Homework 2

CS 6347: Statistical methods in AI/ML

Instructor: Vibhav Gogate

Vibhav.Gogate@utdallas.edu

Student: Xiaodi Li

Net ID: XXL170011

[Problems marked with ** are time consuming. Start early.]

Problem 1: [55 points]**

Write a C++ (or Python or Java) program for performing variable elimination. The program should take as input two files: (a) a Markov network in UAI format and (b) Evidence (namely an assignment of values to some subset of variables) in UAI format and output the partition function given evidence. Your program should eliminate the variables along the min-degree order (ties broken randomly). The UAI format is described here:

http://www.hlt.utdallas.edu/~vgogate/uai16-evaluation/uaiformat.html

Recall that the variable elimination algorithm has three steps:

1. Instantiate Evidence (Reduce the CPTs or factors).
2. Order the variables. (You are using the min-degree order for the purpose of this assignment)
3. Eliminate variables one by one along the order. Recall that to eliminate a variable X, we compute a product of all functions that mention X. Let us call the new function f. Then, we sum-out the variable from f to yield a new function f'. Then we replace all functions that mention X with f'.

Thus, following the divide and conquer approach to programming, you can first develop the following helper functions and then put them together into a variable elimination algorithm.

(Note that either you can use the following approach or develop your own. The following might be easier).

1. Function Read: Create a class called Graphical Model and create an object of this class from the given UAI file
2. Function Order: Compute a min-degree ordering over the non-evidence variables (note that you only have to eliminate only the non-evidence variables if you instantiate the evidence properly).
3. Function Instantiate: Take a factor φ and evidence as input and instantiate evidence in the factor.

4. Function Product: Take two factors φ_1 and φ_2 as input and output $\varphi_3 = \varphi_1 \times \varphi_2$.
5. Function Sum-out: Take a factor φ and a set of variables Y as input and output $\varphi_1 = \sum_y \varphi$.

To efficiently implement the operations describe above, see Box 10.A, pages 358-361 in Koller and Friedman.

How to test your code?. A number of networks along with their correct probability of evidence or partition function values are available on the course website.

What to turn in? Source code and a README file on how to compile/execute your code. Submit using E-learning. Please don't email me your code.

Answer: See Problem1.py and README.txt

1.uai

```
D:\LXD\UTD\Statistical Methods in AI and Machine Learning\
1.py ./homework2_files/1.uai ./homework2_files/1.uai.evid
Elimination order: [1, 2, 3, 6, 7, 8]
Partition function: 14.889866514255774
```

2.uai

```
D:\LXD\UTD\Statistical Methods in AI and Machine Learning\HW\HW2\Homework2
1.py ./homework2_files/2.uai ./homework2_files/2.uai.evid
Elimination order: [0, 3, 12, 15, 1, 4, 7, 13, 2, 5, 6, 8, 9, 10, 11, 14]
Partition function: 44.44954357211642
```

3.uai

```
D:\LXD\UTD\Statistical Methods in AI and Machine Learning\HW\HW2\Homework2>python Problem
1.py ./homework2_files/3.uai ./homework2_files/3.uai.evid
Elimination order: [0, 11, 22, 33, 44, 55, 66, 77, 88, 99, 110, 111, 112, 113, 114, 115,
116, 117, 118, 119, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21
, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 45, 46,
 47, 48, 49, 50, 51, 52, 53, 54, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 67, 68, 69, 70,
71, 72, 73, 74, 75, 76, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 89, 90, 91, 92, 93, 94, 9
5, 96, 97, 98, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109]
Partition function: 163.21432208759873
```

Problem 2: [5 points] Do Exercise 6.5. from AD

Consider a naïve Bayes structure with edges $X \to Y_1, \ldots, X \to Y_n$.

(a) What is the width of variable order $Y_1, \ldots, Y_n, X$?
(b) What is the width of variable order $X, Y_1, \ldots, Y_n$?

Answer:

Algorithm 4 `OrderWidth`($\mathcal{N}, \pi$)

input:

$\mathcal{N}$: Bayesian network

π: ordering of the variables in network $\mathcal{N}$

output: the width of elimination order π

main:

```
G ← interaction graph of the CPTs in network N
w ← 0
for i = 1 to length of elimination order π do
    w ← max(w, d), where d is the number of π(i)'s neighbors in G
    add an edge between every pair of non-adjacent neighbors of π(i) in G
    delete variable π(i) from G
end for
return w
```

(a) According to Algorithm 4 OrderWidth in Chapter 6.6 of AD as shown above, the width of variable order $Y_1, \ldots, Y_n, X$ will be 1.

(b) According to Algorithm 4 OrderWidth in Chapter 6.6 of AD as shown above, the width of variable order $X, Y_1, \ldots, Y_n$ will be n.

Problem 3: [5 points] Do Exercise 6.9 from AD

Compute an elimination order for the variables in Figure 6.11 using the min-degree method. In case of a tie, choose variables that come first alphabetically.

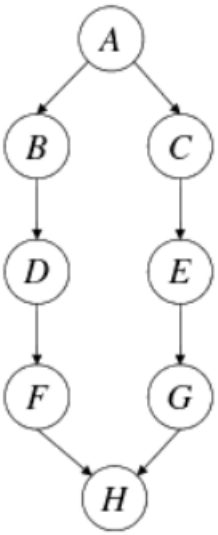


Figure 6.11: A Bayesian network structure.

Answer:

Algorithm 5 `MinDegreeOrder`($\mathcal{N}$, $\mathbf{X}$)

input:

$\mathcal{N}$: Bayesian network

$\mathbf{X}$: variables in network $\mathcal{N}$

output: an ordering π of variables $\mathbf{X}$

main:

1: $G \leftarrow$ interaction graph of the CPTs in network $\mathcal{N}$
2: **for** $i = 1$ to number of variables in $\mathbf{X}$ **do**
3: $\pi(i) \leftarrow$ a variable in $\mathbf{X}$ with smallest number of neighbors in G
4: add an edge between every pair of non-adjacent neighbors of $\pi(i)$ in G
5: delete variable $\pi(i)$ from G and from $\mathbf{X}$
6: **end for**
7: **return** π

According to Algorithm 5 in Chapter 6.6 of AD as shown above, the ordering π of variables X is A, B, C, D, E, F, G, H.

Problem 4: [5 points] Do Exercise 6.10 from AD

What is the treewidth of the network in Figure 6.11? Justify your answer.

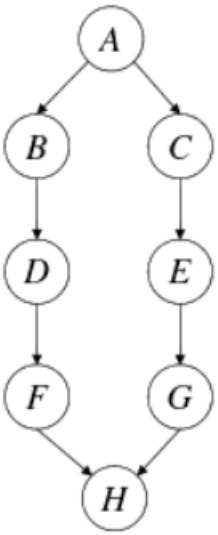


Figure 6.11: A Bayesian network structure.

Answer:

The treewidth of the network is 2. Because each node has two neighbors and those two neighbors are disconnected. Thus, no matter which elimination order we choose, we will get the same treewidth. Namely, the width of the best complete elimination order (i.e., the one with the smallest width) will be 2.

Problem 5: [15 points] Consider the Bayesian network given in Figure 1.

- Convert this Bayesian network into an equivalent Markov network. Convert the resulting Markov network into an equivalent Bayesian network.
- Let H be evidence variables. Trace the operations of Bucket elimination for computing $Pr(H = h)$ along the order (A, E, B, C, D, F, G).
- Is the ordering (A, E, B, C, D, F, G) optimal? What is the treewidth of this network (assume that H is an evidence variable and so the resulting network does not contain H)?
- Construct a tree decomposition for this network (again assume that H is an evidence variable). Show how the junction tree propagation algorithm will operate on this tree decomposition. Show the expression for each message.

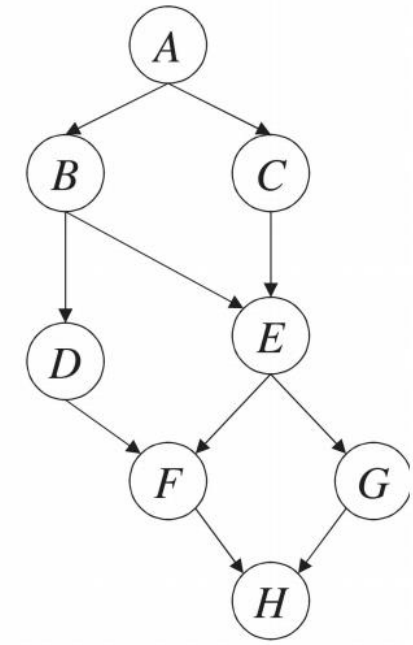
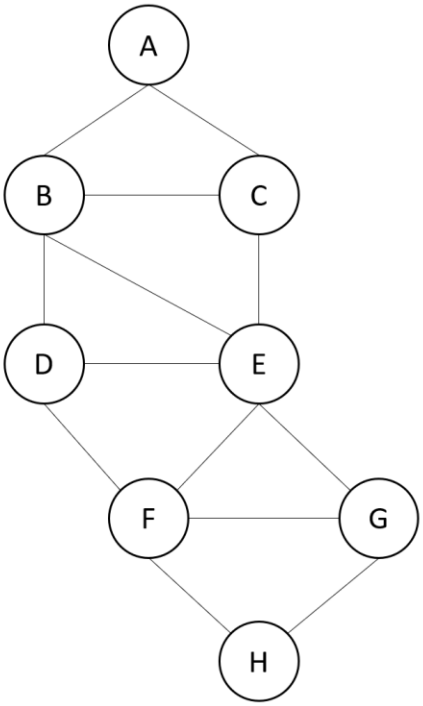


Figure 1: A Bayesian network

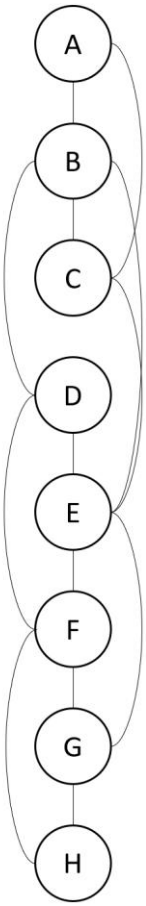
Answer:

- Bayesian network into an equivalent Markov network (connect all parents of a node to each other and remove directionality):

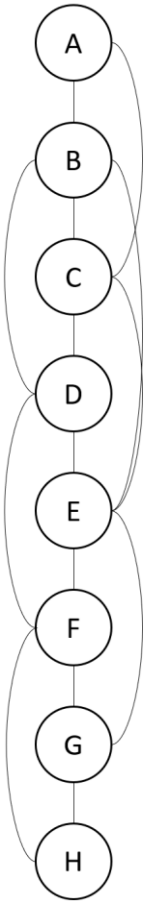


Markov network into an equivalent Bayesian network (connect child nodes of a node of the ordered graph, and add arrows from bottom to up):

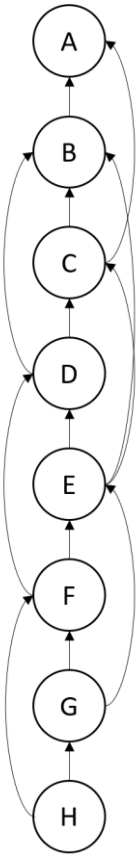
Step 1: draw the Markov network in the order A, B, C, D, E, F, G, H



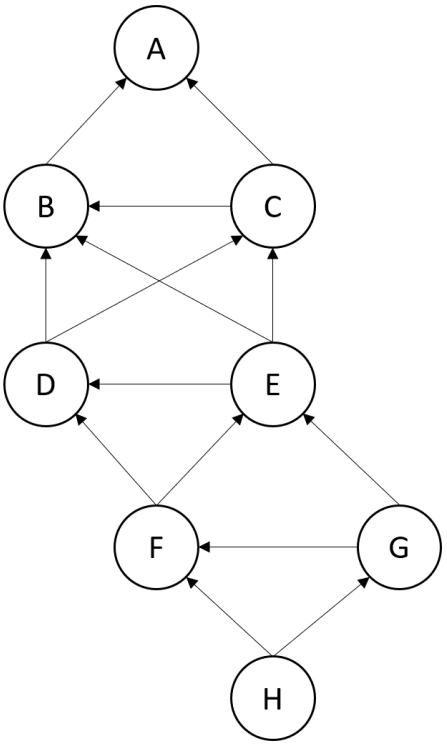
Step 2: connect chile nodes of a node of the ordered graph



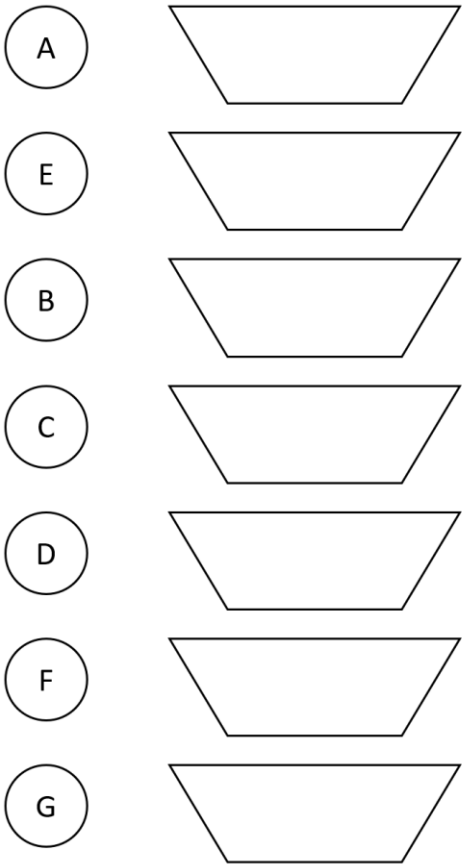
Step 3: add arrows from bottom to up



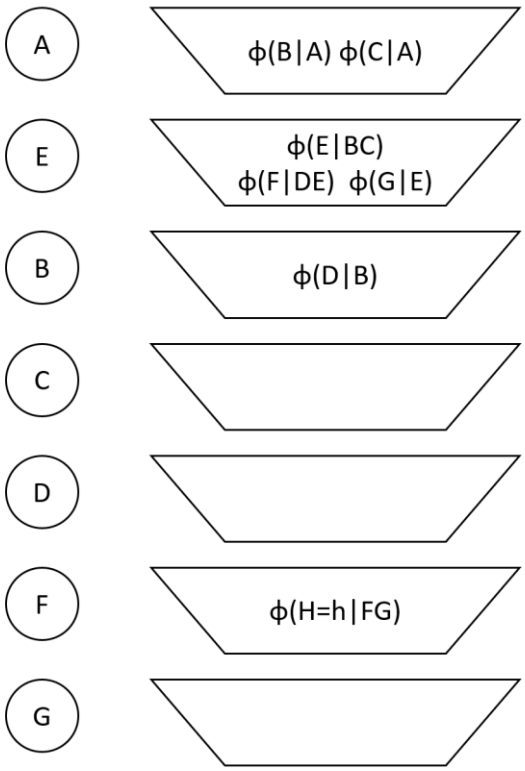
Step 4: convert the graph into the previous format



- Let H be evidence variables. Trace the operations of Bucket elimination for computing $Pr(H = h)$ along the order (A, E, B, C, D, F, G).
  Initialization:



  Processing Buckets:



$$P(H=h)$$
$$= \sum_G \sum_F \sum_D \sum_C \sum_B \sum_E \sum_A \phi(B|A)\phi(C|A)\phi(E|BC)\phi(F|DE)\phi(G|E)\phi(D|B)\phi(H=h|FG)$$
$$= \sum_G \sum_F \phi(H=h|FG) \sum_D \sum_C \sum_B \phi(D|B) \sum_E \phi(E|BC)\phi(F|DE)\phi(G|E) \sum_A \phi(B|A)\phi(C|A)$$

Let $\psi_A(B,C) = \sum_A \phi(B|A)\phi(C|A)$. Then:

$$P(H=h)$$
$$= \sum_G \sum_F \phi(H=h|FG) \sum_D \sum_C \sum_B \phi(D|B)\psi_A(B,C) \sum_E \phi(E|BC)\phi(F|DE)\phi(G|E)$$

Let $\psi_E(B,C,D,F,G) = \sum_E \phi(E|BC)\phi(F|DE)\phi(G|E)$. Then:

$$P(H=h) = \sum_G \sum_F \phi(H=h|FG) \sum_D \sum_C \sum_B \phi(D|B)\psi_A(B,C)\psi_E(B,C,D,F,G)$$

Let $\psi_B(C,D,F,G) = \sum_B \phi(D|B)\psi_A(B,C)\psi_E(B,C,D,F,G)$. Then:

$$P(H=h) = \sum_G \sum_F \phi(H=h|FG) \sum_D \sum_C \psi_B(C,D,F,G)$$

Let $\psi_C(D,F,G) = \sum_C \psi_B(C,D,F,G)$. Then:

$$P(H=h) = \sum_G \sum_F \phi(H=h|FG) \sum_D \psi_C(D,F,G)$$

Let $\psi_D(F,G) = \sum_D \psi_C(D,F,G)$. Then:

$$P(H=h)=\sum_{G}\sum_{F}\phi(H=h|FG)\psi_D(F,G)$$

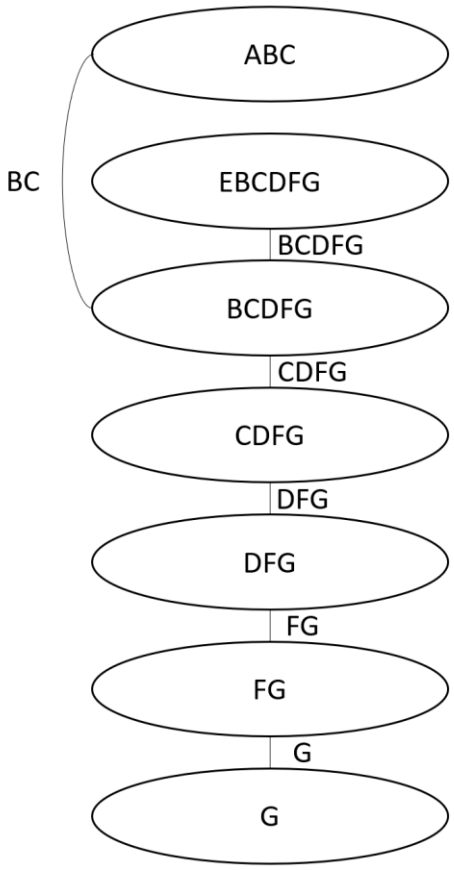
Let $\psi_F(G,H=h)=\sum_F \phi(H=h|FG)\psi_D(F,G)$. Then:

$$P(H=h)=\sum_{G}\psi_F(G,H=h)$$

Let $\psi_G(H=h)=\sum_G \psi_F(G,H=h)$. Thus:

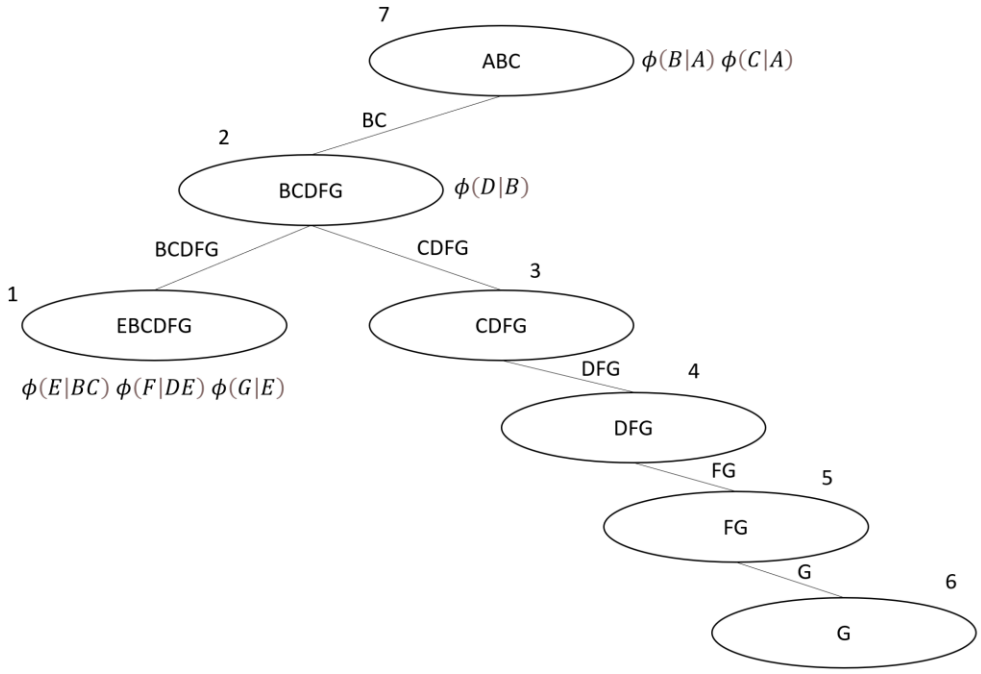
$$P(H=h)=\psi_G(H=h)$$

- Is the ordering (A,E,B,C,D,F,G) optimal? What is the treewidth of this network (assume that H is an evidence variable and so the resulting network does not contain H)?
  No, the ordering (A,E,B,C,D,F,G) is not optimal. Because we can find a better ordering like (A,C,B,D,E,F,G) with treewidth as 2. The treewidth of this network is 4.
- Construct a tree decomposition for this network (again assume that H is an evidence variable). Show how the junction tree propagation algorithm will operate on this tree decomposition. Show the expression for each message.
  Tree decomposition:



Junction tree propagation algorithm:

Step 1 (Select “ABC” as the root):



Step 2 (Pass messages from leaves to root):

$m_{1\to2}(BCDFG) = \sum_E \phi(E|BC)\phi(F|DE)\phi(G|E)$

$m_{6\to5}(G) = 1$

$m_{5\to4}(FG) = 1$

$m_{4\to3}(DFG) = 1$

$m_{3\to2}(CDFG) = 1$

$m_{2\to7}(BC) = \sum_G \sum_F \sum_D m_{1\to2}(BCDFG) * m_{3\to2}(CDFG) * \phi(D|B)$

Step 3 (Pass messages from root to leaves):

$m_{7\to2}(BC) = \sum_A \phi(B|A)\phi(C|A)$

$m_{2\to1}(BCDFG) = m_{7\to2}(BC) * m_{3\to2}(CDFG) * \phi(D|B)$

$m_{2\to3}(CDFG) = \sum_B m_{2\to1}(BCDFG)$

$m_{3\to4}(DFG) = \sum_C m_{2\to3}(CDFG)$

$m_{4\to5}(FG) = \sum_D m_{3\to4}(DFG)$

$m_{5\to6}(G) = \sum_F m_{4\to5}(FG)$

Problem 6: (15 points) Exercise 5.5 from Koller and Friedman

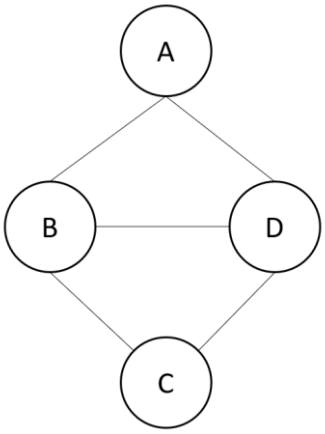
In this exercise, we consider the use of tree-structured local models in the undirected setting.

a. Show how we can use a structure similar to tree-CPDs to represent a factor in a Markov network. What do the values at the leaves of such a tree represent?

b. Given a context $U = u$, define a simple algorithm that takes a tree factor $\phi(Y)$ and returns the reduced factor $\phi[U = u](Y - U)$ (see definition 4.5).
c. The preceding expression takes $Y - U$ to be the scope of the reduced factor. In some cases, it turns out that we can further reduce the scope. Give an example and specify a general rule for when a variable in $Y - U$ can be eliminated from the scope of the reduced tree-factor.

Answer:

a. For example, we are given the Markov Network as follows:



We can use the following table to capture the relations among the four variables A, B, C, and D:

A	B	D	C	$\phi(A, B, D, C)$
0	0	0	0	0.25
0	0	0	1	0.15
0	0	1	0	0.3
0	1	0	0	0.7
1	0	0	0	0.2
0	0	1	1	0.8
0	1	0	1	0.65
1	0	0	1	0.5
0	1	1	0	0.4
1	1	0	0	0.5
1	0	1	0	0.35
0	1	1	1	0.45
1	0	1	1	0.1
1	1	0	1	0.75
1	1	1	0	0.9
1	1	1	1	0.85

Just like tree-CPDs in Bayesian Networks, we have tables to incorporate relations among nodes in Markov Networks. However, different from Bayesian Networks, the values at the leaves dose not define different possible (conditional) distribution over C. It only contains one value representing the probability of the intersection of variable assignments. It only tells us that

configurations with higher values are more likely. Thus, the values do not need to sum to one. What is more, there is no conditioning. It is proportional to the joint distribution of all the variables involved, as opposed to conditional distributions in tree-CPDs.

b. First, we use BFS to traverse the tree. Second, for each node $X \in U$, we connect the parents of X with one child that is consistent with the context u. Finally, we delete X, its children that are not consistent with u, and their subtrees.

c. From the algorithm in question b, we delete entire subtrees in addition to removing from the tree all the nodes in U. By doing this, we also remove from the tree all instances of some other variable Z which is not in U. In this case, we can remove Z from the scope of the reduced tree factor safely because its value no longer depends on the instantiation of Z. Generally speaking, we can remove from the scope of a tree-factor all the variables that do not appear in it.