Probabilistic Foundations of Artificial Intelligence

Solutions to Problem Set 3

Nov 10, 2017

1. Variable elimination

In this exercise you will use variable elimination to perform inference on a bayesian network. Consider the network in figure 1 and its corresponding conditional probability tables (CPTs).

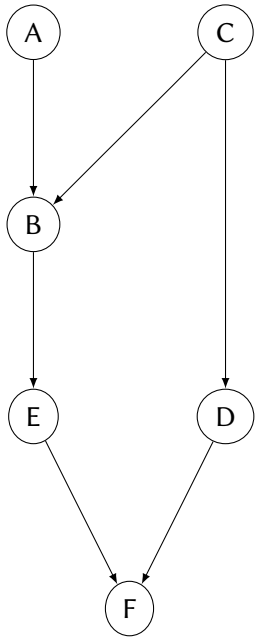


Figure 1: Bayesian network for problem 1.

$$P(A = t) = 0.3 \tag{1}$$
$$P(C = t) = 0.6 \tag{2}$$

Table 1: CPTs for problem 1.

A	C	$P(B = t)$
f	f	0.2
f	t	0.8
t	f	0.3
t	t	0.5

C	$P(D = t)$
f	0.9
t	0.75

B	$P(E = t)$
f	0.2
t	0.4

D	E	$P(F = t)$
f	f	0.95
f	t	1
t	f	0
t	t	0.25

Assuming a query on A with evidence for B and D, i.e. $P(A \mid B, D)$, use the variable elimination algorithm to answer the following queries. Make explicit the selected ordering for the variables and compute the probability tables of the intermediate factors.

1. $P(A = t \mid B = t, D = f)$
2. $P(A = f \mid B = f, D = f)$
3. $P(A = t \mid B = t, D = t)$

Consider now the ordering, C, E, F, D, B, A, use again the variable elimination algorithm and write down the intermediate factors, this time without computing their probability tables. Is this ordering better or worse than the one you used before? Why?

Solution

From the problem statement, we want to calculate $P(\mathbf{Q} \mid \mathbf{E})$ and we know the following:

$$\mathbf{Q} = \{A\} \tag{3}$$
$$\mathbf{E} = \{B, D\} \tag{4}$$
$$\mathbf{X} \backslash \{\mathbf{Q}, \mathbf{E}\} = \{C, E, F\} \tag{5}$$

The variables to eliminate are given in equation 5. A good ordering must be chosen for them, a possibility is: $[F, E, C]$. First let's write down the initial factors derived from the CPTs:

Table 2: Initial factors

A	$g_1(A)$
f	0.7
t	0.3

C	$g_2(C)$
f	0.4
t	0.6

C	D	$g_3(C, D)$
f	f	0.10
f	t	0.90
t	f	0.25
t	t	0.75

B	E	$g_4(B, E)$
f	f	0.8
f	t	0.2
t	f	0.6
t	t	0.4

A	B	C	$g_5(A, B, C)$
f	f	f	0.8
f	f	t	0.2
f	t	f	0.2
f	t	t	0.8
t	f	f	0.7
t	f	t	0.5
t	t	f	0.3
t	t	t	0.5

D	E	F	$g_6(D, E, F)$
f	f	f	0.05
f	f	t	0.95
f	t	f	0.00
f	t	t	1.00
t	f	f	1.00
t	f	t	0.00
t	t	f	0.75
t	t	t	0.25

Eliminate F

$$g_7(D, E) = \sum_f \prod_{j \epsilon \{6\}} g_j = \sum_f g_6(D, E, f) \tag{6}$$

The resulting factor is presented in table 3. Note that this marginalization trivializes the factor $g_7(D, E)$ because the conditional query is independent of F.

Table 3: Intermediate factor $g_7(D, E)$

D	E	$g_7(D, E)$
f	f	$0.05 + 0.95 = 1$
f	t	$0.00 + 1.00 = 1$
t	f	$1.00 + 0.00 = 1$
t	t	$0.75 + 0.25 = 1$

Eliminate E

$$g_8(B, D) = \sum_e g_4(B, e) g_7(D, e) = \sum_e f_1(B, D, e) \tag{7}$$

A product of the factors, $f_1(B, D, E) = g_4(B, E) g_7(D, E)$, is presented in table 4.

Table 4: Product factor $f_1(B, D, E)$

B	D	E	$f_1(B, D, E)$
f	f	f	$1 \times 0.8 = 0.8$
f	f	t	$1 \times 0.2 = 0.2$
f	t	f	$1 \times 0.8 = 0.8$
f	t	t	$1 \times 0.2 = 0.2$
t	f	f	$1 \times 0.6 = 0.6$
t	f	t	$1 \times 0.4 = 0.4$
t	t	f	$1 \times 0.6 = 0.6$
t	t	t	$1 \times 0.4 = 0.4$

After marginalizing E, the resulting factor is in table 5.

Table 5: Intermediate factor $g_8(B, D)$

B	D	$g_8(B, D)$
f	f	1
f	t	1
t	f	1
t	t	1

Therefore, the query is independent of E or F given the evidence variables.

Eliminate C

$$f_2(C, D) = g_2(C)g_3(C, D) \tag{8}$$
$$f_3(A, B, C, D) = f_2(C, D)g_5(A, B, C) \tag{9}$$
$$g_9(A, B, D) = \sum_c g_2(c)g_3(c, D)g_5(A, B, c) = \sum_c f_3(A, B, c, D) \tag{10}$$

Tables 6 and 7 show the product of the factors containing C.

Table 6: Product factor $f_2(C, D)$

C	D	$f_2(C, D)$
f	f	$0.10 \times 0.4 = 0.04$
f	t	$0.90 \times 0.4 = 0.36$
t	f	$0.25 \times 0.6 = 0.15$
t	t	$0.75 \times 0.6 = 0.45$

Table 7: Product factor $f_3(A, B, C, D)$

A	B	C	D	$f_3(A, B, C, D)$
f	f	f	f	$0.04 \times 0.8 = 0.032$
f	f	f	t	$0.36 \times 0.8 = 0.288$
f	f	t	f	$0.15 \times 0.2 = 0.030$
f	f	t	t	$0.45 \times 0.2 = 0.090$
f	t	f	f	$0.04 \times 0.2 = 0.008$
f	t	f	t	$0.36 \times 0.2 = 0.072$
f	t	t	f	$0.15 \times 0.8 = 0.120$
f	t	t	t	$0.45 \times 0.8 = 0.360$
t	f	f	f	$0.04 \times 0.7 = 0.028$
t	f	f	t	$0.36 \times 0.7 = 0.252$
t	f	t	f	$0.15 \times 0.5 = 0.075$
t	f	t	t	$0.45 \times 0.5 = 0.225$
t	t	f	f	$0.04 \times 0.3 = 0.012$
t	t	f	t	$0.36 \times 0.3 = 0.108$
t	t	t	f	$0.15 \times 0.5 = 0.075$
t	t	t	t	$0.45 \times 0.5 = 0.225$

Table 8 shows the resulting factor after marginalizing C.

Answering queries

$$P(a \mid b, d) \propto g_1(a)g_8(b, d)g_9(a, b, d) \tag{11}$$

$$g_1(A = f)g_8(B = t, D = f)g_9(A = f, B = t, D = f) = 0.7 \times 1 \times 0.128 = 0.0896 \tag{12}$$
$$g_1(A = t)g_8(B = t, D = f)g_9(A = t, B = t, D = f) = 0.3 \times 1 \times 0.087 = 0.0261 \tag{13}$$

Table 8: Intermediate factor $g_9(A, B, D)$

A	B	D	$g_9(A, B, D)$
f	f	f	$0.032 + 0.030 = 0.062$
f	f	t	$0.288 + 0.090 = 0.378$
f	t	f	$0.008 + 0.120 = 0.128$
f	t	t	$0.072 + 0.360 = 0.432$
t	f	f	$0.028 + 0.075 = 0.103$
t	f	t	$0.252 + 0.225 = 0.477$
t	t	f	$0.012 + 0.075 = 0.087$
t	t	t	$0.108 + 0.225 = 0.333$

Re-normalizing,

$$P(A \mid B = t, D = f) =< 0.7744, \mathbf{0.2256} > \tag{14}$$

$$g_1(A = f)g_8(B = f, D = f)g_9(A = f, B = f, D = f) = 0.7 \times 1 \times 0.062 = 0.0434 \tag{15}$$
$$g_1(A = t)g_8(B = f, D = f)g_9(A = t, B = f, D = f) = 0.3 \times 1 \times 0.103 = 0.0309 \tag{16}$$

Re-normalizing,

$$P(A \mid B = f, D = f) =< \mathbf{0.5841}, 0.4159 > \tag{17}$$

$$g_1(A = f)g_8(B = t, D = t)g_9(A = f, B = t, D = t) = 0.7 \times 1 \times 0.432 = 0.3024 \tag{18}$$
$$g_1(A = t)g_8(B = t, D = t)g_9(A = t, B = t, D = t) = 0.3 \times 1 \times 0.333 = 0.0999 \tag{19}$$

Re-normalizing,

$$P(A \mid B = t, D = t) =< 0.7517, \mathbf{0.2483} > \tag{20}$$

Another ordering

Let's consider now the ordering C, E, F for variable elimination. The intermediate factors are:

$$g_7(A, B, D) = \sum_c g_2(c) g_3(c, D) g_5(A, B, c) \tag{21}$$

$$g_8(B, D, F) = \sum_e g_4(B, e) g_6(D, e, F) \tag{22}$$

$$g_9(B, D) = \sum_f g_8(B, D, f) \tag{23}$$

Is this ordering better? Note that with the former ordering, only one of the factors has 3 variables, i.e. $g_9(A, B, D)$ in equation 10. On the other hand, with the latter ordering we would have to compute the table for $g_8(B, D, F)$ in equation 22 only to marginalize F in the next step. Therefore, the former ordering is better as it requires less operations to answer the given queries.

2. Belief propagation

In this exercise, you will implement the belief propagation algorithm for performing inference in Bayesian networks. As you have seen in the class lectures, the algorithm is based on converting the Bayesian network to a factor graph and then passing messages between variable and factor nodes of that graph until convergence.

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 `bprop.py` marked with `TODO`. In particular, you have to fill in parts of the two functions that are responsible for sending messages from variable to factor nodes and vice versa, as well as parts of the function that returns the resulting marginal distribution of a variable node after message passing has terminated.

3. Now, set up the full-fledged earthquake network, whose structure was introduced in Problem Set 2 and is shown again in Figure 2. Here is the story behind this network:

   While Fred is commuting to work, he receives a phone call from his neighbor saying that the burglar alarm in Fred's house is ringing. Upon hearing this, Fred immediately turns around to get back and check his home. A few minutes on his way back, however, he hears on the radio that there was an earthquake near his home earlier that day. Relieved by the news, he turns around again and continues his way to work.

   To build up the conditional probability tables (CPTs) for the network of Figure 2 you may make the following assumptions about the variables involved:

   - All variables in the network are binary.

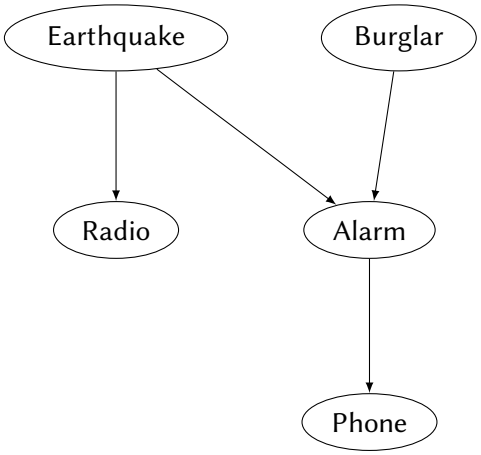


Figure 2: The earthquake network to be implemented.

- As can be seen from the network structure, burglaries and earthquakes are assumed to be independent. Furthermore, each of them is assumed to occur with probability 0.1%.
- The alarm is triggered in the following ways: (1) When a burglar enters the house, the alarm will ring 99% of the time; (2) when an earthquake occurs, there will be a false alarm 1% of the time; (3) the alarm might go off due to other causes (wind, rain, etc.) 0.1% of the time. These three types of causes are assumed to be independent of each other.
- The neighbor is assumed to call only when the alarm is ringing, but only does so 70% of the time when it is actually ringing.
- The radio is assumed to never falsely report an earthquake, but it might fail to report an earthquake that actually happened 50% of the time. (This includes the times that Fred fails to listen to the announcement.)

4. After having set up the network and its CPTs, answer the following questions using your belief propagation implementation:

   (a) Before Fred gets the neighbor's call, what is the probability of a burglary having occurred? What is the probability of an earthquake having occurred?

   (b) How do these probabilities change after Fred receives the neighbor's phonecall?

   (c) How do these probabilities change after Fred listens to the news on the radio?

Solution

The complete solution code can be found in the `.zip` file accompanying this file. Now we show how to derive the CPT from the problem statement and the expected results to check the obtained solution from your code.

Defining the CPTs

First, we show here the derivation of the conditional probability table (CPT) for the alarm variable A. Let us define the following quantities from the problem description:

- $f_b = 0.99$: the probability that the alarm will ring because of a burglar,
- $f_e = 0.01$: the probability that the alarm will ring because of an earthquake,
- $f_o = 0.001$: the probability that the alarm will ring because of other causes.

Using the facts that these three causes are independent of each other, and that the alarm is triggered by either of them, we can compute the following probabilities:

$$P(A = f \mid B = f, E = f) = 1 - f_o = 0.999 \tag{24}$$
$$P(A = f \mid B = f, E = t) = (1 - f_e) \times (1 - f_o) = 0.98901 \tag{25}$$
$$P(A = f \mid B = t, E = f) = (1 - f_b) \times (1 - f_o) = 0.00999 \tag{26}$$
$$P(A = f \mid B = t, E = t) = (1 - f_b) \times (1 - f_e) \times (1 - f_o) = 0.0098901. \tag{27}$$

The CPTs for the alarm variable is presented in table 9 together with the other variables which can be obtained straightforwardly from the description.

Table 9: CPTs for problem 2.

B	E	$P(A = t)$
f	f	0.0010000
f	t	0.0109900
t	f	0.9900100
t	t	0.9901099

A	$P(P = t)$
f	0.0
t	0.7

E	$P(R = t)$
f	0.0
t	0.5

The probabilities of the independent variables are:

$$P(E = t) = 0.001 \tag{28}$$
$$P(B = t) = 0.001 \tag{29}$$

Queries

The correct results for the queries from point 5 are:

1. $P(B = t) = 0.1\%$ and $P(E = t) = 0.1\%$
2. $P(B = t \mid P = t) = 49.5\%$ and $P(E = t \mid P = t) = 0.6\%$
3. $P(B = t \mid P = t, R = t) = 8.3\%$ and $P(E = t \mid P = t, R = t) = 100\%$

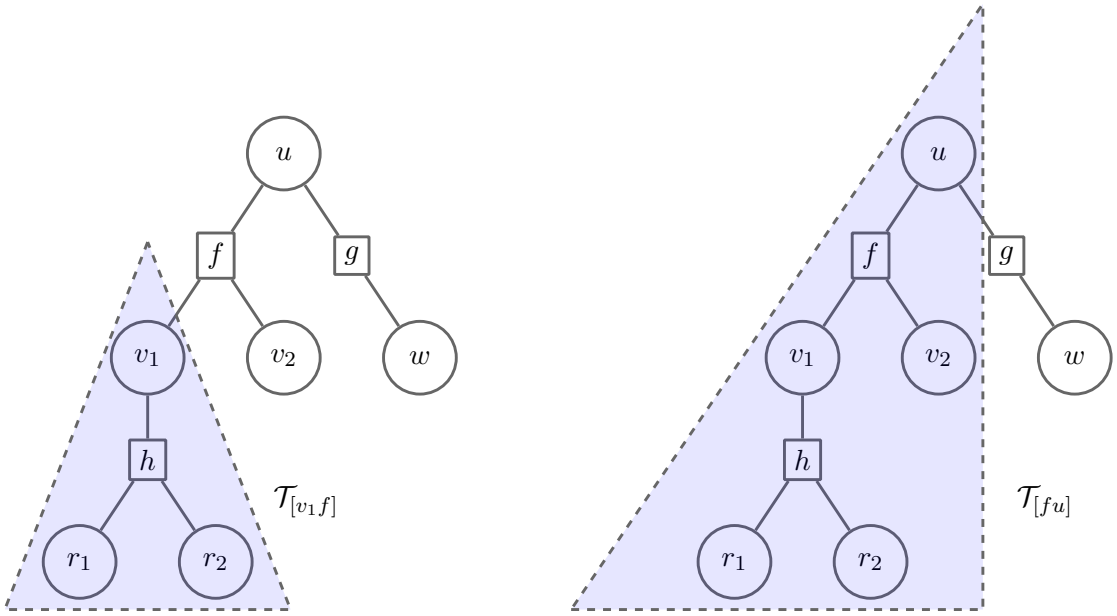


Figure 3: An example factor graph and two of its subtrees.

3. Belief propagation on tree factor graphs*

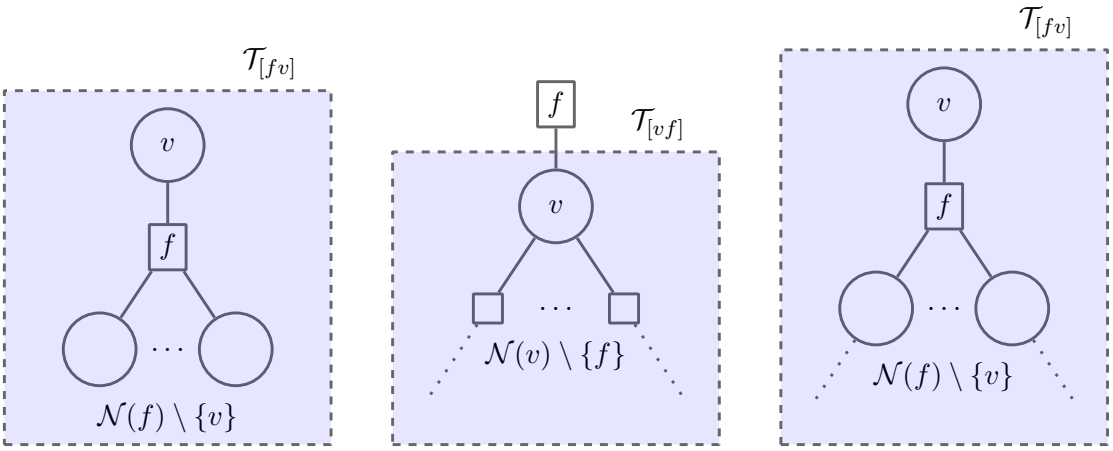
In this exercise we will prove that the belief propagation algorithm converges to the exact marginals after a fixed number of iterations given that the factor graph is a tree, that is, given that the original Bayesian network is a polytree.

We will assume that the factor graph contains no single-variable factors. (You have already seen that if those exist, they can easily be incorporated into multi-variable factors without increasing the complexity of the algorithm.) Since the factor graph is a tree, we will designate a variable node, say a, as the root of the tree. We will consider subtrees $\mathcal{T}_{[rt]}$, where r and t are adjacent nodes in the factor graph and t is closer to the root than r, which are of two types:

- if r is a factor node (and t a variable node), then $\mathcal{T}_{[rt]}$ denotes a subtree that has t as its root, contains the whole subtree under r and, additionally, the edge $\{r, t\}$,
- if r is a variable node (and t a factor node), then $\mathcal{T}_{[rt]}$ denotes the whole subtree under r with r as its root.

See Figure 3 for two example subtrees, one of each type. The depth of a tree is defined as the maximum distance between the root and any other node. Note that, both types of subtrees $\mathcal{T}_{[rt]}$ defined above have always depths that are even numbers.

We will use the subscript notation $[rt]$ to refer to quantities constrained to the subtree $\mathcal{T}_{[rt]}$. In particular, we denote by $\mathcal{F}_{[rt]}$ the set of factors in the subtree and by $P_{[rt]}(x_v)$ the marginal distribution of v when we only consider the nodes of the subtree. More concretely, if r is a



(a) Base case. (b) First induction step. (c) Second induction step.

Figure 4: The three cases considered in the proof by induction.

variable node, by the sum rule we get

$$P_{[rt]}(x_r) \propto \sum_{\boldsymbol{x}_{[rt]\setminus\{r\}}} \prod_{f \in \mathcal{F}_{[rt]}} f(\boldsymbol{x}_f), \tag{1}$$

where $\propto$ denotes equality up to a normalization constant.

Remember the form of the messages passed between variable and factor nodes at each iteration of the algorithm:

$$\mu_{v\to f}^{(t+1)}(x_v) := \prod_{g \in \mathcal{N}(v)\setminus\{f\}} \mu_{g\to v}^{(t)}(x_v) \tag{2}$$

$$\mu_{f\to v}^{(t+1)}(x_v) := \sum_{\boldsymbol{x}_{f\setminus\{v\}}} f(\boldsymbol{x}_f) \prod_{w \in \mathcal{N}(f)\setminus\{v\}} \mu_{w\to f}^{(t)}(x_w). \tag{3}$$

We also define the estimated marginal distribution of variable v at iteration t as

$$\hat{P}^{(t)}(x_v) := \prod_{g \in \mathcal{N}(v)} \mu_{g\to v}^{(t)}(x_v). \tag{4}$$

Our ultimate goal is to show that the estimated marginals are equal to the true marginals for all variables after a number iterations. However, we will first consider the rooted version of the factor graph and show that the previous statement holds for the root node a. More concretely, if we denote variable nodes with v and factor nodes with f, we will show using induction that for all subtrees $\mathcal{T}_{[fv]}$ of depth τ, it holds that, for all $t \geq \tau$,

$$\mu_{f\to v}^{(t)}(x_v) \propto P_{[fv]}(x_v). \tag{5}$$

Solution

(i) First, note that in this case the messages received by f from any neighboring variable node w are by definition $\mu^{(t)}_{w\to f}(x_w) = 1$, for all $t \geq 1$. Therefore, we have, for all $t \geq 2$,

$$\begin{aligned}
\mu^{(t)}_{f\to v}(x_v) &= \sum_{\boldsymbol{x}_{f\setminus\{v\}}} f(\boldsymbol{x}_f) \prod_{w\in\mathcal{N}(f)\setminus\{v\}} \mu^{(t-1)}_{w\to f}(x_w) && \text{by (3)} \\
&= \sum_{\boldsymbol{x}_{f\setminus\{v\}}} f(\boldsymbol{x}_f) && \text{using } \mu^{(t-1)}_{w\to f}(x_w) = 1 \\
&\propto P_{[fv]}(x_v). && \text{by (1)}
\end{aligned}$$

The last equality up to normalization follows from the fact that $\mathcal{T}_{[fv]}$ contains only one factor, namely f.

(ii) There are two core observations used in this step.

- For any neighboring factor node g of v (except for f), the subtree $\mathcal{T}_{[gv]}$ has the same depth as $\mathcal{T}_{[vf]}$, which in this case is τ. Therefore, the induction hypothesis (5) holds for these subtrees.
- Since $\mathcal{T}_{[vf]}$ is a tree, the factors can be partitioned into the factors of each subtree $\mathcal{T}_{[gv]}$. This means that there are no common factors between these subtrees and the union of the factors of all subtrees are exactly the factors of $\mathcal{T}_{[vf]}$. An analogous statement can be made for partitioning the variables of $\mathcal{T}_{[vf]}$ except for v.

Using the above, we get, for all $t \geq \tau + 1$,

$$\begin{aligned}
P_{[vf]}(x_v) &\propto \sum_{\boldsymbol{x}_{[vf]\setminus\{v\}}} \prod_{g\in\mathcal{F}_{[vf]}} g(\boldsymbol{x}_g) && \text{by (1)} \\
&= \sum_{\boldsymbol{x}_{[vf]\setminus\{v\}}} \prod_{g\in\mathcal{N}(v)\setminus\{f\}} \prod_{h\in\mathcal{F}_{[gv]}} h(\boldsymbol{x}_h) && \text{partition factors} \\
&= \prod_{g\in\mathcal{N}(v)\setminus\{f\}} \sum_{\boldsymbol{x}_{[gv]\setminus\{v\}}} \prod_{h\in\mathcal{F}_{[gv]}} h(\boldsymbol{x}_h) && \text{partition variables} \\
&\propto \prod_{g\in\mathcal{N}(v)\setminus\{f\}} P_{[gv]}(x_v) && \text{by (1)} \\
&\propto \prod_{g\in\mathcal{N}(v)\setminus\{f\}} \mu^{(t-1)}_{g\to v}(x_v) && \text{by induction hypothesis (depth of } \mathcal{T}_{[gv]} \text{ is } \tau) \\
&= \mu^{(t)}_{v\to f}(x_v). && \text{by (2)}
\end{aligned}$$

(iii) The arguments for this part are very similar to the ones in the previous part.

- Since $\mathcal{T}_{[fv]}$ has depth $\tau + 2$, all subtrees $\mathcal{T}_{[wf]}$ where w is a neighbor of f except for v will have a depth of τ, which allows us to use on them the result of the previous part.
- In this case, we can partition all factors except for f and all variables except for v into factors and variables of the corresponding subtrees $\mathcal{T}_{[wf]}$.

Using the above, we get, for all $t \geq \tau + 2$,

$$
\begin{aligned}
P_{[fv]}(x_v) &\propto \sum_{\boldsymbol{x}_{[fv]\setminus\{v\}}} \prod_{g \in \mathcal{F}_{[fv]}} g(\boldsymbol{x}_g) && \text{by (1)} \\
&= \sum_{\boldsymbol{x}_{[fv]\setminus\{v\}}} f(\boldsymbol{x}_f) \prod_{w \in \mathcal{N}(f)\setminus\{v\}} \prod_{h \in \mathcal{F}_{[wf]}} h(\boldsymbol{x}_h) && \text{partition factors} \\
&= \sum_{\boldsymbol{x}_{[fv]\setminus\{v\}}} f(\boldsymbol{x}_f) \prod_{w \in \mathcal{N}(f)\setminus\{v\}} \sum_{\boldsymbol{x}_{[wf]\setminus\{w\}}} \prod_{h \in \mathcal{F}_{[wf]}} h(\boldsymbol{x}_h) && \text{partition variables} \\
&= \sum_{\boldsymbol{x}_{[fv]\setminus\{v\}}} f(\boldsymbol{x}_f) \prod_{w \in \mathcal{N}(f)\setminus\{v\}} P_{[wf]}(x_v) && \text{by (1)} \\
&\propto \sum_{\boldsymbol{x}_{[fv]\setminus\{v\}}} f(\boldsymbol{x}_f) \prod_{w \in \mathcal{N}(f)\setminus\{v\}} \mu^{(t-1)}_{w\to f}(x_w) && \text{by part (ii) (depth of } \mathcal{T}_{[wf]} \text{ is } \tau) \\
&= \mu^{(t)}_{f\to v}(x_v). && \text{by (3)}
\end{aligned}
$$

This step combined with step (i) proves our inductive statement (5).

(iv) This is a direct consequence of the inductive statement proved in the previous steps, combined with the fact that a is the root of the factor graph. We again use the property that subtrees partition variables and factors. The following holds for all $t \geq d$:

$$
\begin{aligned}
\hat{P}^{(t)}(x_a) &= \prod_{g \in \mathcal{N}(a)} \mu^{(t)}_{g\to a}(x_a). && \text{by (4)} \\
&\propto \prod_{g \in \mathcal{N}(a)} P_{[ga]}(x_a) && \text{by (5)} \\
&= \prod_{g \in \mathcal{N}(a)} \sum_{\boldsymbol{x}_{[ga]\setminus\{a\}}} \prod_{h \in \mathcal{F}_{[ga]}} h(\boldsymbol{x}_h) && \text{by (1)} \\
&= \sum_{\boldsymbol{x}} \prod_{g \in \mathcal{N}(a)} \prod_{h \in \mathcal{F}_{[ga]}} h(\boldsymbol{x}_h) && \text{by variable partition} \\
&= \sum_{\boldsymbol{x}} \prod_{f \in \mathcal{F}} f(\boldsymbol{x}_f) && \text{by factor partition} \\
&= P(x_a). && \text{by definition}
\end{aligned}
$$

(v) For any variable node v, we can see the factor graph as a rooted tree with v as its root and follow the exact procedure of steps (i)–(iv) to prove that $\hat{P}^{(t)}(x_v) \propto P(x_v)$ holds for all $t \geq$ the depth of the corresponding tree. But, since the graph has diameter $\mathcal{D}$, it follows that every such tree will have depth at most $\mathcal{D}$, therefore the above statement holds for all $t \geq \mathcal{D}$.