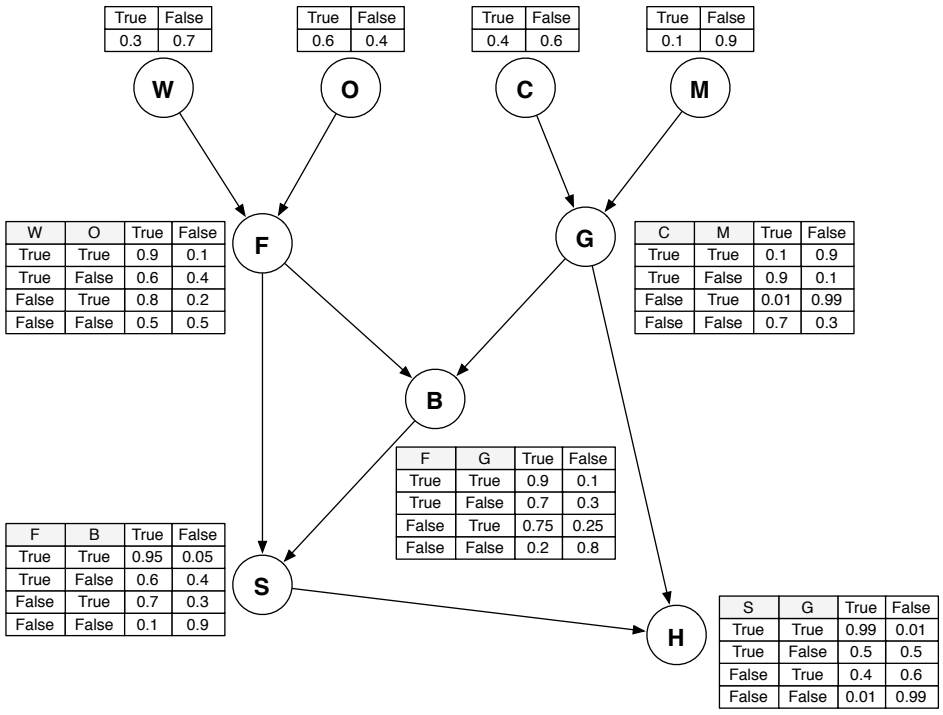
Probabilistic Graphical Models

Solution to HWK#2 (part A)
Assigned Sunday, Feb. 9, 2020
Due: Thursday, March 5, 2020

Problem 1 Variable Elimination:



True	False
0.3	0.7

True	False
0.6	0.4

True	False
0.4	0.6

True	False
0.1	0.9

W	O	True	False
True	True	0.9	0.1
True	False	0.6	0.4
False	True	0.8	0.2
False	False	0.5	0.5

C	M	True	False
True	True	0.1	0.9
True	False	0.9	0.1
False	True	0.01	0.99
False	False	0.7	0.3

F	G	True	False
True	True	0.9	0.1
True	False	0.7	0.3
False	True	0.75	0.25
False	False	0.2	0.8

F	B	True	False
True	True	0.95	0.05
True	False	0.6	0.4
False	True	0.7	0.3
False	False	0.1	0.9

S	G	True	False
True	True	0.99	0.01
True	False	0.5	0.5
False	True	0.4	0.6
False	False	0.01	0.99

Figure 1: Football network

After two losses in three games to start the season, the world champion Pittsburgh Steelers seem to be in a world of trouble. Not knowing what else to do, Coach Cowher has enlisted your help in trying to turn things around during the bye week. In Figure 1, you will find a Bayes net that hopefully contains the key to a successful season. The variables of interest are: Weather (W), Opponent (O), Fans (F), Big Ben (B), Steelers Win (S), Good Health (G), Chunky Soup (C), Motorcycle (M), and Happy (H).

Coach Cowher would like to know the answers to the following questions, but keep in mind, he does not like to be kept waiting. There's only one thing he hates more than losing, and that is exponential computational complexity.

1. What is the chance that the Steelers win given that Big Ben eats his Chunky Soup? [$P(S = T|C = T) = ?$]
2. How fired up are the fans given that Big Ben is in good health? [$P(F = T|G = T) = ?$]
3. $P(M = T|G = T) = ?$
4. $P(M = T|G = T, S = T) = ?$
5. $P(W = T|G = T, B = F, S = T) = ?$

Additionally, report the ordering used and the factors produced after eliminating each variable for the first query $[P(S = T|C = T)]$.

Solution:

Query 1:

Order: H, C, M, W, O, G, F, B

$$P(H, C, M, W, O, F, G, B, S)$$
$$= P(W)P(O)P(C)P(M)P(F|W, O)P(G|C, M)P(B|F, G)P(S|F, B)P(H|S, G)$$

1. Before elimination note, C can only be 1. Also, we use 1 and 0 interchangeably with True and False respectively.
2. First, we eliminate H. The message sums to 1.

$$\sum_{H\in\{0,1\}} P(W)P(O)P(C)P(M)P(F|W, O)P(G|C, M)P(B|F, G)P(S|F, B)P(H|S, G)$$
$$= P(W)P(O)P(M)P(F|W, O)P(G|C, M)P(B|F, G)P(S|F, B)P(C)\sum_{H\in\{0,1\}} P(H|S, G)$$

Thus, $m_H(S, G) = 1$.

3. Next, we eliminate C which can only be 1 as per our evidence.

$$\sum_{C\in\{1\}} P(W)P(O)P(C)P(M)P(F|W, O)P(G|C, M)P(B|F, G)P(S|F, B)$$
$$= P(W)P(O)P(M)P(F|W, O)P(B|F, G)P(S|F, B)\sum_{C\in\{1\}} P(C)P(G|C, M)$$

Thus, $m_C(G, M) = \sum_{C\in\{1\}} P(C)P(G|C, M)$

	$m_C(G,M)$
$G = True, M = True$	$P(C=1)P(G=1\|C=1,M=1) = 0.4*0.1 = 0.04$
$G = True, = False$	$P(C=1)P(G=1\|C=1,M=0) = 0.4*0.9 = 0.36$
$G = False, = True$	$P(C=1)P(G=0\|C=1,M=1) = 0.4*0.9 = 0.36$
$G = False, = False$	$P(C=1)P(G=0\|C=1,M=0) = 0.4*0.1 = 0.04$

4. Next, we eliminate M.

$$\sum_{M\in\{0,1\}} P(W)P(O)P(F|W,O)P(B|F,G)P(S|F,B)P(M)m_C(G,M)$$
$$= P(W)P(O)P(F|W,O)P(B|F,G)P(S|F,B)\sum_{M\in\{0,1\}} P(M)m_C(G,M)$$

Thus, $m_M(G) = \sum_{M\in\{0,1\}} P(M)m_C(G,M)$

	$m_M(G)$
$G = True$	0.1*0.04+0.9*0.36=0.328
$G = False$	0.1*0.36+0.9*0.04=0.072

5. Next, we eliminate W

$$\sum_{W\in\{0,1\}} P(W)P(O)P(F|W,O)P(B|F,G)P(S|F,B)m_M(G)$$
$$= P(O)P(B|F,G)P(S|F,B)m_M(G)m_W(O,F)$$
$$m_W(O,F) = \sum_{W\in\{0,1\}} P(W)P(F|W,O)$$

	$m_W(O,F)$
$O = True, F = True$	0.3*0.9+0.7*0.8=0.83
$O = True, F = False$	0.3*0.1+0.7*0.2=0.17
$O = False, F = True$	0.3*0.6+0.7*0.5=0.53
$O = False, F = False$	0.3*0.4+0.7*0.5=0.47

6. Next, we eliminate O

$$\sum_{O\in\{0,1\}} P(O)P(B|F,G)P(S|F,B)m_M(G)m_W(O,F)$$
$$= P(B|F,G)P(S|F,B)m_M(G)m_O(F)$$
$$m_O(F) = \sum_{O\in\{0,1\}} P(O)m_W(O,F)$$

	$m_O(F)$
$F = True$	0.83*0.6+0.53*0.4 = 0.71
$F = False$	0.17 *0.6+0.47*0.4 = 0.29

7. Next, we eliminate G

$$\sum_{G\in\{0,1\}} P(B|F,G)P(S|F,B)m_M(G)m_O(F) = m_O(F)P(S|F,B)m_G(B,F)$$

$$m_G(B,F) = \sum_{G\in\{0,1\}} m_M(G)P(B|F,G)$$

	$m_G(B,F)$
$B = True, F = True$	0.328*0.9+0.072*0.7=0.3456
$B = True, F = False$	0.328*0.75 +0.072*0.2=0.2604
$B = False, F = True$	0.328*0.1+0.072*0.3=0.0544
$B = False, F = False$	0.328*0.25+0.072*0.8=0.1396

8. Next, we eliminate B

$$\sum_{B\in\{0,1\}} m_O(F)P(S|F,B)m_G(B,F) = m_O(F)m_B(S,F)$$

$$m_B(S,F) = \sum_{B\in\{0,1\}} P(S|F,B)m_G(B,F)$$

	$m_B(S,F)$
$S = True, F = True$	0.95*0.3456+0.6*0.0544=0.3610
$S = True, F = False$	0.7*0.2604+0.1*0.1396=0.1962
$S = False, F = True$	0.05*0.3456+0.4*0.0544=0.0390
$S = False, F = False$	0.3*0.2604+0.9*0.1396=0.2038

9. Next, we eliminate F

$$\sum_{F\in\{0,1\}} m_O(F)m_B(S,F) = m_F(S)$$

	$m_F(S)$
$S = True$	0.71*0.3610+0.29*0.1962=0.3132
$S = False$	0.71*0.0390+0.29*0.2038=0.0867

10. Finally, we normalize to get the required solution to the query:

$$P(S = True|C = True) = \frac{m_F(S = True)}{m_F(S = True) + m_F(S = False)} = 0.783$$

Query 2: 0.7099

Query 3: 0.0065

Query 4: 0.0065

Query 5: 0.3147

Problem 2 (HMM)

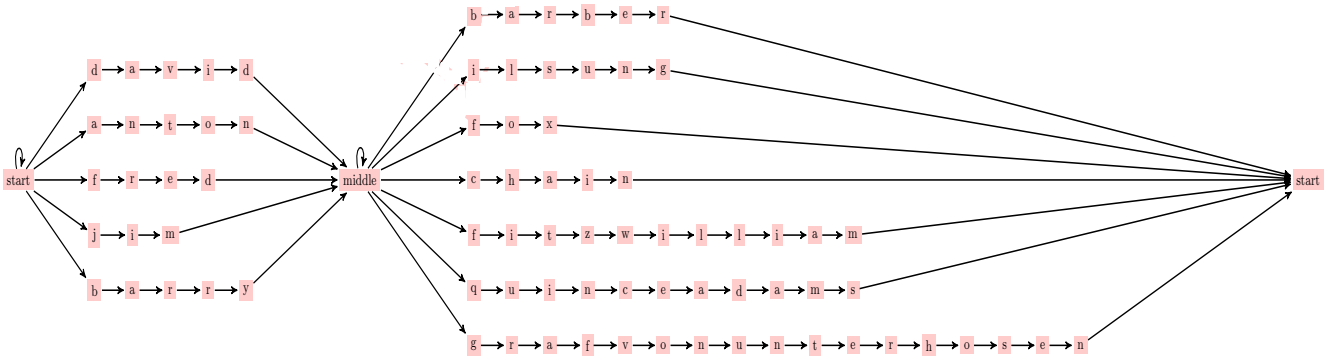
There are a set of firstnames: david, anton, fred, jim, barry which are numbered 1 (david) to 5 (barry) and a set of surnames: barber, ilsung, fox, chain, fitzwilliam, quinceadams, grafvonunterhosen, which are numbered 1 (barber) to 7 (grafvonunterhosen). A string is generated by first randomly sampling a character from a to z. Then we either continue this random character sampling with probability 0.8, or we start to generate a firstname. A firstname is chosen uniformly at random. After generating a firstname, we generate a character at random. We continue to generate another letter at random with probability 0.8, or start to generate a surname (chosen uniformly from the set of surnames). We continue until the last letter of the surname is generated. The process then goes back to start (unless we are at the end timepoint T = 10000). For example, we might generate then dtyjimdfilsungffdavidmjfox.... The catch is that the process of character generation is very noisy. The character we want to generate is only generated correctly with probability 0.3, with probability 0.7 that another character (uniformly at random) will be generated. Answer the following problem:

Given 10000 character sequence in the file ``noisystring.mat'', you must decode the ``noisystring' to find the most likely intended 'clean' sequence. Once you have this sequence, you will be able to find a set of (firstname,surname) pairs as you traverse along the clean sequence. Construct a matrix m(i,j) with the counts of the number of occurrences of the pair (firstname(i),surname(j)) in your clean sequence and display/print this matrix as your final answer in your solution.

You must provide your code as well.

Solution

This can be addressed by using a HMM in which we define a state for each character in all the first names and surnames, with an additional two states, giving the state-transition diagram as depicted in below.



The start and middle states self-transition with probability 0.8. The transitions into the names are uniform and the transitions within and out of the names are deterministic. This defines a sparse 84 by 84 transition matrix A for the HMM. The 26 by 84 dimensional emission matrix has an entry 0.3 for emitting the correct letter, with a uniform probability on the other letters. The start and middle states emit a letter with uniform probability. This then defines the HMM, for which running Viterbi decoding on the T = 10, 000 length sequence and computing the matrix m gives:

```
            j:          [0          2]
ns =

     7     4     8    11    19     9    11
     9     7     5     6    15    10    12
     8     8     8    12    10     8    19
    14    12    12    16    20    15    19
    10     5     9    14     6     9    13
```

See file:

“solutionFirstnameSurname.m”

Problem 3 Maximum Probable Explanation

Consider a distribution defined over binary variables:

$$P(a, b, c) \equiv P(a|b)P(b|c)P(c)$$

with

$$P(a = tr \mid b = tr) = 0.3 \ , \ P(a = tr \mid b = fa) = 0.2 \ , \ P(b = tr \mid c = tr) = 0.75$$

$$P(b = tr \mid c = fa) = 0.1 \ , \qquad P(c = tr) = 0.4,$$

What is the most likely joint configuration? That is $\text{argmax}_{a,b,c} P(a, b, c)$?

Hint: It is not acceptable to just naively work out the 8 possible states to solve the problem. Naïve approach is only feasible for this example but not for a general problem with many variables. Instead, you must use the max-product algorithm to solve this problem.

Solution:

$$\gamma_c(b) \equiv \max_c p(b|c)p(c)$$

For the state $b = \mathsf{tr}$,

$$p(b = \mathsf{tr}|c = \mathsf{tr})p(c = \mathsf{tr}) = 0.75 \times 0.4, \qquad p(b = \mathsf{tr}|c = \mathsf{fa})p(c = \mathsf{fa}) = 0.1 \times 0.6$$

Hence, $\gamma_c(b = \mathsf{tr}) = 0.75 \times 0.4 = 0.3$. Similarly, for $b = \mathsf{fa}$,

$$p(b = \mathsf{fa}|c = \mathsf{tr})p(c = \mathsf{tr}) = 0.25 \times 0.4 \qquad p(b = \mathsf{fa}|c = \mathsf{fa})p(c = \mathsf{fa}) = 0.9 \times 0.6$$

Hence, $\gamma_c(b = \mathsf{fa}) = 0.9 \times 0.6 = 0.54$.

We now consider

$$\gamma_b(a) \equiv \max_b p(a|b)\gamma_c(b)$$

For $a = \mathsf{tr}$, the state $b = \mathsf{tr}$ has value

$$p(a = \mathsf{tr}|b = \mathsf{tr})\gamma_c(b = \mathsf{tr}) = 0.3 \times 0.3 = 0.09$$

and state $b = \mathsf{fa}$ has value

$$p(a = \mathsf{tr}|b = \mathsf{fa})\gamma_c(b = \mathsf{fa}) = 0.2 \times 0.54 = 0.108$$

Hence $\gamma_b(a = \mathsf{tr}) = 0.108$. Similarly, for $a = \mathsf{fa}$, the state $b = \mathsf{tr}$ has value

$$p(a = \mathsf{fa}|b = \mathsf{tr})\gamma_c(b = \mathsf{tr}) = 0.7 \times 0.3 = 0.21$$

and state $b = \mathsf{fa}$ has value

$$p(a = \mathsf{fa}|b = \mathsf{fa})\gamma_c(b = \mathsf{fa}) = 0.8 \times 0.54 = 0.432$$

giving $\gamma_b(a = \mathsf{fa}) = 0.432$. Now we can compute the optimal state

$$a^* = \operatorname*{argmax}_a \gamma_b(a) = \mathsf{fa}$$

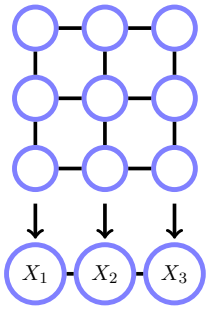
Given this optimal state, we can backtrack, giving

$$b^* = \operatorname*{argmax}_b p(a = \mathsf{fa}|b)\gamma_c(b) = \mathsf{fa}, \quad c^* = \operatorname*{argmax}_c p(b = \mathsf{fa}|c)p(c) = \mathsf{fa}$$

Note that in the backtracking process, we already have all the information required from the computation of the messages γ.

Problem 4. (Markov Network)

Consider a binary variable Markov Random Field $p(x) = Z^{-1} \prod_{i>j} \phi(x_i, x_j)$, *defined on the* $n \times n$ *lattice with* $\phi(x_i, x_j) = e^{\mathbb{I}[x_i = x_j]}$ *for* i *a neighbour of* j *on the lattice and* $i > j$. *A naive way to perform inference is to first stack all the variables in the* t^{th} *column and call this cluster variable* X_t, *as shown. The resulting graph is then singly connected. What is the complexity of computing the normalisation constant based on this cluster representation? Compute* $\log Z$ *for* $n = 10$.



You need to provide your software code for answering log(Z) for n=10. Give also your answer for log(Z) so that we do not need to run your code.

Solution:

Complexity is O(n 2^{2n}).

For ***n = 10***, we have: ***log Z = 186.7916***

See the code:

"IsingZ.m"

Problem 5

The file 'diseaseNet.mat' contains the potentials for a disease bi-partite belief network, with 20 diseases $d_1, \ldots, d_{20}$ and 40 symptoms, $s_1, \ldots, s_{40}$. The disease variables are numbered from 1 to 20 and the symptoms from 21 to 60. Each disease and symptom is a binary variable, and each symptom connects to 3 parent diseases.

1. Using the BRMLtoolbox, construct a junction tree for this distribution and use it to compute all the marginals of the symptoms, i.e., $p(s_i = 1)$. In addition to providing your code, you must display/print the results in your solution paper work.

2. Explain how to compute the marginals $p(s_i = 1)$ (in the previous part) in a way more efficient than using the junction tree formalism. By implementing this method, compare it with the results from the junction tree algorithm from the previous part.

3. Symptoms 1 to 5 are present (state 1), symptoms 6 to 10 not present (state 2), and the rest not known. Compute the marginal $p(d_i = 1 | s_{1:10})$ for all diseases.

In all above questions, in addition to providing your codes, you must display/print the results in your solution paper work.

Solution:

Part (1) See file "diseaseNet.m"

p(s(1)=1): Junction tree 0.441834 : efficient inference: 0.441834
p(s(2)=1): Junction tree 0.456675 : efficient inference: 0.456675
p(s(3)=1): Junction tree 0.441405 : efficient inference: 0.441405
p(s(4)=1): Junction tree 0.491275 : efficient inference: 0.491275
p(s(5)=1): Junction tree 0.493892 : efficient inference: 0.493892
p(s(6)=1): Junction tree 0.657483 : efficient inference: 0.657483
p(s(7)=1): Junction tree 0.504563 : efficient inference: 0.504563
p(s(8)=1): Junction tree 0.268693 : efficient inference: 0.268693
p(s(9)=1): Junction tree 0.649077 : efficient inference: 0.649077
p(s(10)=1): Junction tree 0.49074 : efficient inference: 0.49074
p(s(11)=1): Junction tree 0.422552 : efficient inference: 0.422552
p(s(12)=1): Junction tree 0.429096 : efficient inference: 0.429096
p(s(13)=1): Junction tree 0.545021 : efficient inference: 0.545021
p(s(14)=1): Junction tree 0.632959 : efficient inference: 0.632959
p(s(15)=1): Junction tree 0.42954 : efficient inference: 0.42954
p(s(16)=1): Junction tree 0.458794 : efficient inference: 0.458794
p(s(17)=1): Junction tree 0.427559 : efficient inference: 0.427559
p(s(18)=1): Junction tree 0.404255 : efficient inference: 0.404255
p(s(19)=1): Junction tree 0.582093 : efficient inference: 0.582093
p(s(20)=1): Junction tree 0.589591 : efficient inference: 0.589591
p(s(21)=1): Junction tree 0.76127 : efficient inference: 0.76127
p(s(22)=1): Junction tree 0.695588 : efficient inference: 0.695588
p(s(23)=1): Junction tree 0.508702 : efficient inference: 0.508702
p(s(24)=1): Junction tree 0.419962 : efficient inference: 0.419962
p(s(25)=1): Junction tree 0.351942 : efficient inference: 0.351942
p(s(26)=1): Junction tree 0.389611 : efficient inference: 0.389611
p(s(27)=1): Junction tree 0.325973 : efficient inference: 0.325973
p(s(28)=1): Junction tree 0.469624 : efficient inference: 0.469624
p(s(29)=1): Junction tree 0.522868 : efficient inference: 0.522868
p(s(30)=1): Junction tree 0.717312 : efficient inference: 0.717312
p(s(31)=1): Junction tree 0.524199 : efficient inference: 0.524199
p(s(32)=1): Junction tree 0.353704 : efficient inference: 0.353704
p(s(33)=1): Junction tree 0.512679 : efficient inference: 0.512679
p(s(34)=1): Junction tree 0.529404 : efficient inference: 0.529404
p(s(35)=1): Junction tree 0.385751 : efficient inference: 0.385751
p(s(36)=1): Junction tree 0.489095 : efficient inference: 0.489095
p(s(37)=1): Junction tree 0.633595 : efficient inference: 0.633595
p(s(38)=1): Junction tree 0.589604 : efficient inference: 0.589604
p(s(39)=1): Junction tree 0.423164 : efficient inference: 0.423164
p(s(40)=1): Junction tree 0.528234 : efficient inference: 0.528234

Part (2)

The point is that in computing the marginals $p(s_i)$, we can do this efficiently using

$$p(s_i) = \sum_{pa(s_i)} p(s_i \mid pa(s_i)) \cdot p(pa(s_i))$$

where the parent diseases pa(s_i) are independent. Since there are only 3 parental diseases, we can compute these marginals by $2^3 = 8$ summations. In contrast, the Junction Tree has cliques of size 12, meaning that the complexity of carrying out absorption for the JT is at least 2^{12}, which is much more expensive. The point therefore is that the JT doesn't know that there is structure that can be exploited in the tables, and is therefore an upper bound on the complexity of inference.

Note also from solution table that JT is an exact algorithm (as expected) and gives the same answer as efficient algorithm in which you sum over all 3 parents.

Part (3)

4. The marginals p(disease=1|evidence):

p(d(1)=1|s(1:10))=0.0297756

p(d(2)=1|s(1:10))=0.38176

p(d(3)=1|s(1:10))=0.954235

p(d(4)=1|s(1:10))=0.396644

p(d(5)=1|s(1:10))=0.496467

p(d(6)=1|s(1:10))=0.435154

p(d(7)=1|s(1:10))=0.187487

p(d(8)=1|s(1:10))=0.701183

p(d(9)=1|s(1:10))=0.0431266

p(d(10)=1|s(1:10))=0.610313

p(d(11)=1|s(1:10))=0.287322

p(d(12)=1|s(1:10))=0.489833

p(d(13)=1|s(1:10))=0.8996

p(d(14)=1|s(1:10))=0.619565

p(d(15)=1|s(1:10))=0.920476

p(d(16)=1|s(1:10))=0.706096

Problem 6 Clique Trees

Consider the distribution

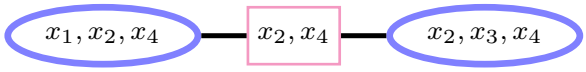
$$p(x_1, x_2, x_3, x_4) = \phi(x_1, x_2)\phi(x_2, x_3)\phi(x_3, x_4)\phi(x_4, x_1)$$

1. *Write down a junction tree for the above distribution.*

2. *Carry out the absorption procedure and demonstrate that this gives the correct result for the marginal* $p(x_1)$.

Solution

Part (1)



Part (2)

Assign

$$\phi(x_1, x_2, x_4) = \phi(x_1, x_2)\phi(x_1, x_4), \text{ and } \phi(x_2, x_3, x_4) = \phi(x_2, x_3)\phi(x_3, x_4).$$

We'll go from left to right with initial separators set to unity:

$$\phi^*(x_2, x_4) = \sum_{x_1} \phi(x_1, x_2, x_4) = \sum_{x_1} \phi(x_1, x_2)\phi(x_1, x_4),$$

$$\phi^*(x_2, x_3, x_4) = \phi(x_2, x_3, x_4)\phi^*(x_2, x_4) = \phi(x_2, x_3)\,\phi(x_3, x_4) \sum_{x_1} \phi(x_1, x_2)\phi(x_1, x_4)$$

$$= \sum_{x_1} p(x_1, x_2, x_3, x_4) = p(x_2, x_3, x_4)$$

Going back from right to left we have

$$\phi^{**}(x_2, x_4) = \sum_{x_3} \phi^{*}(x_2, x_3, x_4) = \sum_{x_3} p(x_2, x_3, x_4) = p(x_2, x_4)$$

$$\phi^{*}(x_1, x_2, x_4) = \frac{\phi(x_1, x_2, x_4)\phi^{**}(x_2, x_4)}{\phi^{*}(x_2, x_4)}$$

$$= \frac{\phi(x_1, x_2, x_4)\sum_{x_3} \phi(x_2, x_3)\phi(x_3, x_4)\sum_{x_1} \phi(x_1, x_2)\phi(x_1, x_4)}{\sum_{x_1} \phi(x_1, x_2)\phi(x_1, x_4)}$$

$$= \sum_{x_3} \phi(x_1, x_2)\phi(x_2, x_3)\phi(x_3, x_4)\phi(x_1, x_4)$$

$$= \sum_{x_3} p(x_1, x_2, x_3, x_4) = p(x_1, x_2, x_4)$$

We can then find

$$\sum_{x_2, x_4} \phi^{**}(x_1, x_2, x_4) = \sum_{x_2, x_4} p(x_1, x_2, x_4) = p(x_1)$$