CS 228, Winter 2006
Final
Handout #16

1. **[8 points] Influence Diagrams** Consider a Influence Diagram containing a single utility factor in which all variables other than D(a decision node) and its parents have been eliminated using standard VE.

   Show that the decision rule δ_D that maximizes:

   $$\sum_{D,\mathbf{Pa}_D} \delta_D \cdot \mu_{-D}(D, \mathbf{Pa}_D).$$

   is defined as:

   $$\delta_D(\boldsymbol{w}) = \begin{cases} 1, & d = \text{argmax}_{d \in Val(D)} \mu_{-D}(d, \boldsymbol{w}) \\ 0, & otherwise \end{cases}$$

   for $\boldsymbol{w} \in Val(\mathbf{Pa}_D)$.

2. **[18 points] Causality**

   (a) **[5 points]** For probabilistic queries, we have that

   $$\min_x P(y \mid x) \leq P(y) \leq \max_x P(y \mid x).$$

   Show that the same property does not hold for intervention queries. Specifically, provide an example where it is not the case that:

   $$\min_x P(y \mid do(x)) \leq P(y) \leq \max_x P(y \mid do(x)).$$

   (b) **[6 points]** As for probabilistic independence, we can define a notion of causal independence: $(\boldsymbol{X} \perp_C \boldsymbol{Y} \mid \boldsymbol{Z})$ if, for any values $\boldsymbol{x}, \boldsymbol{x'} \in Val(\boldsymbol{X})$, we have that $P(\boldsymbol{Y} \mid do(\boldsymbol{Z}), do(\boldsymbol{x})) = P(\boldsymbol{Y} \mid do(\boldsymbol{Z}), do(\boldsymbol{x'}))$. (Note that, unlike probabilistic independence — $(\boldsymbol{X} \perp \boldsymbol{Y} \mid \boldsymbol{Z})$ — causal independence is not symmetric over $\boldsymbol{X}, \boldsymbol{Y}$.) Is causal independence equivalent to the statement: "For any value $\boldsymbol{x} \in Val(\boldsymbol{X})$, we have that $P(\boldsymbol{Y} \mid do(\boldsymbol{Z}), do(\boldsymbol{x})) = P(\boldsymbol{Y} \mid do(\boldsymbol{Z}))$. (Hint: Use your result from (a).)

   (c) **[7 points]** Prove that $(\boldsymbol{X} \perp_C \boldsymbol{Y} \mid \boldsymbol{Z}, \boldsymbol{W})$ and $(\boldsymbol{W} \perp_C \boldsymbol{Y} \mid \boldsymbol{X}, \boldsymbol{Z})$ implies that $(\boldsymbol{X}, \boldsymbol{W} \perp_C \boldsymbol{Y} \mid \boldsymbol{Z})$. Intuitively, this property states that, if changing $\boldsymbol{X}$ cannot affect $P(\boldsymbol{Y})$ when $\boldsymbol{W}$ is fixed, and changing $\boldsymbol{W}$ cannot affect $P(\boldsymbol{Y})$ when $\boldsymbol{X}$ is fixed, then changing $\boldsymbol{X}$ and $\boldsymbol{W}$ together cannot affect $P(\boldsymbol{Y})$.

3. **[10 points] Learning in DBNs**

   (a) **[5 points]** Suppose that we have fully-observed sequences (from time 0 to T) of the variables in a DBN. What is the marginal likelihood $P(D \mid \mathcal{G})$ of the DBN using $Dirichlet(1, \ldots, 1)$ priors on the parameters?

   (b) **[5 points]**
   Now suppose that we want to learn the optimal DBN structure (a $\langle \mathcal{B}_0, \mathcal{B}_{\rightarrow} \rangle$ pair) for the data from part(a) given some initial structure. Can we do this by running the usual greedy structure search algorithm (p. 399) on the BN produced by unrolling our DBN T time steps? If so, why? If not, what changes must be made?

4. **[21 points] Inference**

   Consider a chain probabilistic network $X_1 - X_2 - \cdots - X_n$ and a corresponding clique tree of the form $\boldsymbol{C}_1 - \cdots - \boldsymbol{C}_{n-1}$ where $Scope[\boldsymbol{C}_i] = \{X_i, X_{i+1}\}$. Each variable X_i has $|Val(X_i)| = d$ possible values. In this question we will consider finding the top most probable assignments, or explanations (MPEs). For simplicity, we will consider only the top three most probable assignments, and make the so-called *strict order assumption*, that no two assignments have exactly the same probability.

   Based on Section 8.4 and Exercise 8.14, we first run *max-product* messages passing to calibrate the clique tree and thus get the max-marginals:

$$M(X_i = a, X_{i+1} = b) = \max_{\{\boldsymbol{X} | X_i = a, X_{i+1} = b\}} P(\boldsymbol{X}) \qquad \forall i, a, b$$

   We derive the max-marginal for a single variable by:

$$M(X_i = a) = \max_{\{\boldsymbol{X} | X_i = a\}} P(\boldsymbol{X}) = \max_b M(X_i = a, X_{i+1} = b) \qquad \forall i, a$$

   (a) **[2 points]**

   Assume the above max-marginals are precomputed, find the most probable assignment (MPE): $\xi = \text{argmax}_{\boldsymbol{X}} P(\boldsymbol{X})$. Justisfy your answer.

   You don't need to give the probability of the MPE because the values in the max-marginals are unnormalized.

   (b) **[5 points]**

   Now find the second most probable assignment: $\text{argmax}_{\{\boldsymbol{X} | \boldsymbol{X} \neq \xi\}} P(\boldsymbol{X})$. Explain how to do that by just using the pre-computed max-marginals.

   (c) **[7 points]**

   Finding the third most probable assignment is more complicated as it cannot be computed from the pre-computed max-marginals alone. We define a new set of constrained max-marginals that has some variable constrained to certain values. Let $D_k \subset Val(X_k)$ be a subset of values that we want to contrain X_k to. The constrained max-marginal of X_i where X_k is constrained is then

$$M_{X_k \in D_k}(X_i = a) = \max_{\{\boldsymbol{X} | X_i = a, X_k \in D_k\}} P(\boldsymbol{X}) \qquad \forall i, a$$

   Explain how to compute the above constrained max-marginals for all i and a using max-product message passing.

   (d) **[7 points]**

   Find the third most probable assignment by using two sets of constrained max-marginals. Justify your answer.

   Note that you construct a set of constrained max-marginals by picking a variable and the range of values it is constrained to, and then using part (c) to compute the max-marginals.

5. **[14 points] Structure Learning**

   It is often useful when learning the structure of a Bayesian network to consider more global search operations. In this problem we will consider an operator called *Reinsertion* operator which works of follows: for the current structure $\mathcal{G}_t$, we choose a variable X to be

our *target variable*. The first step is to remove the variable from the network, by severing all connections to its children and parents. We then select the optimal set of parents and children for X, and reinsert it into the network with edges from the selected parents and to the selected children. In addition, we may want to limit the number of parents to a variable to K_p and the number of children to K_c. This restriction helps us to avoid the need to consider overly complex networks.

Throughout this problem, assume the use of the BIC score for structure evaluation.

(a) **[3 points]** Assume we've selected variable X_i as our target variable for this round. For now, let us assume that we have somehow chosen $\boldsymbol{U}_i$ to be optimal parents of X_i. Consider the case of $K_c = 1$ where we want to choose the *single* optimal child for X_i. Our possible children (nodes that won't introduce a cycle) are $Y_1....Y_N$. Write an argmax expression for finding the optimal child C. Explain your answer.

(b) **[7 points]** Now we want to consider the case of $K_c = 2$, where want to choose the optimal children pair for X_i. How do we find the optimal pair of children C_1 and C_2? Assuming our family score for any $\{X_k, \boldsymbol{U}_k\}$ can be computed in a constant time f. What is the best asymptotic computational complexity of finding this pair of children? Explain. Now consider the case where we want to choose the optimal set of children for arbitrary $K_c < N$. Using the same ideas, what is the computational complexity of this task?

(c) **[4 points]** We now consider the choice of parents for X_i. For this, we now assume that we have already somehow chosen the optimal set of children, and will hold them fixed. Can we do the same trick as above when choosing the parents? If so, show how. If not, argue why not.

6. **[13 points] Parameter Learning, Markov Networks**

In class, we discussed the problem of estimating the parameters of Bayesian networks using Maximum Likelihood Estimation (MLE). In this problem, we consider MLE for the parameters of Markov networks.

Assume that we are given a pairwise Markov network whose joint distribution is defined as

$$P(\boldsymbol{X}) = \frac{1}{Z} \prod_{\phi_i \in \mathcal{F}} \phi_i(\boldsymbol{C}_i)$$

where $\boldsymbol{C}_i$ is the subset of variables in $\boldsymbol{X}$ in the scope of factor ϕ_i, and Z is the partition function defined as

$$Z = \sum_{\mathbf{x}'} \prod_{\phi_i \in \mathcal{F}} \phi_i(\boldsymbol{c}_i')$$

where $\mathbf{c}_i'$ is a specific assignment to variables in $\boldsymbol{C}_i$ given by the assignment $\mathbf{x}'$.

We now wish to learn the potentials of the Markov network. In contrast to the case of Bayesian networks, there is no closed form solution for MLE estimate. Instead, we will use gradient ascent.

To do so, we first parameterize our model. Let $Val(C_i)$ be the set of joint values that the variables in C_i can take, and denote $\boldsymbol{c}_{ij}$ the j'th such value. Now define the parameters of the model as $\theta_{ij} = \log \phi_i(c_{ij})$.

(a) **[1 points]** Write the log-likelihood $l(\Theta : \mathcal{D})$ using this parameterization

(b) **[11 points]** Prove that the gradient of the log-likelihood function with respect to one of these parameters is given by:

$$\frac{\partial l(\Theta : \mathcal{D})}{\partial \theta_{ij}} = M\left[\boldsymbol{E}_{\hat{P}}[\mathbf{1}\{\boldsymbol{C}_i = \boldsymbol{c}_{ij}\}] - \boldsymbol{E}_{P}[\mathbf{1}\{\boldsymbol{C}_i = \boldsymbol{c}_{ij}\}]\right].$$

where $\hat{P}$ represents the empirical distibution, P represents the distribution given by the network with the current set of parameters, and M represents the number of data instances.

(c) **[1 points]** Give an interpretation for the direction of the gradient.

7. **[16 points] Approximate Inference** This problem explores one heuristic approach for deterministic search in a Bayesian network. It is an intermediate method between full-particle search and distributional-particle search: It uses partial instantiations as particles, but does not perform inference on the resulting conditional distribution.

   Assume that our goal is to provide upper and lower bounds on the probability of some event $\boldsymbol{y}$ in a Bayesian network $\mathcal{B}$ over $\mathcal{X}$. Let $X_1, \ldots, X_n$ be some topological ordering of $\mathcal{X}$. We enumerate particles that are partial assignments to $\mathcal{X}$, where each partial assignment instantiates some subset $X_1, \ldots, X_k$; note that the set $X_1, \ldots, X_k$ is not an arbitrary subset of $X_1, \ldots, X_n$, but rather the first k variables in the ordering. Different partial assignments may instantiate different prefixes of the variables. We organize these partial assignments in a tree, where each node is labeled with some partial assignment $(x_1, \ldots, x_k)$. The children of a node labelled $(x_1, \ldots, x_k)$ are $(x_1, \ldots, x_k, x_{k+1})$, for each $x_{k+1} \in \mathit{Val}(X_{k+1})$. We can iteratively grow the tree by choosing some leaf in the tree, corresponding to an assignment $(x_1, \ldots, x_k)$, and expanding the tree to include its children $(x_1, \ldots, x_j, x_{k+1})$ for all possible values x_{k+1}.

   Consider a particular tree, with a set of leaves $L = \{\ell[1], \ldots, \ell[M]\}$, where each leaf $\ell[m] \in L$ is associated with the assignment $\boldsymbol{x}[m]$ to some subset of variables $\boldsymbol{X}[m]$.

   (a) **[3 points]** Each leaf $\ell[m]$ in the tree defines a particle. Specify the assignment and probability associated with this particle, and describe how we would compute its probability efficiently.

   (b) **[8 points]** Show how to use your probability estimates from part 7a (a) to provide both a lower and an upper bound for $P(\boldsymbol{y})$.

   (c) **[5 points]** Based on your answer from part (b) provide a simple heuristic for choosing the next leaf to expand in the partial search tree.