SUM-PRODUCT algorithm. So the SUM-PRODUCT algorithm is basically the junction tree algorithm, with one extra marginalization step.

Problem 3. Consider the SUM-PRODUCT algorithm on an undirected tree with potential functions ψ_s and ψ_{st}. Consider any initialization of the messages such that $M_{ts}(x_s) > 0$ for all edges (s,t).

a) Prove by induction that the flooding schedule converges in at most diameter of graph iterations and that the message fixed point M^* can be used to compute marginals for every node of the tree:

$$p(x_s) \propto \psi_s(x_s) \prod_{t \in N(s)} M^*_{ts}(x_s)$$

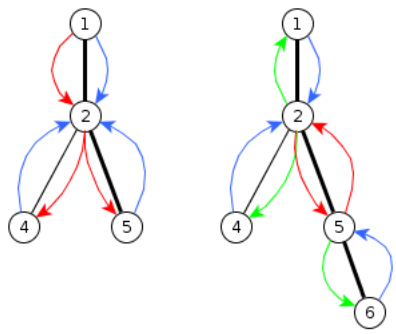

Figure 4: Trees with diameter 2 (left) and 3 (right). Messages sent using the flooding schedule are shown in color, iteration 1 (blue), 2 (red), and 3 (green). The longest shortest path is illustrated by the heavy edges.

Figure 4 shows an example of two trees that converge (send all their messages) in a number of iterations equal to their diameters. Assume a tree G_d with $dia(G_d) = d$ converges in d iterations. Does tree G_{d+1} converge in $d+1$ iterations? Let $\{a_i b_i\}$ be the set of longest shortest paths (diameters) of length $d+1$, where by construction any node b_i is a leaf node. If we prune all nodes $\{b_i\}$, we are left with a tree of diameter d, which by inductive assumption converges in d iterations. The re-addition of nodes $\{b_i\}$ adds one more set of messages which need to be propagated, and thus one more iteration (see figure 4), implying that a tree of diameter G_{d+1} converges in $d+1$ iterations.

At this point, each node has received all the messages it needs to compute it's marginal, so the relation $p(x_s) \propto \psi_s(x_s) \prod_{t \in N(s)} M^*_{ts}(x_s)$ holds for every node. This statement is true because of the message passing protocol and flooding schedule. The SUM-PRODUCT algorithm converges when all possible messages at a node x_s are sent and received from all neighbors. Each message contains a set of nested marginalization terms from sub-trees separated by the neighbors, and the marginalizations of the neighbors themselves. The product of these messages produces a marginalization over all other nodes in the tree besides x_s. All that's left to do is normalization by Z.

b) See the README, examples.py, and sum_product.py files attached to the email containing this homework.

Problem 4. Consider an undirected tree $T = (V, E)$.

a) Provide a modification to the SUM-PRODUCT algorithm that will yield edge marginals $p(x_i, x_j)$ for $(i, j) \in E$.

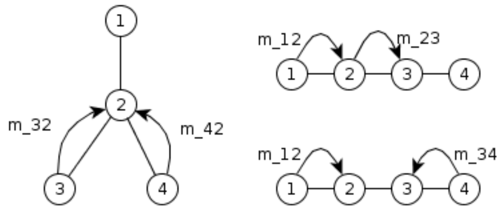

Figure 5: Messages required for computing edge marginals: $p(x_1, x_2)$ on left, $p(x_3, x_4)$ and $p(x_2, x_3)$on right.

Say we want to compute $p(x_1, x_2)$ for the tree on the left in figure 5. Marginalizing the joint probability gives:

$$p(x_1, x_2) = \frac{1}{Z} \sum_{x_3, x_4} \psi_1\psi_2\psi_3\psi_4\psi_{12}\psi_{23}\psi_{24}$$

$$= \frac{1}{Z}\psi_1\psi_2\psi_{12} \sum_{x_3} \psi_3\psi_{23} \sum_{x_4} \psi_4\psi_{24}$$

$$= \frac{1}{Z}\psi_1\psi_2\psi_{12} m_{32}(x_2) m_{42}(x_2)$$

So in this example, the edge marginals still require all messages going into node 2. Now examine the chain on the right of figure 5. Computing $p(x_3, x_4)$ gives:

$$p(x_3, x_4) = \frac{1}{Z} \sum_{x_1, x_2} \psi_1\psi_2\psi_3\psi_4\psi_{12}\psi_{23}\psi_{34}$$

$$= \frac{1}{Z}\psi_3\psi_4\psi_{34} \sum_{x_1, x_2} \psi_1\psi_2\psi_{12}\psi_{23}$$

$$= \frac{1}{Z}\psi_3\psi_4\psi_{34} \sum_{x_2} \psi_2\psi_{23} \sum_{x_1} \psi_1\psi_{12}$$

$$= \frac{1}{Z}\psi_3\psi_4\psi_{34} m_{23}(x_3)$$

And computing $p(x_2, x_3)$ gives:

$$p(x_2, x_3) = \frac{1}{Z} \sum_{x_1, x_4} \psi_1\psi_2\psi_3\psi_4\psi_{12}\psi_{23}\psi_{34}$$

$$= \frac{1}{Z}\psi_2\psi_3\psi_{23}\sum_{x_1}\psi_1\psi_{12}\sum_{x_4}\psi_4\psi_{34}$$

$$= \frac{1}{Z}\psi_2\psi_3\psi_{23}m_{12}(x_2)m_{43}(x_3)$$

This gives some intuition as to what's going on. In order to compute edge marginals, we need all messages that go into those edges. A modification to the SUM-PRODUCT algorithm that computes edge marginals would involve running the usual algorithm, and compute marginals for edge $(i, j) \in E$ as:

$$p(x_i, x_j) = \frac{1}{Z}\psi_i\psi_j\psi_{ij}\prod_{a\in N(i)\setminus j} m_{ai}(x_i)\prod_{b\in N(j)\setminus i} m_{bj}(x_j)$$

b) Consider computing arbitrary pairwise marginals in a tree. How can such a marginal be computed for a single pair? What can be said about the running time for this algorithm?

Given the chain on the right hand side of figure 5, say we want to compute $p(x_1, x_4)$:

$$p(x_1, x_4) = \frac{1}{Z}\sum_{x_2,x_3}\psi_1\psi_2\psi_3\psi_4\psi_{12}\psi_{23}\psi_{34}$$

$$= \frac{1}{Z}\psi_1\psi_4\sum_{x_2,x_3}\psi_2\psi_{12}\psi_{23}\psi_3\psi_{34}$$

$$= \frac{1}{Z}\psi_1\psi_4\sum_{x_2}\psi_2\psi_{12}\sum_{x_3}\psi_{23}\psi_3\psi_{34}$$

We get non-message terms such as $\sum_{x_3}\psi_{23}\psi_3\psi_{34}$. Without having much evidence, I'll make a claim that very little cost savings in terms of running time can be made when computing non-edge marginals, because of these non-message terms that are not local with respect to edges.

Problem 5. Consider a zero-mean Gaussian random vector $(x_1, ..., x_N)$ with a strictly positive NxN covariance matrix Σ. For a given undirected graph $G = (V, E)$ with N verticies, suppose that $(x_1, ..., x_N)$ obeys all the basic conditional independence properties of G, i.e. one for each vertex cut set.

a) Given the inverse covariance matrix $\Theta = \Sigma^{-1}$, show that $\Theta_{ij} = 0$ for all $(i, j) \notin E$.

Let $x = (x_1, ..., x_N)$. The pdf for the multi-variate Gaussian is given as:

$$p(x|\mu, \Sigma) = \frac{1}{Z} exp\{(x - \mu)\Theta(x - \mu)\}$$

$$= \frac{1}{Z} \prod_{(i,j)} exp\{(x_i - \mu_i)(x_j - \mu_j)\Theta_{ij}\}$$

where the product is taken over all (i, j). Let A and B be two index sets that respect a conditional independence relation given an index set C:

$$A \perp B \,|\, C$$

In order for the distribution to respect this relation, this density function has to factorize such that:

$$p(x_A, x_B|x_C, \mu, \Sigma) \propto f(x_A, x_C)g(x_B, x_C)$$

We can rewrite the density function like this:

$$\begin{aligned} p(x|\mu, \Sigma) \quad &= \quad \frac{1}{Z} \prod_{(i \in A, j \in C)} exp\{(x_i - \mu_i)(x_j - \mu_j)\Theta_{ij}\} \prod_{(i \in B, j \in C)} exp\{(x_i - \mu_i)(x_j - \mu_j)\Theta_{ij}\} \\ &\prod_{(i \in A, j \in B)} exp\{(x_i - \mu_i)(x_j - \mu_j)\Theta_{ij}\} \end{aligned}$$

The only way for this to factorize in the way we want is if $\Theta_{ij} = 0$ for $i \in A$, $j \in B$, which gives the following form:

$$p(x|\mu, \Sigma) = \frac{1}{Z} \prod_{(i \in A, j \in C)} exp\{(x_i - \mu_i)(x_j - \mu_j)\Theta_{ij}\} \prod_{(i \in B, j \in C)} exp\{(x_i - \mu_i)(x_j - \mu_j)\Theta_{ij}\}$$

For an undirected graph to have a joint distribution like this, it can't have edges between elements in A and B. By the Hammersely-Clifford theorem, if we have a joint distribution that factorizes as above, it satisfies the same conditional independence relations as the graph G, the family of distributions they characterize are the same.

b) Interpret this sparsity relation in terms of cut sets and conditional independence.

The nodes x_C are a cut set that separates x_A and x_B and provide the conditional independence between the two sets. The covariance between nodes in x_A and x_B can be nonzero when x_C is unknown, but the covariance between elements of the two sets, when conditioned on x_C, should be zero.