Probabilistic Foundations of Artificial Intelligence
Solutions to Problem Set 2

Oct 21, 2016

1. Bayesian Networks: d-separation

As discussed in class, conditional independence properties entailed by a Bayesian Network can be read directly from the graph, using the notion of *d-separation*. Given the Bayesian network in Figure 1, which of the following conditional independence statements hold:

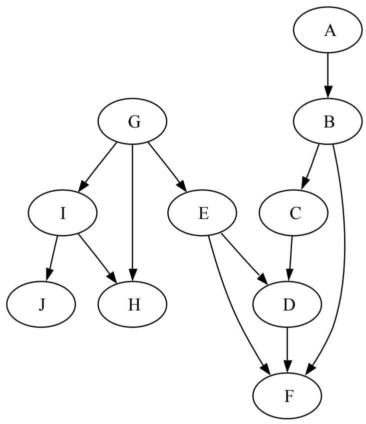

Figure 1: Bayes' net for Problem 1

1. $A \perp F$
2. $A \perp G$
3. $B \perp I \mid F$
4. $D \perp J \mid G, H$
5. $I \perp B \mid H$
6. $J \perp D$
7. $I \perp C \mid H, F$

Solution

As discussed in class, conditional independence properties entailed by a Bayesian Network can be read directly from the graph, using the notion of *d-separation*. Given the Bayesian network in Figure 1, which of the following conditional independence statements hold:

1. $A \perp F$ - Answer: No - Active Trail: ABF

2. $A \perp G$ - Answer: Yes
3. $B \perp I \mid F$ - Answer: No - Active Trail: $BFEGI$
4. $D \perp J \mid G, H$ - Answer: Yes
5. $I \perp B \mid H$ - Answer: Yes
6. $J \perp D$ - Answer: No - Active Trail: $JIGED$
7. $I \perp C \mid H, F$ - Answer: No - Active Trail: $IGEFBC$

2. Bayesian Networks: Variable elimination

Suppose you wish to do variable elimination on the Bayesian Network in Figure 1. Consider the variable ordering $A, B, C, D, E, F, G, H, I, J$. For each iteration of the algorithm (i.e., for each of the variables), determine which factors are removed and introduced. As an example, in the first iteration (variable A), the factor $P(A)$ and $P(B \mid A)$ are removed, and replaced by a new factor $g_1(B)$.

Solution

The elimination proceeds in the following steps

1. Joint probability implied by BN structure: $P(A, \ldots, J) = P(A) \cdot P(B|A) \cdot P(C|B) \cdot P(G) \cdot P(E|G) \cdot P(D|C, E) \cdot P(F|B, E, D) \cdot P(I|G) \cdot P(H|G, I) \cdot P(J|I)$
2. Eliminating A:
$$\begin{aligned} P(B, \ldots, J) = &\sum_a P(a) \cdot P(B|a) \cdot P(C|B) \cdot P(G) \cdot P(E|G) \cdot P(D|C, E) \cdot P(F|B, E, D) \cdot P(I|G) \cdot P(H|G, I) \cdot P(J|I) \\ =&P(C|B) \cdot P(G) \cdot P(E|G) \cdot P(D|C, E) \cdot P(F|B, E, D) \cdot P(I|G) \cdot P(H|G, I) \cdot P(J|I) \cdot \sum_a P(a)P(B|a) \\ =&P(C|B) \cdot P(G) \cdot P(E|G) \cdot P(D|C, E) \cdot P(F|B, E, D) \cdot P(I|G) \cdot P(H|G, I) \cdot P(J|I) \cdot g_1(B) \end{aligned}$$
3. Eliminating B: $P(C, \ldots, J) = P(G) \cdot P(E|G) \cdot P(D|C, E) \cdot P(I|G) \cdot P(H|G, I) \cdot P(J|I) \cdot g_2(C, F, E, D)$
4. Eliminating C: $P(D, \ldots, J) = P(G) \cdot P(E|G) \cdot P(I|G) \cdot P(H|G, I) \cdot P(J|I) \cdot g_3(F, E, D)$
5. Eliminating D: $P(E, \ldots, J) = P(G) \cdot P(E|G) \cdot P(I|G) \cdot P(H|G, I) \cdot P(J|I) \cdot g_4(F, E)$
6. Eliminating E: $P(F, \ldots, J) = P(G) \cdot P(I|G) \cdot P(H|G, I) \cdot P(J|I) \cdot g_5(F, G)$
7. Eliminating F: $P(G, \ldots, J) = P(G) \cdot P(I|G) \cdot P(H|G, I) \cdot P(J|I) \cdot g_6(G)$
8. Eliminating G: $P(H, I, J) = P(J|I) \cdot g_7(I, H)$
9. Eliminating H: $P(I, J) = P(J|I) \cdot g_8(I)$
10. Eliminating I: $P(J) = g_9(J)$

3. An algorithm for d-separation

In this exercise, you will implement an algorithm for computing d-separation of variables in Bayesian networks. In fact, the algorithm works in the opposite way by finding all variables that are *not* d-separated from a variable of interest. More formally, given a query variable X and a set of observed variables $\mathcal{Y} = \{Y_1, \ldots, Y_n\}$, the algorithm decides which variables could be dependent on X given $\mathcal{Y}$, that is, it returns the set of reachable variables

$$\mathcal{R}(X \mid \mathcal{Y}) = \{Z \mid Z \notin \text{d-sep}(X; Z \mid \mathcal{Y})\}\,.$$

You are provided some skeleton Python code in the `.zip` file accompanying this document. Take the following steps for this exercise.

1. Install the Python dependencies listed in `README.txt`, if your system does not already satisfy them. After that, you should be able to run `demo.py` and produce some plots, albeit wrong ones for now.
2. Implement the missing code in `core.py` marked with `TODO`.
3. If your implementation is correct, you should get correct results for the v-structure of the demo file. You can try out more example networks from `examples_dsep.py` to test your implementation.
4. Now, create the network shown in Figure 2 and answer the following questions:
   (a) Which variables are reachable from Radio, if nothing is observed?
   (b) Which variables are reachable from Radio, if Phone is observed?
   (c) Which variables are reachable from Radio, if both Phone and Earthquake are observed?

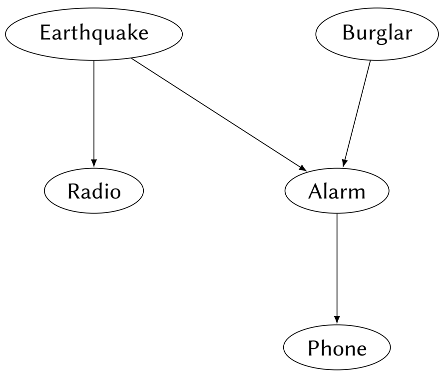

Figure 2: The earthquake network to be implemented.

Solution

For solution please refer to hw3-sol.zip.