INTRODUCTION TO INFERENCE FOR BAYESIAN NETWORKS

ROBERT COWELL
City University, London.
The School of Mathematics, Actuarial Science and Statistics,
City University, Northampton Square, London EC1E 0HT

1. Introduction

The field of Bayesian networks, and graphical models in general, has grown enormously over the last few years, with theoretical and computational developments in many areas. As a consequence there is now a fairly large set of theoretical concepts and results for newcomers to the field to learn. This tutorial aims to give an overview of some of these topics, which hopefully will provide such newcomers a conceptual framework for following the more detailed and advanced work. It begins with revision of some of the basic axioms of probability theory.

2. Basic axioms of probability

Probability theory, also known as inductive logic, is a system of reasoning under uncertainty, that is under the absence of certainty. Within the Bayesian framework, probability is interpreted as a numerical measure of the degree of consistent belief in a proposition, consistency being with the data at hand.

Early expert systems used deductive, or Boolean, logic, encapsulated by sets of production rules. Attempts were made to cope with uncertainty using probability theory, but the calculations became prohibitive, and the use of probability theory for inference in expert systems was abandoned. It is with the recent development of efficient computational algorithms that probability theory has had a revival within the AI community.

Let us begin with some basic axioms of probability theory. The probability of an event A, denoted by $P(A)$, is a number in the interval [0,1], which obeys the following axioms:

1 $P(A) = 1$ if and only if A is certain.

2 If A and B are mutually exclusive, then $P(A \text{ or } B) = P(A) + P(B)$.

We will be dealing exclusively with discrete random variables and their probability distributions. Capital letters will denote a variable, or perhaps a set of variables, lower case letter will denote values of variables. Thus suppose A is a random variable having a finite number of *mutually exclusive states* $(a_1, \ldots, a_n)$. Then $P(A)$ will be represented by a vector of non-negative real numbers $P(A) = (x_1, \ldots, x_n)$ where $P(A = a_i) = x_i$ is a scalar, and $\sum_i x_i = 1$.

A basic concept is that of conditional probability, a statement of which takes the form: *Given the event $B = b$ the probability of the event $A = a$ is x, written $P(A = a \mid B = b) = x$.* It is important to understand that this is not saying: "If $B = b$ is true then the probability of $A = a$ is x". Instead it says: "If $B = b$ is true, and any other information to hand is irrelevant to A, then $P(A = a) = x$". (To see this, consider what the probabilities would be if the state of A was part of the extra information).

Conditional probabilities are important for building Bayesian networks, as we shall see. But Bayesian networks are also built to facilitate the calculation of conditional probabilities, namely the conditional probabilities for variables of interest given the data (also called evidence) at hand.

The fundamental rule for probability calculus is the product rule[1]

$$P(A \text{ and } B) = P(A \mid B)P(B). \tag{1}$$

This equation tells us how to combine conditional probabilities for individual variables to define joint probabilities for sets of variables.

3. Bayes' theorem

The simplest form of Bayes' theorem relates the joint probability $P(A \text{ and } B)$ – written as $P(A, B)$ – of two events or hypotheses A and B in terms of marginal and conditional probabilities:

$$P(A, B) = P(A \mid B)P(B) = P(B \mid A)P(A). \tag{2}$$

By rearrangement we easily obtain

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}, \tag{3}$$

which is Bayes' theorem.

This can be interpreted as follows. We are interested in A, and we begin with a *prior* probability $P(A)$ for our belief about A, and then we observe

[1]Or more generally $P(A \text{ and } B \mid C) \equiv P(A \mid B, C)P(B \mid C)$.

B. Then Bayes' theorem, (3), tells us that our revised belief for A, the *posterior* probability $P(A \mid B)$ is obtained by multiplying the prior $P(A)$ by the ratio $P(B \mid A)/P(B)$. The quantity $P(B \mid A)$, as a function of varying A for fixed B, is called the *likelihood* of A. We can express this relationship in the form:

$$\begin{array}{rcl} \textbf{posterior} & \propto & \textbf{prior} \times \textbf{likelihood} \\ P(A \mid B) & \propto & P(A)P(B \mid A). \end{array}$$

Figure 1 illustrates this prior-to-posterior inference process. Each diagram

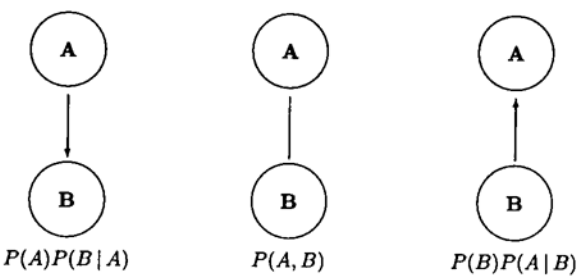


Figure 1. Bayesian inference as reversing the arrows

represents in different ways the joint distribution $P(A, B)$, the first represents the prior beliefs while the third represents the posterior beliefs. Often, we will think of A as a possible "cause" of the "effect" B, the downward arrow represents such a causal interpretation. The "inferential" upwards arrow then represents an "argument against the causal flow", from the observed effect to the inferred cause. (We will not go into a definition of "causality" here.)

Bayesian networks are generally more complicated than the ones in Figure 1, but the general principles are the same in the following sense. A Bayesian network provides a model representation for the joint distribution of a set of variables in terms of conditional and prior probabilities, in which the orientations of the arrows represent influence, usually though not always of a causal nature, such that these conditional probabilities for these particular orientations are relatively straightforward to specify (from data or eliciting from an expert). When data are observed, then typically an inference procedure is required. This involves calculating marginal probabilities conditional on the observed data using Bayes' theorem, which is diagrammatically equivalent to reversing one or more of the Bayesian network arrows. The algorithms which have been developed in recent years

allows these calculations to be performed in an efficient and straightforward manner.

4. Simple inference problems

Let us now consider some simple examples of inference. The first is simply Bayes' theorem with evidence included on a simple two node network; the remaining examples treat a simple three node problem.

4.1. PROBLEM I

Suppose we have the simple model $X \rightarrow Y$, and are given: $P(X)$, $P(Y \mid X)$ and $Y = y$. The problem is to calculate $P(X \mid Y = y)$.

Now from $P(X)$, $P(Y \mid X)$ we can calculate the marginal distribution $P(Y)$ and hence $P(Y = y)$. Applying Bayes' theorem we obtain

$$P(X \mid Y = y) = \frac{P(Y = y \mid X)P(X)}{P(Y = y)}. \tag{4}$$

4.2. PROBLEM II

Suppose now we have a more complicated model in which X is a parent of both Y and Z: $Z \leftarrow X \rightarrow Y$ with specified probabilities $P(X)$, $P(Y \mid X)$ and $P(Z \mid X)$, and we observe $Y = y$. The problem is to calculate $P(Z \mid Y = y)$. Note that the joint distribution is given by $P(X, Y, Z) = P(Y \mid X)P(Z \mid X)P(X)$. A 'brute force' method is to calculate:

1. The joint distribution $P(X, Y, Z)$.
2. The marginal distribution $P(Y)$ and thence $P(Y = y)$.
3. The marginal distribution $P(Z, Y)$ and thence $P(Z, Y = y)$.
4. $P(Z \mid Y = y) = P(Z, Y = y)/P(Y = y)$.

An alternative method is to exploit the given factorization:

1. Calculate $P(X \mid Y = y) = P(Y = y \mid X)P(X)/P(Y = y)$ using Bayes' theorem, where $P(Y = y) = \sum_X P(Y = y \mid X)P(X)$.
2. Find $P(Z \mid Y = y) = \sum_X P(Z \mid X)P(X \mid Y = y)$.

Note that the first step essentially reverses the arrow between X and Y. Although the two methods give the same answer, the second is generally more efficient. For example, suppose that all three variables have 10 states. Then the first method in explicitly calculating $P(X, Y, Z)$ requires a table of 1000 states. In contrast the largest table required for the second method has size 100. This gain in computational efficiency by exploiting the given factorizations is the basis of the arc-reversal method for solving influence

diagrams, and of the junction-tree propagation algorithms. The following example shows the same calculation using propagation on a junction tree.

4.3. PROBLEM III

Suppose now that we are given the *undirected* structure $ZX - X - XY$, and probabilities $P(Z,X)$, $P(X)$ and $P(Y,X)$. Again the problem is to calculate $P(Z \mid Y = y)$. Note that:

$$\begin{aligned} P(Z,X) &= P(Z \mid X)P(X) \\ P(Y,X) &= P(Y \mid X)P(X) \\ P(X,Y,Z) &= P(Z,X)P(Y,X)/P(X). \end{aligned}$$

The calculational steps now proceeds using a 'message' in step 1 which is 'sent' in step 2:

1. Calculate $\bar{P}(X) \equiv \sum_Y P(X, Y = y)$.
2. Find $\bar{P}(Z,X) \equiv P(Z,X)\bar{P}(X)/P(X)$.
3. Find $P(Z, Y = y) = \sum_X \bar{P}(Z,X)$.
4. Find $P(Z \mid Y = y) = P(Z, Y = y)/\sum_Z P(Z, Y = y)$

5. Conditional independence

In the last example we had that

$$P(X,Y,Z) = P(Y \mid X)P(Z \mid X)P(X),$$

from which we get

$$\begin{aligned} P(Y \mid Z,X) &= \frac{P(X,Y,Z)}{P(Z,X)} \\ &= \frac{P(Y \mid X)P(Z \mid X)P(X)}{P(Z,X)} \\ &= P(Y \mid X) \end{aligned}$$

and likewise for $P(Z \mid Y,X) = P(Z \mid X)$. Hence given $X = x$ say, we obtain $P(Y \mid Z, X = x) = P(Y \mid X = x)$ and $P(Z \mid Y, X = x) = P(Z \mid X = x)$. This is an example of conditional independence (Dawid(1979)). We associated the graph $Z \leftarrow X \rightarrow Y$ with this distribution, though this is not unique. In fact the joint probability can be factorized according to three distinct directed graphs:

$$\begin{aligned} &Z \leftarrow X \rightarrow Y : P(X,Y,Z) = P(X)P(Y \mid X)P(Z \mid X). \\ &Z \rightarrow X \rightarrow Y : P(X,Y,Z) = P(Y \mid X)P(X \mid Z)P(Z). \\ &Z \leftarrow X \leftarrow Y : P(X,Y,Z) = P(X \mid Y)P(Z \mid X)P(Y). \end{aligned}$$

Each of these factorizations follows from the conditional independence properties which each graph expresses, viz $Z \perp\!\!\!\perp Y \mid X$, (which is to be read as "Z is conditionally independent of Y given X") and by using the general factorization property:

$$\begin{aligned} P(X_1, \ldots X_n) &= P(X_1 \mid X_2, \ldots, X_n) P(X_2, \ldots, X_n) \\ &= P(X_1 \mid X_2, \ldots, X_n) P(X_2 \mid X_3, \ldots, X_n) P(X_3, \ldots, X_n) \\ &= \vdots \\ &= P(X_1 \mid X_2, \ldots, X_n) \ldots P(X_{n-1} \mid X_n) P(X_n). \end{aligned}$$

Thus for the third example

$$P(X, Y, Z) = P(Z \mid X, Y) P(X \mid Y) P(Y) = P(Z \mid X) P(X \mid Y) P(Y).$$

Note that the graph $Z \rightarrow X \leftarrow Y$ does not obey the conditional independence property $Z \perp\!\!\!\perp Y \mid X$ and is thus excluded from the list; it factorizes as $P(X, Y, Z) = P(X \mid Y, Z) P(Z) P(Y)$.

This example shows several features of general Bayesian networks. Firstly, the use of the conditional independence properties can be used to simplify the general factorization formula for the joint probability. Secondly, that the result is a factorization that can be expressed by the use of directed acyclic graphs (DAGs).

6. General specification in DAGs

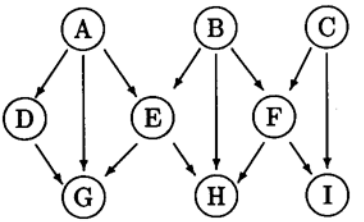
It is these features which work together nicely for the general specification of Bayesian networks. Thus a Bayesian network is a directed acyclic graph, whose structure defines a set of conditional independence properties. These properties can be found using graphical manipulations, eg *d-separation* (see eg Pearl(1988)). To each node is associated a conditional probability distribution, conditioning being on the parents of the node: $P(X \mid \mathrm{pa}(X))$. The joint density over the set of all variables U is then given by the product of such terms over all nodes:

$$P(U) = \prod_X P(X \mid \mathrm{pa}(X)).$$

This is called a *recursive factorization* according to the DAG; we also talk of the distribution being graphical over the DAG. This factorization is equivalent to the general factorization but takes into account the conditional independence properties of the DAG in simplifying individual terms in the product of the general factorization. Only if the DAG is complete will this formula and the general factorization coincide, (but even then only for one ordering of the random variables in the factorization).

6.1. EXAMPLE

Consider the graph of Figure 2.



$$
\begin{aligned}
&P(A,B,C,D,E,F,G,H,I)\\
&=P(A)P(B)P(C)\\
&\quad P(D\,|\,A)P(E\,|\,A,B)P(F\,|\,B,C)\\
&\quad P(G\,|\,A,D,E)P(H\,|\,B,E,F)P(I\,|\,C,F).
\end{aligned}
$$

Figure 2. Nine node example.

It is useful to note that marginalising over a childless node is equivalent to simply removing it and any edges to it from its parents. Thus for example, marginalising over the variable H in the above gives:

$$
\begin{aligned}
P(A,B,C,D,E,F,G,I) &= \sum_H P(A,B,C,D,E,F,G,H,I)\\
&= \sum_H P(A)P(B)P(C)P(D\,|\,A)P(E\,|\,A,B)P(F\,|\,B,C)\\
&\qquad P(G\,|\,A,D,E)P(H\,|\,B,E,F)P(I\,|\,C,F)\\
&= P(A)P(B)P(C)P(D\,|\,A)P(E\,|\,A,B)P(F\,|\,B,C)\\
&\qquad P(G\,|\,A,D,E)P(I\,|\,C,F)\sum_H P(H\,|\,B,E,F)\\
&= P(A)P(B)P(C)P(D\,|\,A)P(E\,|\,A,B)P(F\,|\,B,C)\\
&\qquad P(G\,|\,A,D,E)P(I\,|\,C,F),
\end{aligned}
$$

which can be represented by Figure 2 with H and its incident edges removed.

Directed acyclic graphs can always have their nodes linearly ordered so that for each node X all of its parents $\mathrm{pa}(X)$ precedes it in the ordering. Such and ordering is called a *topological ordering* of the nodes. Thus for example (A,B,C,D,E,F,G,H,I) and (B,A,E,D,G,C,F,I,H) are two of the many topological orderings of the nodes of Figure 2.

A simple algorithm to find a topological ordering is as follows: Start with the graph and an empty list. Then successively delete from the graph any node which does not have any parents, and add it to the end of the list. Note that if the graph is not acyclic, then at some stage a graph will be obtained in which no node has no parent nodes, hence this algorithm can be used as an efficient way of checking that the graph is acyclic.

Another equivalent way is to start with the graph and an empty list, and successively delete nodes which have no children and add them to the beginning of the list (cf. marginalisation of childless nodes.)

6.2. DIRECTED MARKOV PROPERTY

An important property is the *directed Markov property.* This is a conditional independence property which states that a variable is conditionally independent of its non-descendents given it parents:

$$X \perp\!\!\!\perp \mathrm{nd}(X) \,|\, \mathrm{pa}(X).$$

Now recall that the conditional probability $P(X \,|\, \mathrm{pa}(X))$ did not necessarily mean that if $\mathrm{pa}(X) = \pi^*$ say, then $P(X = x) = P(x \,|\, \pi^*)$, but included the caveat that any other information is irrelevant to X for this to hold. For the DAGs this 'other information' means, from the directed Markov property, knowledge about the node itself or any of its descendents. For if all of the parents of X are observed, but additionally observed are one or more descendents D_X of X, then because X influences D_X, knowing D_X and $\mathrm{pa}(X)$ is more informative than simply knowing about $\mathrm{pa}(X)$ alone. However having information about a non-descendent does not tell us anything more about X, because either it cannot influence or be influenced by X either directly or indirectly, or if it can influence X indirectly, then only through influencing the parents which are all known anyway.

For example, consider again Figure 2. Using the previous second topological ordering we may write the general factorization as:

$$\begin{aligned} P(A,B,C,D,E,F,G,I,H) = & P(B) \\ & * P(A \,|\, B) \\ & * P(E \,|\, B, A) \\ & * P(D \,|\, B, A, E) \\ & * P(G \,|\, B, A, E, D) \\ & * P(C \,|\, B, A, E, D, G) \\ & * P(F \,|\, B, A, E, D, G, C) \\ & * P(I \,|\, B, A, E, D, G, C, F) \\ & * P(H \,|\, B, A, E, D, G, C, F, I) \end{aligned} \tag{5}$$

but now we can use $A \perp\!\!\!\perp B$ from the directed Markov property to simplify $P(A \,|\, B) \rightarrow P(A)$, and similarly for the other factors in (5) etc, to obtain the factorization in Figure 2. We can write the general pseudo-algorithm of what we have just done for this example as

Topological ordering +
General factorization +
Directed Markov property
$\Longrightarrow$ Recursive factorization.

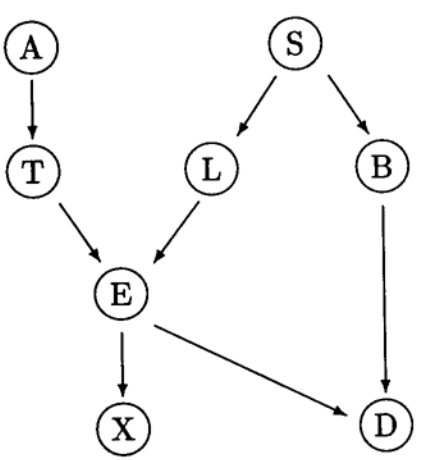
7. Making the inference engine

We shall now move on to building the so called "inference engine" to introduce new concepts and to show how they relate to the conditional independence/recursive factorization ideas that have already been touched upon. Detailed justification of the results will be omitted, the aim here is to give an overview, using the use the fictional ASIA example of Lauritzen and Spiegelhalter.

7.1. ASIA: SPECIFICATION

Lauritzen and Spiegelhalter describe their fictional problem domain as follows:

> Shortness-of-breath (**Dyspnoea**) may be due to **Tuberculosis**, **Lung** cancer or **Bronchitis**, or none of them, or more than one of them. A recent visit to **Asia** increases the chances of **Tuberculosis**, while **Smoking** is known to be a risk factor for both **Lung** cancer and **Bronchitis**. The results of a single **X**-ray do not discriminate between **Lung** cancer and **Tuberculosis**, as neither does the presence or absence of **Dyspnoea**.



$$
\begin{aligned}
P(U) = & P(A)P(S) \\
& P(T \mid A)P(L \mid S) \\
& P(B \mid S)P(E \mid L,T) \\
& P(D \mid B,E)P(X \mid E)
\end{aligned}
$$

Figure 3. ASIA

The network for this fictional example is shown in Figure 3. Each variable is a binary with the states ("yes", "no"). The **E** node is a logical node taking value "yes" if either of its parents take a "yes" value, and "no" otherwise; its introduction facilitates modelling the relationship of **X**-ray to **L**ung cancer and **T**uberculosis.

Having specified the relevant variables, and defined their dependence with the graph, we must now assign (conditional) probabilities to the nodes. In real life examples such probabilities may be elicited either from some large database (if one is available) as frequency ratios, or subjectively from the expert from whom the structure has been elicited (eg using a fictitious gambling scenario or probability wheel), or a combination of both. However as this is a fictional example we can follow the third route and use made-up values. (Specific values will be omitted here.)

7.2. CONSTRUCTING THE INFERENCE ENGINE

With our specified graphical model we have a representation of the joint density in terms of a factorization:

$$P(U) = \prod_V P(V \mid \mathrm{pa}(V)) \tag{6}$$

$$= P(A) \ldots P(X \mid E). \tag{7}$$

Recall that our motivation is to use the model specified by the joint distribution to calculate marginal distributions conditional on some observation of one or more variables. In general the full distribution will be computationally difficult to use directly to calculate these marginals directly. We will now proceed to outline the various stages that are performed to find a representation of $P(U)$ which makes the calculations more tractable. (The process of constructing the inference engine from the model specification is sometimes called *compiling* the model.)

The manipulations required are almost all graphical. There are five stages in the graphical manipulations. Let us first list them, and then go back and define new terms which are introduced.

1. Add undirected edges to all co-parents which are not currently joined (a process called *marrying parents*).
2. Drop all directions in the graph obtained from Stage 1. The result is the so-called *moral graph.*
3. *Triangulate the moral graph*, that is, add sufficient additional undirected links between nodes such that there are no cycles (ie. closed paths) of length 4 or more distinct nodes without a short-cut.
4. Identify the *cliques* of this triangulated graph.
5. Join the cliques together to form the *junction tree.*

Now let us go through these steps, supplying some justification and defining the new terms just introduced as we go along. Consider first the joint density again. By a change of notation this can be written in the form

$$P(U) = \prod_V a(V, \mathrm{pa}(V)) \tag{8}$$

$$= a(A) \ldots a(X, E). \tag{9}$$

where $a(X, \mathrm{pa}(X)) \equiv P(V \mid \mathrm{pa}(V))$. That is, the conditional probability factors for V can be considered as a function of V and its parents. We call such functions *potentials*. Now after steps 1 and 2 we have an undirected graph, in which for each node both it and its set of parents in the original graph form a complete subgraph in the moral graph. (A complete graph is one in which every pair of nodes is joined together by an edge.) Hence, the original factorization of $P(U)$ on the DAG G goes over to an equivalent factorization on these complete subsets in the moral graph G^m. Technically we say that the distribution is graphical on the undirected graph G^m. Figure 4 illustrates the moralisation process for the Asia network. Now let us de-

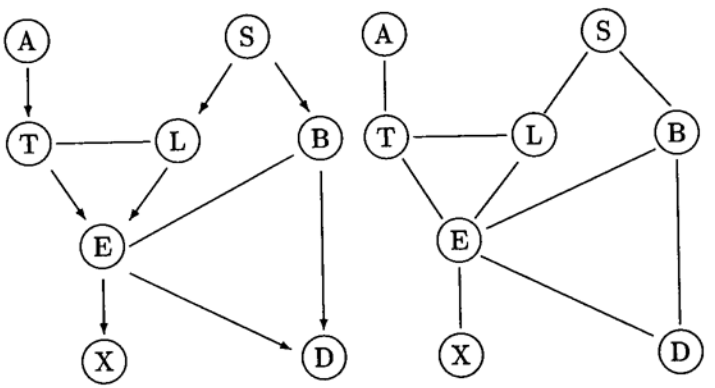


Figure 4. Moralising Asia: Two extra links are required, $A - S$ and $L - B$. Directionality is dropped after all moral edges have been added.

note the set of cliques of the moral graph by C^m. (A clique is a complete subgraph which is not itself a proper subgraph of a complete subgraph, so it is a maximal complete subgraph.) Then each of the complete subgraphs formed from $\{V\} \cup \mathrm{pa}(V)$ is contained within at least one clique. Hence we can form functions a_C such that

$$P(U) = \prod_{c \in C^m} a_C(V_C)$$

where $a_C(V_C)$ is a function of the variables in the clique C. Such a factorization can be constructed as follows: Initially define each factor as unity, i.e.,

$a_C(V_C) = 1$ for all cliques in C^m. Then for each factor $P(V \mid \mathrm{pa}(V))$ find one and only one clique which contains the complete subgraph of $\{V\} \cup \mathrm{pa}(V)$ and multiply this conditional distribution into the function of that clique to obtain a new function. When this is done the result is a potential representation of the joint distributions in terms of functions on the cliques of the moral G^m.

Note that by adding the extra edges in the moralisation process it is not possible to read of all of the conditional independences of the original DAG, though they are still there "buried" in the numerical specification. Those which remain "visible" in the moral graph are used to exploit the efficient local computations which will be described later.

8. Aside: Markov properties on ancestral sets

In fact the moral graph is a powerful construction for elicudating conditional independence. First we require some more definitions. A node A is an ancestor of a node B if either (i) A is a parent of B or (ii) A is an ancestor of (at least) one of the parents of B. The ancestral set of a node is the node itself and the set of its ancestors. The ancestral set of a set of nodes Y is the union of the ancestral sets of the nodes in Y. A set S *separates* the sets A and B if every path between a node $a \in A$ and $b \in B$ passes through some node of S. With these definitions we have:

Lemma 1

Let P factorize recursively according to $\mathcal{G}$. Then

$$A \perp\!\!\!\perp B \mid S$$

whenever A and B are separated by S in $(\mathcal{G}_{\mathrm{An}(A \cup B \cup S)})^m$, the moral graph of the smallest ancestral set containing $A \cup B \cup S$.

Lemma 2

Let A, B and S be disjoint subsets of a directed, acyclic graph $\mathcal{G}$. Then S d-separates A from B if and only if S separates A from B in $(\mathcal{G}_{\mathrm{An}(A \cup B \cup S)})^m$.

What these lemmas tell us is that if we want to check conditional independences we can either look at d-separation properties or the smallest ancestral sets of the moral graphs – they are alternative ways of calculation.

To understand why ancestral set come into the picture, let us consider the following simple algorithm for finding them. Suppose that we have the graph G and that we wish to find the ancestral set of a set of nodes $Y \subseteq U$. Then successively delete nodes from G which have no children, provided they are not in the set Y. When it is not possible any longer delete any nodes, the subgraph left is the minimal ancestral set.

Now recall that deleting a childless node is equivalent to marginalising over that node. Hence the marginal distribution of the minimal ancestral set containing $A \perp\!\!\!\perp B \mid S$ factorizes according to the sub-factors of the original joint distribution. So these lemmas are saying that rather than go through the numerical exercise of actually calculating such marginals we can read it off from the graphical structure instead, and use that to test conditional independences. (Note also that the directed Markov property is also lurking behind the scenes here.) The "moral" is that when ancestral sets appear in theorems like this it is likely that such marginals are being considered.

9. Making the junction tree

The remaining three steps of the inference-engine construction algorithm seem more mysterious, but are required to ensure we can formulate a consistent and efficient message passing scheme. Consider first step 3 – adding edges to the moral graph G^m to form a triangulated graph G^t. Note that adding edges to the graph does not stop a clique of the moral graph formed from being a complete subgraph in G^t. Thus for each clique in C^m of the moral graph there is at least one clique in the triangulated graph which contains it. Hence we can form a potential representation of the joint probability in terms of products of functions of the cliques in the triangulated graph:

$$P(U) = \prod_{c \in C^t} a_C(X_C)$$

by analogy with the previous method outline for the moral graph. The point is that after moralisation and triangulation there exists for each a node-parent set at least one clique which contains it, and thus a potential representation can be formed on the cliques of the triangulated graph.

While the moralisation of a graph is unique, there are in general many alternative triangulations of a moral graph. In the extreme, we can always add edges to make the moral graph complete. There is then one large clique. The key to the success of the computational algorithms is to form triangulated graphs which have small cliques, in terms of their state space size.

Thus after finding the cliques of the triangulated graph – stage 4 – we are left with joining them up to form a junction tree. The important property of the junction tree is the *running intersection property* which means that if variable V is contained in two cliques, then it is contained in every clique along the path connecting those two cliques. The edge joining two cliques is called a separator. This joining up property can always be done, not necessarily uniquely for each triangulated graph. However the choice of

tree is immaterial except for computational efficiency considerations. The junction tree captures many, but not necessarily all, of the conditional independence properties of the distribution on the original DAG. It loses some of the conditional independences by the process of adding extra edges to the moral graph. However it does retain conditional independence between (not necessarily neighbouring) cliques given separators between them. It is because of this fact that local computation with message passing becomes possible. The running intersection property ensures *consistence* in the message passing between cliques, and the cliques become the basic unit of the local computation, ie., they define the granularity of the computational algorithms. If the cliques are of manageable size then local computation is possible. Figure 5 shows a triangulated version of Asia and a possible junction tree.

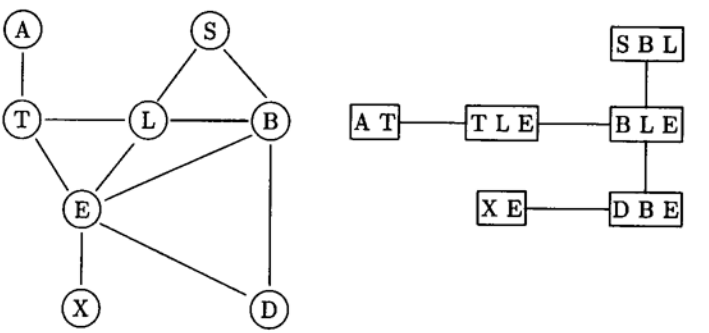


Figure 5. Junction tree for Asia

10. Inference on the junction tree

We will summarise some of the basic results of message passing on the junction tree. We have seen that we can form a potential representation of the joint probability using functions defined on the cliques:

$$P(U) = \prod_{C \in C^t} a_C(X_C).$$

This can be generalized to include functions on the separators (the intersections of neighbouring cliques) to form the following so called *generalized potential representation*:

$$P(U) = \frac{\prod_{C \in C^t} a_C(X_C)}{\prod_{S \in S^t} b_S(X_S)}.$$

(for instance by making the separator functions the identity). Now, by sending messages between neighbouring cliques consisting of functions of the separator variables only, which modify the intervening separator and the clique receiving the message, but in such a way that the overall ratio of products remains invariant, we can arrive at the following *marginal representation*:

$$p(U) = \frac{\prod_{C \in \mathcal{C}} p(C)}{\prod_{S \in \mathcal{S}} p(S)}. \tag{10}$$

Marginals for individual variables can be obtained from these clique (or separator) marginals by further marginalisation.

Suppose that we observe "evidence" $\mathcal{E} : X_A = x_A^*$. Define a new function P^* by

$$P^*(x) = \begin{cases} P(x) & \text{if } X_A = x_A^* \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

Then $P^*(U) = P(U, \mathcal{E}) = P(\mathcal{E})P(U \mid \mathcal{E})$. We can rewrite (11) as

$$P^*(U) = P(U) \prod_{v \in A} l(v), \tag{12}$$

where $l(v)$ is 1 if $x_v = x_v^*$, 0 otherwise. Thus $l(v)$ is the *likelihood function* based on the partial evidence $X_v = x_v^*$. Clearly this also factorizes on the junction tree, and by message passing we may obtain the following *clique-marginal representation*

$$p(V \mid \mathcal{E}) = \frac{\prod_{C \in \mathcal{C}} p(C \mid \mathcal{E})}{\prod_{S \in \mathcal{S}} p(S \mid \mathcal{E})}. \tag{13}$$

or by omitting the normalization stage,

$$p(V, \mathcal{E}) = \frac{\prod_{C \in \mathcal{C}} p(C, \mathcal{E})}{\prod_{S \in \mathcal{S}} p(S, \mathcal{E})}. \tag{14}$$

Again marginal distributions for individual variables, conditional upon the evidence, can be obtained by further marginalisation of individual clique tables, as can the probability (according to the model) of the evidence, $P(\mathcal{E})$.

11. Why the junction tree?

Given that the moral graph has nice properties, why is it necessary to go on to form the junction tree? This is best illustrated by an example, Figure 6:

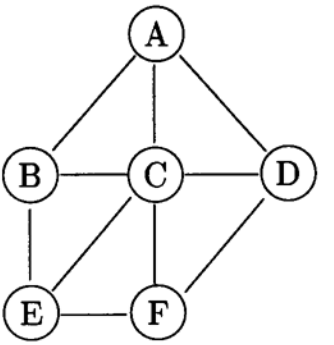


Figure 6. A non-triangulated graph

The cliques are (A, B, C), (A, C, D), (C, D, F), (C, E, F) and (B, C, E) with successive intersections (A, C), (C, D), (C, F), (C, E) and (B, C). Suppose we have clique marginals $P(A, B, C)$ *etc.*. We cannot express $P(A, B, C, D)$ in terms of $P(A, B, C)$ and $P(A, C, D)$ - the graphical structure does *not* imply $B \perp\!\!\!\perp D|(A, C)$. In general there is no closed for expression for the joint distribution of all six variables in terms of its cliques marginals.

12. Those extra edges again

Having explained why the cliques of the moral graph are generally not up to being used for local message passing, we will now close by indicating where the extra edges to form a triangulated graph come from.

Our basic message passing algorithm will be one in which marginals of the potentials in the cliques will form the messages on the junction tree. So let us begin with our moral graph with a potential representation in terms of functions on the cliques, and suppose we marginalise a variable Y say,which belongs to more than one clique of the graph, say two cliques, C_1 and C_2, with variables $Y \cup Z_1$ and $Y \cup Z_2$ respectively. They are cliques, but the combined set of variables do not form a single clique, hence there must be at least one pair of variables, one in each clique, which are not joined to each other, u_1 and u_2 say.

Now consider the effect of marginalisation of the variable Y. We will have

$$\sum_{Y} a_{C_1}(Y \cup Z_1) a_{C_2}(Y \cup Z_2) \equiv f(Z_1 \cup Z_2),$$

a function of the combined variables of the two cliques minus Y. Now this function cannot be accommodated by a clique in the moral graph because the variables u_1 and u_2 are not joined (and there may be others).

Hence we cannot form a potential representation of the joint distribution of $P(U - Y)$ on the moral graph with node Y removed. However, if we fill in the missing edges between the pairs of variables of the two cliques, then this marginal can be accommodated, and we can find a potential representation for $P(U - Y)$ on the reduced moral graph having these extra edges. This is why one adds edges to the moral graph, to be able to accommodate such intermediate marginal expressions. It turns out that one must fill-in sufficiently to form a triangulated graph, and doing so results in being able to set up a consistent message passing scheme.

13. Suggested further reading

Pearl is one of the pioneers who helped Bayesian methods for uncertain reasoning become popular in the artificial intelligence community. His textbook (Pearl, 1988) contains a wealth of material, from introducing probability theory and arguments for its use; axiomatics for graphical models; Markov properties; etc, to propagation in singly connected DAGs (ie prior to the development of making junction trees and propagating with them.) A good collection of papers on uncertain reasoning is Shafer and Pearl(1990), which covers not only probabilistic reasoning but also other formalisms for handling uncertainty. This also contains good overviews by the editors explaining the historical significance of the selected papers. An introductory review for probabilistic expert systems is (Spiegelhalter *et al.*, 1993). Each of these three references contain a large number of references for further reading.

Dawid(1979) introduced the axiomatic basis for treating conditional independence. More recent accounts of conditional independence with emphasis on graphical models and their Markov properties are given by Whittaker(1990) and Lauritzen(1996). (The latter also contains proofs of the lemmas stated in section 8.)

The Asia example was given by Lauritzen and Spiegelhalter(1988), who showed how to do consistent probability calculations in multiply connected DAG's using propagation, (it is also reprinted in (Shafer and Pearl, 1990)). Junction trees arise in other areas and are known by different names (eg join trees in relational databases); see (Lauritzen and Spiegelhalter, 1988) for more on this and also the discussion section of that paper. A recent and general formulation of propagation in junction trees is given by Dawid(1992). A recent introductory textbook on Bayesian networks is (Jensen, 1996).

References

Dawid, A. P. (1979). Conditional independence in statistical theory (with discussion). *Journal of the Royal Statistical Society, Series B*, **41**, pp. 1–31.

Dawid, A. P. (1992). Applications of a general propagation algorithm for probabilistic expert systems. *Statistics and Computing*, **2**, pp. 25–36.

Jensen, F. V. (1996). *An introduction to Bayesian networks.* UCL Press, London.

Lauritzen, S. L. (1996). *Graphical models.* OUP.

Lauritzen, S. L. and Spiegelhalter, D. J. (1988). Local computations with probabilities on graphical structures and their application to expert systems (with discussion). *Journal of the Royal Statistical Society, Series B*, **50**, pp. 157–224.

Pearl, J. (1988). *Probabilistic inference in intelligent systems.* Morgan Kaufmann, San Mateo.

Shafer, G. R. and Pearl, J. (ed.) (1990). *Readings in uncertain reasoning.* Morgan Kaufmann, San Mateo, California.

Spiegelhalter, D. J., Dawid, A. P., Lauritzen, S. L., and Cowell, R. G. (1993). Bayesian analysis in expert systems. *Statistical Science*, **8**, pp. 219–47.

Whittaker, J. (1990). *Graphical models in applied multivariate statistics.* John Wiley and Sons, Chichester.

ADVANCED INFERENCE IN BAYESIAN NETWORKS

ROBERT COWELL
City University, London.
The School of Mathematics, Actuarial Science and Statistics,
City University, Northampton Square, London EC1E 0HT

1. Introduction

The previous chapter introduced inference in discrete variable Bayesian networks. This used evidence propagation on the junction tree to find marginal distributions of interest. This chapter presents a tutorial introduction to some of the various types of calculations which can also be performed with the junction tree, specifically:

- Sampling.
- Most likely configurations.
- Fast retraction.
- Gaussian and conditional Gaussian models.

A common theme of these methods is that of a localized message-passing algorithm, but with different 'marginalisation' methods and potentials taking part in the message passing operations.

2. Sampling

Let us begin with the simple simulation problem. Given some evidence $\mathcal{E}$ on a (possibly empty) set of variable X_E, we might wish to simulate one or more values for the unobserved variables.

2.1. SAMPLING IN DAGS

Henrion proposed an algorithm called *probabilistic logic sampling* for DAGs, which works as follows. One first finds a topological ordering of the nodes of the DAG $\mathcal{G}$. Let us denote the ordering by $(X_1, X_2, \ldots, X_n)$ say after relabeling the nodes, so that all parents of a node precede it in the ordering, hence any parent of X_j will have an index $i < j$.

Assume at first that there is no evidence. Then one can sample X_1 from $P(X_1)$ to obtain x_1^* say. Then one samples a state for node X_2. Now if X_2 has no parents, then we can sample from $P(X_2)$ to obtain x_2^* say. Otherwise it has X_1 as a parent (by the topological ordering there are no other possibilities); hence we can sample from $P(X_2 \mid X_1 = x_1^*)$. In general in the jth stage we can sample from $P(X_j \mid \mathrm{pa}(X_j) = \pi^*)$ because all the parents will have already been sampled themselves and be in some definite state π^*. When we have sampled each of the nodes, we will obtain the case $u^* = (x_1^*, x_2^*, \ldots, x_n^*)$ with probability $P(U = u^*)$, ie sampled correctly from the full distribution $P(U)$. If we wish to generate a set of such cases, we begin over again with X_1. Each such case will be sampled independently.

Now suppose that we have evidence on one or more nodes. The previous scheme can still be applied, but one now introduces rejection steps to ensure samples are drawn from the correct distribution $P(U \mid \mathcal{E})$. Thus suppose we are at stage j, but that the state of X_j is known from the evidence to be $x_j^\dagger$ say. We cannot simply set $x_j^* = x_j^\dagger$ and then continue, because the resulting case will not have been drawn at random from the correct distribution. Instead, one samples X_j to obtain x_j^* say. Then, if it is the case that $x_j^* = x_j^\dagger$, we proceed to sample the next node, otherwise we start again at X_1, discarding all of the values generated for the current case. This rejection step ensures the correct balancing of probabilities so that when a complete case is successfully generated it is drawn from the distribution of interest. This rejection sampling is Henrion's probabilistic logic sampling.

However a problem is that with even quite small networks, let alone large ones, rejection increases exponentially with the number of nodes bearing evidence. We shall now see how the junction tree can be used to sample efficiently even with evidence.

2.2. SAMPLING USING THE JUNCTION TREE

Let us assume that we have propagated evidence and have a marginal representation of the joint probability on the junction tree:

$$P(U \mid \mathcal{E}) = \prod_C P(C \mid \mathcal{E}) / \prod_S P(S \mid \mathcal{E}).$$

To draw an analogy to the direct sampling from the DAG, let us now suppose that a clique is fixed as root, and that the edges of the junction tree are made directed as follows: the edges are directed between cliques and separators such that they all point away from the root. The result is a directed graph which because it is also a tree is also acyclic. Let us label the cliques and separators $(C_0, S_1, C_1, \ldots, S_m, C_m)$ such that C_0 is the root clique, that this ordering is a topological ordering of the nodes in the directed

tree, and with S_i the parent of C_i; see Figure 1. (Note that this notation

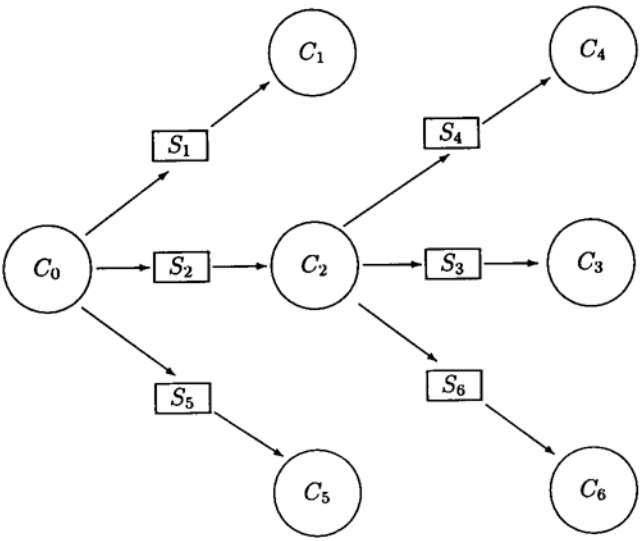


Figure 1. Directed junction tree.

has a subtlety that the tree is connected, but the disconnected case is easily dealt with.) Then we may divide the contents of the parent separator S_i into the clique table in C_i to obtain the following representation:

$$P(U \mid \mathcal{E}) = P(X_{C_0} \mid \mathcal{E}) \prod_{i=1}^{m} P(X_{C_i \setminus S_i} \mid X_{S_i}, \mathcal{E}).$$

This is called the *set-chain* representation, and is now in a form similar to the recursive factorization on the DAG discussed earlier, and can be sampled from in a similar manner. The difference is that instead of sampling individual variables at a time, one samples groups of variables in the cliques.

Thus one begins by sampling a configuration in the root clique, drawing from $P(X_{C_0} \mid \mathcal{E})$ to obtain $x^*_{C_0}$ say. Next one samples from $P(X_{C_1 \setminus S_1} \mid x^*_{S_1}, \mathcal{E})$ where the states of the variables for X_{S_1} are fixed by $x^*_{C_0}$ because $X_{S_1} \subset X_{C_0}$. One continues in this way, so that when sampling in clique C_i the variables X_{S_i} will already have been fixed by earlier sampling, as in direct sampling from the DAG. Thus one can sample directly from the correct distribution and avoid the inefficiencies of the rejection method.

3. Most likely configurations

One contributing reason why local propagation on the junction tree to find marginals "works" is that there is a "commutation behaviour" between the

operation of summation and the product form of the joint density on the tree, which allows one to move summation operations through the terms in the product, for example:

$$\sum_{A,B,C} f(A,B)f(B,C) = \sum_{A,B} \left(f(A,B) \sum_{C} f(B,C) \right).$$

However summation is not the only operation which has this property; another very useful operation is maximization, for example:

$$\max_{A,B,C} f(A,B)f(B,C) = \max_{A,B} \left(f(A,B) \max_{C} f(B,C) \right),$$

provided the factors are non-negative, a condition which will hold for clique and separator potentials representing probability distributions.

3.1. MAX-PROPAGATION

So suppose we have a junction tree representation of a probability distribution

$$P(U,\mathcal{E}) = \prod_{C} a(C) / \prod_{S} b(S)$$

and we pass messages of the form

$$b^*(S) = \max_{C \setminus S} a(C),$$

(which can be performed locally through the commutation property above) what do we get? The answer is the *max-marginal* representation of the joint density:

$$P(U,\mathcal{E}) = \frac{\prod_C P^{\max}(C,\mathcal{E})}{\prod_S P^{\max}(S,\mathcal{E})} \equiv \frac{\prod_C \max_{U \setminus C} P(U,\mathcal{E})}{\prod_S \max_{U \setminus S} P(S,\mathcal{E})}.$$

The interpretation is that for each configuration c^* of the variables in the clique C, the value $P_C^{\max}(c^*)$ is the highest probability value that any configuration of all the variables can take subject to the constraint that the variables of the clique have states c^*. (One simple consequence is that this most likely value appears at least once in every clique and separator.)

To see how this can come about, consider a simple tree with two sets of variables in each clique:

$$P(A,B,C,\mathcal{E}) = a(A,B) \frac{1}{b(B)} a(B,C).$$

Now recall that the message passing leaves invariant the overall distribution. So take $\boxed{AB}$ to be the root clique, and send then the first message, a maximization over C:

$$b^*(B) = \max_C a(B, C).$$

after "collecting" this message we have the representation:

$$P(A, B, C, \mathcal{E}) = \left(a(A, B)\frac{b^*(B)}{b(B)}\right)\frac{1}{b^*(B)}a(B, C).$$

The root clique now holds the table obtained by maximizing $P(A, B, C, \mathcal{E})$ over C, because

$$\begin{aligned} P^{\max}(A, B, \mathcal{E}) &\equiv \max_C P(A, B, C, \mathcal{E}) \\ &= \max_C \left(a(A, B)\frac{b^*(B)}{b(B)}\right)\frac{1}{b^*(B)}a(B, C) \\ &= \left(a(A, B)\frac{b^*(B)}{b(B)}\right)\max_C \frac{1}{b^*(B)}a(B, C) \\ &= \left(a(A, B)\frac{b^*(B)}{b(B)}\right). \end{aligned}$$

By symmetry the distribute message results in the second clique table holding the max-marginal value $\max_A P(A, B, C, \mathcal{E})$ and the intervening separator holding $\max_{A,C} P(A, B, C, \mathcal{E})$. The more general result can be obtained by induction on the numbers of cliques in the junction tree. (Note that one can pass back to the sum-marginal representation from the max-marginal representation by a sum-propagation.)

A separate but related task is to find the configuration of the variables which takes this highest probability. The procedure is as follows: first from a general potential representation with some clique C_0 chosen as root, perform a collect operation using maximization instead of summation. Then, search the root clique for the configuration of its variables, c_0^* say, which has the highest probability. Distribute this as extra "evidence", fixing successively the remaining variables in the cliques further from the root by finding a maximal configuration consistent with the neighbouring clique which has also been fixed, and including the states of the newly fixed variables as evidence, until all cliques have been so processed. The union of the "evidence" yields the most likely configuration. If there is "real" evidence then this is incorporated in the usual way in the collect operation. The interpretation is that the resulting configuration acts as a most likely explanation for the data.

Note the similarity to simulation, where one first does a collect to the root using ordinary marginalisation, then does a distribute by first randomly selecting a configuration from the root, and then randomly selecting configurations from cliques successively further out.

3.2. DEGENERACY OF MAXIMUM

It is possible to find the degeneracy of the most likely configuration, that is the total number of distinct configurations which have the same maximum probability $P^{\max}(U \mid \mathcal{E}) = p^*$ by a simple trick. (For most realistic applications there is unlikely to be any degeneracy, although this might not be true for eg. genetic-pedigree problems.) First one performs a max-propagation to obtain the max-marginal representation. Then one sets each value in each clique and separator to either 0 or 1, depending on whether or not it has attained the maximum probability value, thus:

$$I_C(x_C \mid \mathcal{E}) = \begin{cases} 1 & \text{if } P_C^{\max}(x_C \mid \mathcal{E}) = p^* \\ 0 & \text{otherwise,} \end{cases}$$

and

$$I_S(x_S \mid \mathcal{E}) = \begin{cases} 1 & \text{if } P_S^{\max}(x_S \mid \mathcal{E}) = p^* \\ 0 & \text{otherwise.} \end{cases}$$

Then

$$I(U \mid \mathcal{E}) = \prod_C I_C(x_C \mid \mathcal{E}) / \prod_S I_S(X_S \mid \mathcal{E})$$

is a potential representation of the indicator function of most likely configurations, a simple sum-propagation on this will yield the degeneracy as the normalization.

3.3. TOP N CONFIGURATIONS

In finding the degeneracy of the most likely configuration in the previous section, we performed a max-propagation and then set clique elements to zero which did not have the value of the highest probability. One might be tempted to think that if instead we set to zero all those elements which are below a certain threshold $p < 1$ then we will obtain the number of configurations having probability $\geq p$. It turns out that one can indeed find these configurations after one max-propagation, but unfortunately not by such a simple method. We will discuss a simplified version of an algorithm by Dennis Nilsson which allows one to calculate the top N configurations

by a sequence of propagations. (Nilsson(1997) has recently shown how they can be found after a single max-propagation.)

To begin with assume that we have no evidence. The first step is to have an ordering of the nodes, any ordering will do. Let us write this as $(X_1, X_2, \ldots, X_n)$, and let the j-th node have k_j states $(x_j^1, \ldots x_j^{k_j})$. We now do a max-propagation, and find the most likely configuration, denoted by $M^1 = (x_1^{m_1^1}, \ldots x_n^{m_n^1})$. Necessarily the second-most likely configuration, $M^2 = (x_1^{m_1^2}, \ldots x_n^{m_n^2})$, must differ from the most likely configuration in the state of at least one variable. So now we perform a further n max-propagations with "pseudo-evidence" $\mathcal{E}_j$ as follows.

$$
\begin{aligned}
\mathcal{E}_1 &= X_1 \neq x_1^{m_1^1} \\
\mathcal{E}_2 &= X_1 = x_1^{m_1^1} \text{ and } X_2 \neq x_2^{m_2^1} \\
\mathcal{E}_3 &= X_1 = x_1^{m_1^1} \text{ and } X_2 = x_2^{m_2^1} \text{ and } X_3 \neq x_3^{m_3^1} \\
&\vdots \\
\mathcal{E}_n &= X_1 = x_1^{m_1^1} \text{ and } \ldots X_{n-1} = x_{n-1}^{m_{n-1}^1} \text{ and } X_n \neq x_n^{m_n^1}
\end{aligned}
$$

By this procedure we partition the remaining set of configurations, excluding the most likely one M^1 we have already found. Hence one, and only one, of them has the second most likely configuration M^2, which can be found by looking at the (max-)normalizations of each.

Suppose the second most likely configuration was found in the jth set, ie by propagating $\mathcal{E}_j = X_1 = x_1^{m_1^1}$ and $\ldots$ $X_{j-1} = x_{j-1}^{m_{j-1}^1}$ and $X_j \neq x_j^{m_j^1}$. Now the third most likely configuration M^3 is to be found either in the other set of propagations, or it disagrees with the M^1 and M^2 configuration in at least one place. To find out which we need to perform a further $n-j+1$ propagations using "evidence" for each as follows:

$$
\begin{aligned}
\mathcal{E}_j^1 &= \mathcal{E}_j \text{ and } X_j \neq x_j^{m_j^2} \\
\mathcal{E}_j^2 &= \mathcal{E}_j \text{ and } X_j = x_j^{m_j^2} \text{and } X_{j+1} \neq x_{j+1}^{m_{j+1}^2}, \\
&\vdots \\
\mathcal{E}_j^{n-j+1} &= \mathcal{E}_j \text{ and } \ldots X_{n-1} = x_{n-1}^{m_{n-1}^2} \text{and } X_n \neq x_n^{m_n^2}.
\end{aligned}
$$

This further partitions the allowed states. After propagating these we can find the third most likely configuration. We can then find the fourth most likely configuration by essentially performing a similar partition on the third most likely configuration etc. The idea is quite simple, the main problem to develop a suitable notation to keep track of which partition one is up to.

If we have prior evidence, then we simply take this into account at the beginning, and ensure that the partitions do not violate the evidence. Thus for example if we have evidence about m nodes being in definite states, then instead of n propagations being required to find M^2 after having found M^1, we require instead only $n - m$ further propagations.

One application of finding a set of such most likely explanations is to explanation, ie, answering what the states of the unobserved variables are likely to be for a particular case. We have already seen that the most likely configuration offers such an explanation. If instead we have the top 10 or 20 configurations, then in most applications most of these will have most variables in the same state. This can confirm the diagnosis for most variables, but also shows up where diagnosis is not so certain (in those variables which differ between these top configurations). This means that if one is looking for a more accurate explanation one could pay attention to those variables which differ between the top configurations, hence serve to guide one to what could be the most informative test to do (cf. value of information).

The use of partitioned "dummy evidence" is a neat and quite general idea, and will probably find other applications.[1]

4. A unification

One simple comment to make is that minimization can be performed in a similar way to maximization. (In applications with logical dependencies the minimal configuration will have zero probability and there will be many such configurations. For example in the ASIA example half of the 256 configurations have zero probability.)

Another less obvious observation is that sum, max and min-propagation are all special cases of a more general propagation based upon L^p norms, used in functional analysis. Recall that the L^p norm of a non-negative real-valued function is defined to be

$$L^p(f) = \left(\int_{x \in X} f^p(x) dx \right)^{\frac{1}{p}}$$

For $p = 1$ this gives the usual integral, for $p \to \infty$ this give the maximum of the function over the region of integration, and for $p \to -\infty$ we obtain the minimum of f.

We can use this in our message propagation in our junction tree: the marginal message we pass from clique to separator is the L^p marginal,

[1] See for example (Cowell, 1997) for sampling without replacement from the junction tree.

defined by:

$$b_S^*(X_S) = \left(\sum_{X_C \setminus X_S} a_C^p(X_C) \right)^{\frac{1}{p}}$$

So that we can obtain the L^p marginal representation:

$$P(U \mid \mathcal{E}) = \frac{\prod_{C \in \mathcal{C}} P_C^{L^p}(X_C \mid \mathcal{E})}{\prod_{S \in \mathcal{S}} P_S^{L^p}(X_S \mid \mathcal{E})}$$

which is an *infinite-family* of representations. Apart from the L^2 norm, which may have an application to quadratic scoring of models, it is not clear if this general result is of much practical applicability, though it may have theoretical uses.

5. Fast retraction

Suppose that for a network of variables X we have evidence on a subset of k variables U: $\mathcal{E} = \{\mathcal{E}_u : u \in U^*\}$, with $\mathcal{E}_u$ of the form "$X_u = x_u^*$" Then it can be useful to compare each item of evidence with the probabilistic prediction given by the system for X_u on the basis of the remaining evidence $\mathcal{E}_{\setminus\{u\}}$: "$X_v = x_v^*$ for $v \in U \setminus \{u\}$", as expressed in the conditional density of X_u given $\mathcal{E}_{\setminus\{u\}}$. If we find that abnormally low probabilities are being predicted by the model this can highlight deficiencies of the model which could need attention or may indicate a rare case is being observed.

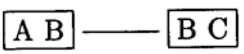
Now one "brute force" method to calculate such probabilities is to perform k separate propagations, in which one takes out in turn the evidence on each variable in question and propagates the evidence for all of the remaining variables.

However it turns out that yet another variation of the propagation algorithm allows one to calculate all of these predictive probabilities in one propagation, at least for the case in which the joint probability is strictly positive, which is the case we shall restrict ourselves to here. (For probabilities with zeros it may still be possible to apply the following algorithm; the matter depends upon the network and junction tree. For the Shafer-Shenoy message passing scheme the problem does not arise because divisions are not necessary.) Because of the computational savings implied, the method is called *fast-retraction.*

5.1. OUT-MARGINALISATION

The basic idea is to work with a potential representation of the prior joint probability even when there is evidence. This means that, unlike the earlier

sections, we do not modify the clique potentials by multiplying them by the evidence likelihoods. Instead we incorporate the evidence only into forming the messages, by a new marginalisation method called *out-marginalisation*, which will be illustrated for a simple two-clique example:



Here A, B and C are disjoint sets of variables, and the clique and separator potentials are all positive. Suppose we have evidence on variables $\alpha \in A$, $\beta \in B$ and $\gamma \in C$. Let us denote the evidence functions by h_α, h_β and h_γ, where h_α is the product of the evidence likelihoods for the variables $\alpha \in A$ etc. Then we have

$$\begin{aligned} P(ABC) &= g(AB) \otimes \frac{1}{g(B)} \otimes g(BC) \\ P(ABC, \mathcal{E}_\alpha) &= P(ABC) h_\alpha \\ P(ABC, \mathcal{E}_\gamma) &= P(ABC) h_\gamma \\ P(ABC, \mathcal{E}_\alpha, \mathcal{E}_\gamma) &= P(ABC) h_\alpha h_\gamma. \end{aligned}$$

where the g's are the clique and separator potentials.

We take the clique $\boxed{AB}$ as root. Our first step is to send an *out-marginal message* from $\boxed{BC}$ to $\boxed{AB}$ defined as:

$$g^*(B) = \sum_C g(BC) h_\gamma.$$

That is we only incorporate into the message that subset of evidence about the variables in C, thus excluding any evidence that may be relevant to the separator variables B. Note that because we are using the restriction that the joint probability is non zero for every configuration, this implies that the potentials and messages are also non zero. Sending this message leaves the overall product of junction tree potentials invariant as usual:

$$P(ABC) = \frac{g(AB) g^*(B)}{g(B)} \otimes \frac{1}{g^*(B)} \otimes g(BC).$$

Now let us use this representation to calculate

$$\begin{aligned} P^{out}(AB, \mathcal{E}_{\setminus A\cup B}) &\equiv \sum_C P(ABC, \mathcal{E}_\gamma) \\ &= \sum P(ABC)h_\gamma \\ &= \sum_C \frac{g(AB)g^*(B)}{g(B)} \otimes \frac{1}{g^*(B)} \otimes g(BC)h_\gamma \\ &= \frac{g(AB)g^*(B)}{g(B)} \otimes \frac{1}{g^*(B)} \left(\sum_C g(BC)h_\gamma = g^*(B) \right) \\ &= \frac{g(AB)g^*(B)}{g(B)}. \end{aligned}$$

This shows that the content of the clique $\boxed{AB}$ is simply the out-margin of the joint probability. We can now send and out-margin message back, and by symmetry the content of the clique $\boxed{BC}$ will also be the out-margin of the joint probability, with marginalisation taken over A. The separator potential $g(B)$ is also an out-marginalisation: $\sum_A P^{out}(AB, \mathcal{E}_{\setminus A\cup B})h_\alpha \equiv P^{out}(B, \mathcal{E}_{\setminus B})$. We thus arrive at the following out-marginal representation of the joint probability:

$$P(ABC) = P^{out}(AB, \mathcal{E}_{\setminus A\cup B}) \frac{1}{P^{out}(B, \mathcal{E}_{\setminus B})} P^{out}(BC, \mathcal{E}_{\setminus B\cup C}).$$

Further out-marginalisation of these clique potentials will then yield the desired predictive probabilities for the individual variables having evidence.

In general we have the following out-marginal representation:

$$P(U) = \frac{\prod_C P^{out}(C, \mathcal{E}_{\setminus C})}{\prod_S P^{out}(S, \mathcal{E}_{\setminus S})},$$

which follows, from the simple two-clique case previously described, by induction on the number of cliques in the tree.

Fast retraction, where applicable, has another use besides comparing predictive probabilities against evidence. Consider the previous way of dealing with evidence, in which the clique potentials are multiplied by the evidence likelihoods. After propagating evidence about one case, one might wish to alter the evidence to look at another case. To do this would require a re-initialisation of the junction tree. However, by propagating evidence with fast-retraction, then because the tree always retains a potential representation of the joint probability, there is no need to re-initialise the junction tree between cases.

6. Modelling with continuous variables

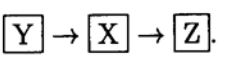
All examples and discussion has have been restricted to the special case of discrete random variables. In principle, however, there is no reason why we should not build models having continuous random variables as well as, or instead of, discrete random variables, with more general conditional probability densities to represent the joint density, and use local message passing to simplify the calculations. In practice the barrier to such general applicability is the inability of performing the required integrations in closed form representable by a computer. (Such general models can be analyzed by simulation, for example Gibbs sampling.)

However there is a case for which such message passing is tractable, and that is when the random variables are such that the overall distribution is multivariate-Gaussian. This further extends to the situation where both discrete and continuous random variables coexist within a model having a so called *conditional-gaussian* joint distribution.

We will first discuss Gaussian models, and then discuss the necessary adjustments to the theory enabling analysis of mixed models with local computation.

7. Gaussian models

Structurally, the directed Gaussian model looks very much like the discrete models we have already seen. The novel aspect is in their numerical specification. Essentially, the conditional distribution of a node given its parents is given by a Gaussian distribution with expectation linear in the values of the parent nodes, and variance independent of the parent nodes. Let us take a familiar example:



Node Y, which has no parents, has a normal distribution given by

$$N_Y(\mu_Y; \sigma_Y^2) \propto \exp\left(\frac{-(y-\mu_Y)^2}{2\sigma_Y^2}\right),$$

where μ_Y and σ_Y are constants. Node X has node Y as a parent, and has the conditional density:

$$N_X(\mu_X + \beta_{X,Y}y; \sigma_X^2) \propto \exp\left(\frac{-(x-\mu_X-\beta_{X,Y}y)^2}{2\sigma_X^2}\right),$$

where μ_X, $\beta_{X,Y}$ and σ_X are constants. Finally, node Z has only X as a parent; its conditional density is given by

$$N_Z(\mu_Z + \beta_{Z,X}x; \sigma_Z^2) \propto \exp\left(\frac{-(z - \mu_Z - \beta_{Z,X}x)^2}{2\sigma_Z^2}\right).$$

In general, if a node X had parents $\{Y_1, \ldots, Y_n\}$ it would have a conditional density:

$$N_X(\mu_X + \sum_i \beta_{X,Y_i} y_i; \sigma_X^2) \propto \exp\left(\frac{-(x - \mu_X - \sum_i \beta_{X,Y_i} y_i)^2}{2\sigma_X^2}\right).$$

Now the joint density is obtained by multiplying together the separate component Gaussian distributions:

$$\begin{aligned} P(X,Y,Z) &= N_Y(\mu_Y; \sigma_Y^2) N_X(\mu_X + \beta_{X,Y}y; \sigma_X^2) N_Z(\mu_Z + \beta_{Z,X}x; \sigma_Z^2) \\ &\propto \exp\left(-\frac{1}{2}(x - \mu_X, y - \mu_Y, z - \mu_Z) K (x - \mu_X, y - \mu_Y, z - \mu_Z)^T\right), \end{aligned}$$

where K is a symmetric (positive definite) 3×3 matrix, and T denotes transpose. In a more general model with n nodes, one obtains a similar expression with an $n \times n$ symmetric (positive definite) matrix.

Expanding the exponential, the joint density can be written as:

$$\exp\left((x \quad y \quad z)\begin{pmatrix} h_X \\ h_Y \\ h_Z \end{pmatrix} - \frac{1}{2}(x \quad y \quad z)\begin{pmatrix} K_{XX} & K_{XY} & K_{XZ} \\ K_{YX} & K_{YY} & K_{YZ} \\ K_{ZX} & K_{ZY} & K_{ZZ} \end{pmatrix}\begin{pmatrix} x \\ y \\ z \end{pmatrix}\right)$$

where $h_X = \mu_X/\sigma_X^2 + \mu_Z\beta_{Z,X}/\sigma_Z^2$ etc. This form of the joint density is the most useful for constructing local messages, and indeed local messages will consist of functions of this type. Let us now define them and list the properties we shall be using.

7.1. GAUSSIAN POTENTIALS

Suppose we have n continuous random variables $X_1, \ldots, X_n$. A Gaussian potential in a subset $\{Y_1, \ldots, Y_k\}$ of variables is a function of the form:

$$\exp\left(g + (y_1 \quad \ldots \quad y_k)\begin{pmatrix} h_1 \\ \vdots \\ h_k \end{pmatrix} - \frac{1}{2}(y_1 \quad \ldots \quad y_k)\begin{pmatrix} K_{1,1} & \ldots & K_{1,k} \\ \vdots & \ddots & \vdots \\ K_{k,1} & \ldots & K_{k,k} \end{pmatrix}\begin{pmatrix} y_1 \\ \vdots \\ y_k \end{pmatrix}\right)$$

where K is a constant positive definite $k \times k$ matrix, h is a k dimensional constant vector and g is a number. For shorthand we write this as a

triple, $\phi = (g, h, K)$. Gaussian potentials can be multiplied by adding their respective triples together:

$$\phi_1 * \phi_2 = (g_1 + g_2, h_1 + h_2, K_1 + K_2).$$

Similarly division is easily handled:

$$\phi_1 / \phi_2 = (g_1 - g_2, h_1 - h_2, K_1 - K_2).$$

These operations will be used in passing the "update factor" from separator to clique.

To initialize cliques we shall require the extension operation combined with multiplication. Thus a Gaussian potential defined on a set of variables Y is extended to a larger set of variables by enlarging the vector h and matrix K to the appropriate size and setting the new slots to zero. Thus for example: $\phi(x) = \exp(g + x^T h - \frac{1}{2} x^T K x)$ extends to

$$\phi(x, y) = \phi(x) = \exp\left(g + \begin{pmatrix} x & y \end{pmatrix} \begin{pmatrix} h \\ 0 \end{pmatrix} - \frac{1}{2} \begin{pmatrix} x & y \end{pmatrix} \begin{pmatrix} K & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}\right).$$

Finally, to form the messages we must define marginalisation, which is now an integration. Let us take Y_1 and Y_2 to be two sets of distinct variables, and

$$\phi(y_1, y_2) = \exp\left(g + \begin{pmatrix} y_1 & y_2 \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \end{pmatrix} - \frac{1}{2} \begin{pmatrix} y_1 & y_2 \end{pmatrix} \begin{pmatrix} K_{1,1} & K_{1,2} \\ K_{2,1} & K_{2,2} \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}\right)$$

so that the h and K are in blocks. Then integrating over Y_1 yields a new vector h and matrix K as follows:

$$h = h_2 - K_{2,1} K_{1,1}^{-1} h_1$$

$$K = K_{2,2} - K_{2,1} K_{1,1}^{-1} K_{1,2}.$$

(Discussion of the normalization will be omitted, because it is not required except for calculating probability densities of evidence.) Thus integration has a simple algebraic structure.

7.2. JUNCTION TREES FOR GAUSSIAN NETWORKS

Having defined the directed Gaussian model, the construction of the junction tree proceeds exactly as for the discrete case, as far as the structure is concerned. The difference is with the initialization.

A Gaussian potential of correct size is allocated to each clique and separator. They are initialized with all elements equal to zero.

Next for each conditional density for the DAG model, a Gaussian potential is constructed to represent it and multiplied into any one clique which contains the node and its parents, using extension if required.

The result is a junction tree representation of the joint density. Assuming no evidence, then sending the clique marginals as messages results in the clique marginal representation, as for the discrete case:

$$P(U) = \prod_C P(X_C) / \prod_S P(X_S).$$

Care must be taken to propagate evidence. By evidence $\mathcal{E}$ on a set of nodes Y we mean that each node in Y is observed to take a definite value. (This is unlike the discrete case in which some states of a variable could be excluded but more than one could still be entertained.) Evidence about a variable must be entered into every clique and separator in which it occurs. This is because when evidence is entered on a variable it reduces the dimensions of every h vector and K matrix in the cliques and separators in which it occurs.

Thus for example, let us again take Y_1 and Y_2 to be two sets of distinct variables, and

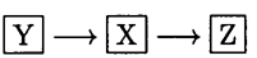
$$\phi(y_1, y_2) \propto \exp\left(\begin{pmatrix} y_1 & y_k \end{pmatrix} \begin{pmatrix} h_1 \\ h_k \end{pmatrix} - \frac{1}{2} \begin{pmatrix} y_1 & y_k \end{pmatrix} \begin{pmatrix} K_{1,1} & K_{1,2} \\ K_{2,1} & K_{2,2} \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \right)$$

so that the h and K are again in blocks. Suppose we now observe the variables of Y_2 to take values y_2^*. Then the potentials become modified to $h = h_1 - y_2^* K_{2,1}$ and $K = K_{1,1}$.

After such evidence has been entered in every clique and separator, then the standard propagation will yield the clique-marginal density representation with evidence included. Further within clique marginals then gives the (Gaussian) distributions on individual nodes.

7.3. EXAMPLE

Let us take out three node example again, with initial conditional distributions as follows:



$$\begin{aligned} N(Y) &= N(0,1) \\ N(X \mid Y) &= N(y,1) \\ N(Z \mid X) &= N(x,1) \end{aligned}$$

The cliques for this tree are $\boxed{\text{X Y}}$ and $\boxed{\text{X Z}}$. After initializing and propagating, the clique potentials are

$$\phi(x,y) \propto \exp\left(-\frac{1}{2}\begin{pmatrix} x & y \end{pmatrix}\begin{pmatrix} 1 & -1 \\ -1 & 2 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix}\right)$$
$$\phi(x,z) \propto \exp\left(-\frac{1}{2}\begin{pmatrix} x & z \end{pmatrix}\begin{pmatrix} 1.5 & -1 \\ -1 & 1 \end{pmatrix}\begin{pmatrix} x \\ z \end{pmatrix}\right)$$

with separator $\phi(x) \propto \exp(-x^2/4)$; Now if we enter evidence $X = 1.5$, say, then the potentials reduce to:

$$\phi(X = 1.5, y) \propto \exp(1.5y - y^2)$$

and

$$\phi(X = 1.5, z) \propto \exp(1.5z - \frac{1}{2}z^2),$$

because in this example X makes up the separator between the two cliques. The marginal densities are then:

$$P(Y) = N(0.75, 0.5) \text{ and } P(Z) = N(1.5, 1).$$

Alternatively, suppose we take $\boxed{\text{X Y}}$ as the root clique, and enter evidence that $Z = 1.5$. Then the message from $\boxed{\text{X Z}}$ to $\boxed{\text{X Y}}$ is given by $\phi(X) \propto \exp(1.5x - 0.75x^2)$ so that after propagation the clique potential on $\boxed{\text{X Y}}$ is of the form:

$$\phi(x,y) \propto \exp\left(\begin{pmatrix} x & y \end{pmatrix}\begin{pmatrix} 1.5 \\ 0 \end{pmatrix} - \frac{1}{2}\begin{pmatrix} x & y \end{pmatrix}\begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix}\right)$$

with marginal densities

$$P(X) = N(1, 2/3) \text{ and } P(Y) = N(1/2, 2/3).$$

8. Conditional Gaussian models

As we have seen, the treatment of Gaussian networks is much the same as for discrete models. The minor differences are in (1) the nature of the potentials employed, and (2) evidence has to be entered into every clique and separator.

The passage to mixed models proceeds with some more important differences. The first is a restriction in the modeling stage: *continuous variables*

are not allowed to have discrete children, ie. discrete nodes can only have discrete parents. The conditional probabilities specified for discrete nodes differ in character to those of continuous nodes. The former are again simple tables, as for discrete models, the latter are Gaussian potentials, but with the constants g, vectors h and matrices K indexed by the parent configurations of the discrete parents. Also, because certain sub configurations of discrete variables might not be allowed, we need to include indicator functions on the Gaussian potentials and we have to more careful with the normalization constants g.

The following is a brief guide to the theory, for more details see the original paper by Lauritzen, whose notation we follow closely here.

8.1. CG-POTENTIALS

The set of variables V is partitioned into the *discrete* variables (Δ) and continuous variables (Γ), thus $V = \Delta \cup \Gamma$. Let $x = (i, y)$ denote a typical element of the joint state space with i denoting the values of the discrete variables and y the values of the continuous variables. The joint density is assumed to be a *CG distribution*, which means that it has the form f with

$$f(x) = f(i, y) = \chi(i) \exp\left\{g(i) + y^T h(i) - y^T K(i) y/2\right\},$$

where $\chi(i) \in \{0, 1\}$ indicates whether f is positive at i. The triple (g, h, K) is called the *canonical characteristics* of the distribution; it is only defined for $\chi(i) > 0$ but when that is the case one can define the *moment characteristics*, denoted by the triple $\{p, \xi, \Sigma\}$ and given by

$$\xi(i) = K(i)^{-1} h(i), \quad \Sigma(i) = K(i)^{-1}.$$

Inverting, we have the canonical characteristics are $K(i) = \Sigma(i)^{-1}$, $h(i) = K(i)\xi(i)$, and

$$g(i) = \log p(i) + \left\{\log\det K(i) - |\Gamma| \log(2\pi) - \xi(i)^T K(i) \xi(i)\right\}/2.$$

As for the Gaussian networks, we generalize CG distributions to *CG potentials* which are any functions ϕ of the form

$$\phi(x) = \phi(i, y) = \chi(i) \exp\{g(i) + y^T h(i) - y^T K(i) y/2\}.$$

$K(i)$ is restricted to be symmetric, though not necessarily invertible. However we still call the triple (g, h, K) the canonical characteristics, and if for all i, $\chi(i) > 0$ and $K(i)$ is positive definite then the moment characteristics are given as before.

Multiplication, division and extension proceed as for the Gaussian potentials have already been discussed. Marginalisation is however different,

because adding two CG potentials in general will result in a mixture of CG potentials – a function of a different algebraic structure. Thus we need to distinguish two types of marginalisation – *strong* and *weak*.

8.2. MARGINALISATION

Marginalising continuous variables corresponds to integration. Let

$$y = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}, \quad h = \begin{pmatrix} h_1 \\ h_2 \end{pmatrix}, \quad K = \begin{pmatrix} K_{11} & K_{12} \\ K_{21} & K_{22} \end{pmatrix}$$

with y_1 having dimension p and y_2 dimension q with K_{11} is positive definite. Then the integral $\int \phi(i, y_1, y_2)\, dy_1$ is finite and equal to a CG potential $\tilde{\phi}$ with canonical characteristics given as

$$\begin{aligned} \tilde{g}(i) &= g(i) + \{p \log(2\pi) - \log\det K_{11}(i) + h_1(i)^T K_{11}(i)^{-1} h_1(i)\}/2 \\ \tilde{h}(i) &= h_2(i) - K_{21}(i) K_{11}(i)^{-1} h_1(i) \\ \tilde{K}(i) &= K_{22}(i) - K_{21}(i) K_{11}(i)^{-1} K_{12}(i). \end{aligned}$$

Marginalising discrete variables corresponds to summation. Since in general addition of CG potentials results is a mixture of CG potentials, an alternative definition based upon the moment characteristics $\{p, \xi, \Sigma\}$ is used which does result in a CG potential; however it is only well defined for $K(i,j)$ positive definite. Specifically, the marginal over the discrete states of $\tilde{\phi}$ is *defined* as the CG potential with moment characteristics $\{\tilde{p}, \tilde{\xi}, \tilde{\Sigma}\}$ where

$$\tilde{p}(i) = \sum_j p(i,j), \quad \tilde{\xi}(i) = \sum_j \xi(i,j) p(i,j)/\tilde{p}(i), \text{ and}$$

$$\tilde{\Sigma}(i) = \sum_j \Sigma(i,j) p(i,j)/\tilde{p}(i) + \sum_j \left(\xi(i,j) - \tilde{\xi}(i)\right)^T \left(\xi(i,j) - \tilde{\xi}(i)\right) p(i,j)/\tilde{p}(i).$$

Note that the latter can be written as

$$\tilde{p}(i) \left(\tilde{\Sigma}(i) + \tilde{\xi}(i)^T \tilde{\xi}(i)\right) = \sum_j p(i,j) \left(\Sigma(i,j) + \xi(i,j)^T \xi(i,j)\right).$$

so that if $\Sigma(i,j)$ and $\xi(i,j)$ are independent of j then they can be taken through the summations as constants. This observation is used to define a marginalisation over both continuous and discrete variables: First marginalise over the the continuous variables and then over the discrete variables. If, after marginalising over the continuous variables the resulting pair (h, K) is independent of the discrete variables to be marginalised over, (summation over these discrete variables then leaves the pair (h, K)

unmodified), we say that we have a *strong marginalisation.* Otherwise one sums over the discrete variables using the moment characteristics, and the overall marginalisation is called a *weak marginalisation.*

Weak and strong marginalisation satisfy composition:

$$\sum_A \left(\sum_B \phi_{A \cup B \cup C} \right) = \sum_{A \cup B} \phi_{A \cup B \cup C},$$

but in general only the strong marginalisation satisfies

$$\sum_A (\phi_{A \cup B} \psi_B) = \psi_B \left(\sum_A \phi_{A \cup B} \right).$$

Under both type of marginalisation, a 'marginalised' density will then have the correct moments to order 2, i.e.

$$P(I = i) = \tilde{p}(i), \quad \mathbf{E}(Y \mid I = i) = \tilde{\xi}(i), \quad \mathbf{V}(Y \mid I = i) = \tilde{\Sigma}(i),$$

where the correct CG distribution is used to take expectations.

8.3. MAKING THE JUNCTION TREE

The non-closure of cg-potentials under marginalisation of discrete variables means that we have to adjust to how we construct the junction tree and pass messages in it. The first step is to construct the junction tree. First we moralize the DAG in the usual way. Then we triangulate by a restricted elimination ordering.[2] Specifically, we first eliminate all of the continuous variables, and then we eliminate the discrete variables. Then from the resulting cliques we can construct a junction tree. Now we must select a root. Unlike the previous pure cases, we cannot freely choose any clique of the junction tree. Instead we choose a so called *strong root* defined as follows:

> *Any clique R which for any pair A, B of cliques themselves neighbours on the tree with A closer to R than B satisfies*
>
> $$(B \setminus A) \subseteq \Gamma \text{ or } (B \cap A) \subseteq \Delta.$$

Thus, when a separator between two neighbouring cliques is not purely discrete, the clique furthest away from the root has only continuous vertices extra beyond the separator.

[2]One way to triangulate a graph is to take an ordering of the nodes and give all of the nodes the status 'unmarked'. One then works through each node in turn, marking it and joining all pairs of its unmarked neighbours. The ordering is called an *elimination ordering.* Finding good triangulations is then equivalent to finding good elimination orderings.

8.4. PROPAGATION ON THE JUNCTION TREE

The point of this restriction is that on the collect operation, only strong marginalisations are required to be performed. This is because our restricted elimination ordering - getting rid of the continuous variables first, is equivalent to doing the integrations over the continuous variables before marginalising any of the discrete variables.

Thus our message passing algorithm takes the form:

1. Initialization: Set all clique and separator potentials to zero with unit indicators, and multiply in the model specifying potentials using the extension operation where appropriate.
2. Enter evidence into all clique and separator potentials, reducing vector and matrix sizes as necessary.
3. Perform a collect operation to the strong root, where the messages are formed by strong marginalisation by first integrating out the redundant continuous variables, and then summing over discrete variables.
4. Perform a distribute operation, using weak marginalisation where appropriate when mixtures might be formed on marginalising over the discrete variables.

The result is a representation of the joint CG-distribution including evidence, because of the invariant nature of the message passing algorithm. Furthermore, because of the use of weak marginalisation for the distribute operation, the marginals on the cliques will themselves be CG-distributions whose first two moments match that of the full distribution. The following is an outline sketch of why this could be.

First by the construction of the junction tree, all collect operations are strong marginals, so that after a collect-to-root operation the root clique contains a strong marginal. Now suppose, for simplicity, that before the distribute operation we move to a set-chain representation (cf. section 2.2). Then apart from the strong root, each clique will have the correct joint density $P(X_{C_i \setminus S_i} \mid X_{S_i})$ where S_i is the separator adjacent to the clique C_i on the path between it and the strong root. Now on the distribute operation the clique C_i will be multiplied by a CG-potential which will either be a strong marginal or a weak marginal. If the former then the clique potential will be the correct marginal joint density. If the latter then we may write the clique potential as the product $P(X_{C_i \setminus S_i}) * Q(X_{S_i})$ where Q is the correct weak marginal for the variables X_{S_i}. Now consider taking an expectation of any linear or quadratic function of the X_{C_i} with respect to this "density". We are free to integrate by parts. However, choosing to integrate wrt. $X_{C_i \setminus S_i}$ first means that we form the expectation wrt the correct CG-density $P(X_{C_i \setminus S_i} \mid X_{S_i})$, and will thus end up with a correct expectation (which will be a linear or quadratic function in the X_{S_i}) multiplied

by the correct weak marginal $Q(X_{S_i})$. Hence performing these integrations we will obtain the correct expectation of the original function wrt the true joint density.

For brevity some details have been skipped over here, such as showing that the separator messages sent are correct weak marginals. Detailed justifications and proofs use induction combined with a careful analysis of the messages sent from the strong root on the distribute operation. See the original paper for more details.

9. Summary

This tutorial has shown the variety of useful applications to which the junction-tree propagation algorithm can be used. It has not given the most general or efficient versions of the algorithms, but has attempted to present the main points of each so that the more detailed descriptions in the original articles will be easier to follow. There are other problems, to which the junction-tree propagation algorithm can be applied or adapted to, not discussed here, such as:

- Influence diagrams: Discrete models, with random variables, decisions and utilities. Potentials are now doublets representing probabilities and utilities. Junction tree is generated with a restricted elimination generalising that for cg-problems to emulate solving the decision tree.
- Learning probabilities. Nodes presenting parametrisations of probabilities can be attached to networks, and Bayesian updating performed using the same framework.
- Time series. A network can represent some state at a given time, and they can be chained together to form a time-window for dynamic modelling. The junction tree can be expanded and contracted to allow forward-prediction or backward smoothing.

Doubtless new examples will appear in the future.

10. Suggested further reading

Probabilistic logic sampling for Bayesian networks is described by Henrion(1988). A variation of the method – *likelihood-weighting sampling* – in which rejection steps are replaced by a weighting scheme is given by Shachter and Peot(1989). Drawing samples directly from the junction tree is described by Dawid(1992), which also shows how the most likely configuration can be found from the junction tree. The algorithm for finding the N- most likely configurations is due to Nilsson(1994), who has also developed a more efficient algorithm requiring only one max-propagation on the junction tree. L^p-propagation is not described anywhere but here.

Fast retraction is introduced in (Dawid, 1992) and developed in more detail in (Cowell and Dawid, 1992).

Gaussian networks are described by Shachter and Kenley(1989), who use arc-reversal and barren-node reduction algorithms for their evaluation. (The equivalence of various evaluation schemes is given in (Shachter *et al.*, 1994).) The treatment of Gaussian and conditional-gaussian networks is based on the original paper by Lauritzen(1992). For pedagogical reasons this chapter specialized the conditional-gaussian presentation of (Lauritzen, 1992) to the pure gaussian case, to show that the latter is not so different from the pure discrete case. Evaluating influence diagrams by junction-trees is treated in (Jensen *et al.*, 1994). For an extensive review on updating probabilities (Buntine, 1994). Dynamic junction trees for handling time series is described by Kjærulff(1993). See also (Smith *et al.*, 1995) for an application using dynamic junction trees not derived from a DAG model.

References

Buntine, W. L. (1994). Operations for learning with graphical models. *Journal of Artificial Intelligence Research*, **2**, pp. 159–225.

Cowell, R. G. (1997). Sampling without replacement in junction trees, Research Report 15, Department of Actuarial Science and Statistics, City University, London.

Cowell, R. G. and Dawid, A. P. (1992). Fast retraction of evidence in a probabilistic expert system. *Statistics and Computing*, **2**, pp. 37–40.

Dawid, A. P. (1992). Applications of a general propagation algorithm for probabilistic expert systems. *Statistics and Computing*, **2**, pp. 25–36.

Henrion, M. (1988). Propagation of uncertainty by probabilistic logic sampling in Bayes' networks. In *Uncertainty in Artificial Intelligence*, (ed. J. Lemmer and L. N. Kanal), pp. 149–64. North-Holland, Amsterdam.

Jensen, F., Jensen, F. V., and Dittmer, S. L. (March 1994). From influence diagrams to junction trees. Technical Report R-94-2013, Department of Mathematics and Computer Science Aalborg University, Denmark.

Kjærulff, U. (1993). A computational scheme for reasoning in dynamic probabilistic networks. Research Report R-93-2018, Department of Mathematics and Computer Science, Aalborg University, Denmark.

Lauritzen, S. L. (1992). Propagation of probabilities, means and variances in mixed graphical association models. *Journal of the American Statistical Association*, **87**, pp. 1098–108.

Nilsson, D. (1994). An algorithm for finding the most probable configurations of discrete variables that are specified in probabilistic expert systems. M.Sc. Thesis, Department of Mathematical Statistics, University of Copenhagen.

Nilsson, D. (1997). An efficient algorithm for finding the M most probable configurations in a probabilistic expert system. Submitted to *Statistics and Computing*.

Shachter, R. D., Andersen, S. K., and Szolovits, P. (1994). Global conditioning for probabilistic inference in belief networks. In *Proceedings of the Tenth Conference on Uncertainty in Artifical Intelligence*, pp 514–522.

Shachter, R. and Kenley, C. (1989). Gaussian influence diagrams. *Management Science*, **35**, pp. 527–50.

Shachter, R. and Peot, M. (1989). Simulation approaches to general probabilistic inference on belief networks. In *Uncertainty in Artificial Intelligence 5*, (ed. M. Henrion, R. D. Shachter, L. Kanal, and J. Lemmer), pp. 221–31. North-Holland, North-

Holland.
Smith, J. Q., French, S., and Raynard, D. (1995). An efficient graphical algorithm for updating the estimates of the dispersal of gaseous waste after an accidental release. In *Probabilistic reasoning and Bayesian belief networks*, (ed. A. Gammerman), pp. 125–44. Alfred Waller, Henley-on-Thames.