Examples and Proofs of Inference in Junction Trees

Peter Lucas
LIACS, Leiden University

February 4, 2016

1 Representation and notation

Let $P(V)$ be a joint probability distribution, where V stands for a set of random variables. Now assume that $V = S \cup U$, where S and U are disjoint sets of random variables. Then we use the notation

$$\sum_{V \setminus S} P(V)$$

to represent the (marginalised) joint probability distribution obtained by summing out the variables V with the exception of S, i.e.

$$\sum_{V \setminus S} P(V) = P(S)$$

Sometimes we use *potentials*, denoted by φ, rather than probability distributions. Potentials are functions that look very similar to probability distributions. However, there are two differences with probability distributions:

(1) They need not sum to 1;

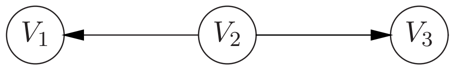
(2) There is no notation for conditioning as in probability distributions (i.e. $P(\cdot \mid \cdot)$).

For example if $\varphi(V_1, V_2)$ is defined in terms of a probability distribution, it may represent $P(V_1 \mid V_2)$, but also $P(V_1, V_2)$ or $P(V_2 \mid V_1)$, and even $P(V_1)$ or $P(V_2)$. Simply look at how it is defined to find out which of these is the case.

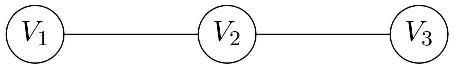
2 Motivation

We illustrate the basic ideas of probabilistic inference in junction trees by means of a simple example. Consider the Bayesian network shown in Figure 1(a). Assume we have the following probability distribution associated with the Bayesian network (following the structure of the graph, i.e. condition a variable on its parents):

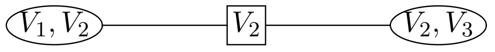
$P(v_1 \mid v_2) = 0.2$	$P(v_1 \mid \neg v_2) = 0.6$
$P(\neg v_1 \mid v_2) = 0.8$	$P(\neg v_1 \mid \neg v_2) = 0.4$
$P(v_3 \mid v_2) = 0.5$	$P(v_3 \mid \neg v_2) = 0.7$
$P(\neg v_3 \mid v_2) = 0.5$	$P(\neg v_3 \mid \neg v_2) = 0.3$
$P(v_2) = 0.9$	$P(\neg v_2) = 0.1$



(a) Bayesian network



(b) triangulated graph



(c) junction tree

Figure 1: Example Bayesian network, triangulated graph and junction tree.

We know that a Bayesian network defines a joint probability distribution on its associated variables, i.e.

$$P(X_V) = \prod_{v \in V} P(X_v \mid X_{\pi(v)})$$

with $\pi(v)$ the set of parents of vertex $v \in V$, and V the set of vertices of the associated acyclic directed graph $G = (V, A)$. So in this case we have:

$$P(V_1, V_2, V_3) = P(V_1 \mid V_2)P(V_2)P(V_3 \mid V_2)$$

For example:

$$P(v_1, \neg v_2, v_3) = P(v_1 \mid \neg v_2)P(\neg v_2)P(v_3 \mid \neg v_2) = 0.6 \cdot 0.1 \cdot 0.3 = 0.018$$

The algorithm to transform the Bayesian network into a junction tree transforms the Bayesian network first into a triangulated graph (See slides of the lecture on probabilistic inference). This is an undirected graph, where children on a converging connection are connected by a line (called *moralisation*), and chords are added to cut a cycle of more than three vertices short (called *triangulation*. In this example, there is no need for moralisation (no converging connection) and triangulation (not cycles), and thus the resulting triangulated graph is just the original directed graph with the direction of the edges removed. Check that Figure 1(a) and 1(b) represent exactly the same independence statements (slides lectures 4–5). We finally transform the triangulated graph into a junction tree by first constructing *cliques* C_i: $C_1 = \{V_1, V_2\}$ and $C_2 = \{V_2, V_3\}$ and their intersections, called *separators* S, here $S = \{V_2\}$.

Finally we have to define probability distributions on the cliques and separators using potentials. Here we have different options.

(Choice 1) One choice is for example to define:

$\varphi_{V_1,V_2} = P(V_1 \mid V_2)$
$\varphi_{V_2} = 1$
$\varphi_{V_2,V_3} = P(V_3 \mid V_2)P(V_2)$

Note that we have given the probabilities given above that

$$\sum_{V_1,V_2} \varphi_{V_1,V_2}(V_1, V_2) = \sum_{V_1,V_2} P(V_1 \mid V_2) = 0.2 + 0.6 + 0.8 + 0.4 = 2$$

and so, this potential is not normalised. Actually, it would not make sense to normalise it in this case, as we are dealing with *two* probability distributions represented in the potential ($P(V_1 \mid v_2)$ and $P(V_1 \mid \neg v_2)$). However, when it had represented a joint probability distributions, as for example φ_{V_2,V_3} does, and it is not normalised, it is possible to normalise it. For φ_{V_2,V_3} this is not needed, as we already have that $\sum_{V_2,V_3} \varphi_{V_2,V_3}(V_2, V_3) = 1$.

Clearly, given the potentials defined above, we have that the joint probability distribution $P(V_1, V_2, V_3)$ can be obtained as follows:

$$P(V_1, V_2, V_3) = \frac{\varphi_{V_1,V_2}\varphi_{V_2,V_3}}{\varphi_{V_2}} = P(V_1 \mid V_2)P(V_2)P(V_3 \mid V_2)$$

However, we might also have defined the potentials as follows (Choice 2):

$\varphi_{V_1,V_2} = P(V_1 \mid V_2)P(V_2)$
$\varphi_{V_2} = 1$
$\varphi_{V_2,V_3} = P(V_3 \mid V_2)$

and even as follows (Choice 3):

$\varphi_{V_1,V_2} = P(V_1 \mid V_2)P(V_2)$
$\varphi_{V_2} = P(V_2)$
$\varphi_{V_2,V_3} = P(V_3 \mid V_2)P(V_2)$

Note that in all cases $P(V_1, V_2, V_3)$ will be the same.

Let us now assume that we adopt the first definition of the potentials. In that case the clique C_1 only knows the family of probability distributions $P(V_1 \mid V_2)$ (one probability distribution $P(V_1 \mid v_2)$ and one probability distribution $P(V_1 \mid \neg v_2)$). However, in order to compute $P(V_1)$ or $P(V_2)$ we also need $P(V_2)$, as:

$$P(V_1) = \sum_{V_2} P(V_1 \mid V_2)P(V_2)$$

and

$$P(V_2) = \sum_{V_1} P(V_1 \mid V_2)P(V_2)$$

This information is not available in C_1, but it is in clique C_2. This explains the idea of using *message passing* to obtain this information from other cliques. For this definition of potentials, C_1 needs information from C_2; for the second example definition of potentials it appears that C_2 needs information from C_1. In general, it is not clear beforehand whether a clique will or will not need information from its neighbours. Hence, we need a general purpose algorithm.

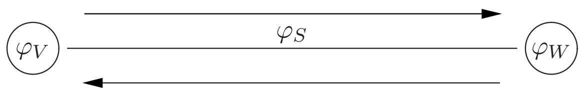


Figure 2: Schematic structure of message passing scheme.

3 Inference in junction trees

The general scheme for message passing in junction trees is shown in Figure 2. The example above can be easily translated into this general scheme, as we have:

$\varphi_V = \varphi_{V_1,V_2}$
$\varphi_S = \varphi_{V_2}$
$\varphi_W = \varphi_{V_2,V_3}$

The algorithm for message passing consists of two phases:

1. Updating from V to W (*forward pass*):

   $$\varphi_S^* = \textstyle\sum_{V\setminus S} \varphi_V \qquad \varphi_W^* = \frac{\varphi_S^*}{\varphi_S}\varphi_W$$

   This sets the separator to the marginal in φ_V.

2. Then, from W to V (*backward pass*):

   $$\varphi_S^{**} = \textstyle\sum_{W\setminus S} \varphi_W^* \qquad \varphi_V^* = \frac{\varphi_S^{**}}{\varphi_S^*}\varphi_V$$

Let us first see how this works for the examples above. Let us consider Choice 1 for the potentials (see above). We get:

$$\begin{aligned}
\varphi_S^* &= \sum_{V\setminus S} \varphi_V \\
&= \sum_{V_1} \varphi_{V_1,V_2} \\
&= \sum_{V_1} P(V_1 \mid V_2) \\
&= 1 \\
&= \varphi_{V_2}^*
\end{aligned}$$

Next, we update $\varphi_W = \varphi_{V2,V_3}$:

$$\varphi_W^* = \frac{\varphi_S^*}{\varphi_S}\varphi_W = \frac{\varphi_{V_2}^*}{\varphi_{V_2}}\varphi_{V_2,V_3} = 1/1\varphi_{V_2,V_3} = \varphi_{V_2,V_3}^*$$

So, here nothing has changed as $\varphi_{V_2,V_3} = \varphi_{V_2,V_3}^*$.

Subsequently, a message is passed back from $W = C_2$ to $V = C_1$:

$$\begin{aligned}
\varphi_S^{**} &= \sum_{W\setminus S} \varphi_W^* \\
&= \sum_{V_3} \varphi_{V_2,V_3}^* \\
&= \sum_{V_3} P(V_3 \mid V_2)P(V_2) \\
&= P(V_2)
\end{aligned}$$

Finally, we compute:

$$\varphi_V^* = \frac{\varphi_S^{**}}{\varphi_S^*}\varphi_V = \frac{\varphi_{V_2}^{**}}{\varphi_{V_2}^*}\varphi_{V_1,V_2} = \frac{P(V_2)}{1}P(V_1 \mid V_2) = P(V_1 \mid V_2)P(V_2)$$

So, now clique C_1 knows $P(V_2)$ and we can compute $P(V_1)$ and $P(V_2)$.

Investigate what happens when we had used Choice 2 or Choice 3 for the definition of the potentials.

Using the probabilities given above for the Bayesian network in Figure 1, we would get: $\varphi_{V_2}^*(v_2) = \varphi_{V_2}^*(\neg v_2) = 1$, $\varphi_{V_2,V_3}^*(v_2, v_3) = 0.5 \cdot 0.9 = 0.45$, $\varphi_{V_2,V_3}^*(\neg v_2, v_3) = 0.7 \cdot 0.1 = 0.07$, $\varphi_{V_2,V_3}^*(\neg v_2, \neg v_3) = 0.3 \cdot 0.1 = 0.03$, $\varphi_{V_2}^{**}(v_2) = 0.9$, $\varphi_{V_2}^{**}(\neg v_2) = 0.1$, $\varphi_{V_1,V_2}^*(v_1, v_2) = 0.9 \cdot 0.2 = 0.18$, $\varphi_{V_1,V_2}^*(\neg v_1, v_2) = 0.9 \cdot 0.8 = 0.72$, $\varphi_{V_1,V_2}^*(v_1, \neg v_2) = 0.1 \cdot 0.6 = 0.06$, $\varphi_{V_1,V_2}^*(\neg v_1, \neg v_2) = 0.1 \cdot 0.4 = 0.04$. Note that now:

$$\sum_{V_1,V_2} \varphi_{V_1,V_2}^*(V_1, V_2) = 0.18 + 0.72 + 0.06 + 0.04 = 1$$

4 Proof of correctness

Although the formulas for message passing given above appear to work correctly, we also would like to know in general whether the algorithm is correct. It does require some derivations, but in the end it is straightforward to obtain them.

4.1 Basics

Refer again to Figure 2. The following definitions will be used

$$P(V, W) = \frac{\varphi_V \varphi_W}{\varphi_S} \qquad (1)$$

This is the definition for the joint probability for junction trees. The general formula is as follows (see slides):

$$P(V) = \frac{\prod_{C \in \mathcal{C}} \varphi_C}{\prod_{S \in \mathcal{S}} \varphi_S}$$

where $\mathcal{C}$ is the set of cliques and $\mathcal{S}$ the set of separators of a junction tree, and $V = \bigcup_{C \in \mathcal{C}} C$.

Using marginalisation, we obtain the following formulas:

$$P(V) = \sum_{W \setminus S} P(V, W)$$

and

$$P(W) = \sum_{V \setminus S} P(V, W)$$

and finally,

$$P(S) = \sum_{(V \cup W) \setminus S} P(V, W) = \sum_{V \setminus S} P(V) = \sum_{W \setminus S} P(W)$$

Also recall the message passing formulas:

$$\varphi_S^* = \textstyle\sum_{V \setminus S} \varphi_V \qquad\qquad \varphi_W^* = \frac{\varphi_S^*}{\varphi_S}\varphi_W$$

and

$$\varphi_S^{**} = \textstyle\sum_{W \setminus S} \varphi_W^* \qquad\qquad \varphi_V^* = \frac{\varphi_S^{**}}{\varphi_S^*}\varphi_V$$

4.2 Forward pass

The derivation of the formula for the forward pass stage of message passing is easy, using Equation (1):

$$\begin{aligned}
P(W) &= \sum_{V\setminus S} P(V,W) \\
&= \sum_{V\setminus S} \frac{\varphi_V \varphi_W}{\varphi_S} \\
&= \frac{\varphi_W}{\varphi_S} \sum_{V\setminus S} \varphi_V \\
&= \frac{\varphi_W}{\varphi_S} \varphi_S^* \\
&= \frac{\varphi_S^*}{\varphi_S} \varphi_W = \varphi_W^* \qquad (2)
\end{aligned}$$

Thus, after the forward pass the updated potential of W, i.e. φ_W^*, is equal to the joint probability distribution $P(W)$. The ratio $\frac{\varphi_S^*}{\varphi_S}$ is often called the *update ratio*.

4.3 Backward pass

We start by using the basic result from subsection 4.1 that

$$P(S) = \sum_{W\setminus S} P(W) = \sum_{W\setminus S} \varphi_W^*$$

according to Equation (2). The algorithm uses the notation

$$\varphi_S^{**} = \sum_{W\setminus S} \varphi_W^* \qquad (3)$$

for this result. Using the notation used in the algorithm:

$$\varphi_V^* = \frac{\varphi_S^{**}}{\varphi_S^*} \varphi_V \qquad (4)$$

We wish to prove that

$$\varphi_V^* = P(V)$$

as this means that, together with the result above, that the algorithm is correct.

We start as follows:

$$\begin{aligned}
P(V,W) &= \frac{\varphi_V \varphi_W}{\varphi_S} \cdot \frac{\varphi_S^*}{\varphi_S^*} \\
&= \frac{\varphi_V \varphi_W^*}{\varphi_S^*} \qquad (5)
\end{aligned}$$

using Equation (2).

From Equation (4), we derive by switching terms, that

$$\varphi_V = \frac{\varphi_V^* \varphi_S^*}{\varphi_S^{**}}$$

Substituting this result into Equation (5) gives:

$$\begin{aligned} P(V,W) &= \frac{P(W)\varphi_V^*}{\varphi_S^{**}} \\ &= \frac{P(W)\varphi_V^*}{P(S)} \end{aligned}$$

using Equation (3).

Now we proceed as follows:

$$\begin{aligned} P(V) &= \sum_{W\backslash S} P(V,W) \\ &= \sum_{W\backslash S} \frac{P(W)\varphi_V^*}{P(S)} \\ &= \frac{\varphi_V^*}{P(S)} \sum_{W\backslash S} P(W) \\ &= \frac{\varphi_V^*}{P(S)} P(S) \\ &= \varphi_V^* \end{aligned}$$

So after termination of the algorithm, we have that $\varphi_V^* = P(V)$ and we already had that $\varphi_W^* = P(W)$, in other words, the algorithm is correct.

4.4 Local consistency

If we now pass back information again from V to W then of course nothing will change, as the situation is already stable. Passing back again means computing:

$$\varphi_S^{***} = \sum_{V\backslash S} \varphi_V^*$$

This must be the same as $\varphi_S^{**} = \sum_{W\backslash S} \varphi_W^*$, and that is what we need to prove.

So, to be proved is whether $\varphi_S^{***} = \varphi_S^{**}$ holds. The proof goes as follows:

$$\begin{aligned} \varphi_S^{***} &= \sum_{V\backslash S} \varphi_V^* \\ &= \sum_{V\backslash S} \frac{\varphi_S^{**}}{\varphi_S^*}\varphi_V \\ &= \frac{\varphi_S^{**}}{\varphi_S^*} \sum_{V\backslash S} \varphi_V \\ &= \frac{\varphi_S^{**}}{\varphi_S^*}\varphi_S^* \\ &= \varphi_S^{**} \end{aligned}$$

So, that was pretty easy. To conclude, after finishing with message passing, probabilistic information about the separator S obtained from V is exactly the same as the probabilistic information obtained from W, called *local consistency*.