Probabilistic Graphical Models

Michael I. Jordan
Christopher M. Bishop

January 29, 2001

Chapter 1

Decision Graphs

In earlier chapters of this book we have developed extensive graphical machinery for solving the inference problem, that is the task of evaluating conditional probabilities given evidence in the form of observed values for random variables. For practical applications, however, the role of the probabilities is an intermediate one in the task of making decisions, or equivalently taking actions. In order to make decisions which are optimal according to some criterion, we combine the inferred conditional probabilities with a utility (or cost) function. In the simplest case we have a single decision to take, and the optimal decision is generally defined to be that which maximizes the expected utility.

More complex decision scenarios involve multiple, sequential stages of decision making, in which new information is revealed after each decision is made. In this chapter we review the basic concepts of decision theory and then we demonstrate the role which graphical models can play in arriving at optimal decisions for complex problems involving multiple decision stages. In particular, we shall formulate the multi-stage decision problem in terms of directed acyclic graphs having both decision nodes and utility nodes in addition to the usual stochastic nodes describing random variables. We shall then develop an exact and efficient algorithm for finding the optimal set of decisions together with the corresponding maximum utility. Our approach is based on an extension of the elimination algorithm introduced in Chapter ??. We shall restrict attention in this chapter to problems in which the random variables as well as the decision variables are all discrete.

1.1 Decision Theory

1.1.1 Single-step Decisions

As a simple example of a decision problem, suppose we have to decide whether to provide treatment for a patient who might potentially have cancer, in which the treatment itself has undesirable side effects. There is a single binary random variable X which describes whether or not the patient has cancer, and there is a single binary decision variable d corresponding to the decision to provide treatment or not. Finally there is a utility function which depends on both X and d, and which is therefore a 2×2 matrix, an example of which is shown in Figure 1.1. In this example we see that there is negative utility associated with having cancer, but that this is diminished if treatment is provided. For a patient with no cancer there is a negative utility associated with receiving treatment due to side effects associated with the treatment itself.

At the time the decision must be taken, the state of the random variable X is unknown. Thus the actually utility arising from a particular decision will not be known when the decision is made. We can, however, calculate the expected utility by marginalizing over the random variable. This

$$\begin{pmatrix} 0 & -30 \\ -1000 & -200 \end{pmatrix} \tag{1.1}$$

Figure 1.1: An example of a utility matrix $U(X, d)$ for the cancer treatment problem. The first and second rows correspond to the states of the random variable X = 'normal' and X = 'cancer', while first and second columns correspond to the states of the decision variable d = 'no treatment' and d = 'treatment'.

allows us to choose the decision variable using the criterion of *maximum expected utility*. While such a criterion is intuitively appealing, it can also be justified more formally from a frequentist perspective as being the criterion which yields the greatest reward in a long run series of decisions (for example in a gambling scenario). An analogous argument can be made from a Bayesian perspective, in which case it applies also to the situation where there is only a single trial.

In order find the expected utility for the cancer example, however, we need to know the probability that the patient has cancer. For example, suppose that

$$P(X = \text{'normal'}) = 0.9 \tag{1.2}$$

$$P(X = \text{'cancer'}) = 0.1. \tag{1.3}$$

The expected utilities associated with providing treatment or not providing treatment are then easily evaluated to give

$$\langle U(d = \text{'no treatment'}) \rangle = 0.9 \times 0 + 0.1 \times (-1000) = -100 \tag{1.4}$$

$$\langle U(d = \text{'treatment'}) \rangle = 0.9 \times (-30) + 0.1 \times (-200) = -47 \tag{1.5}$$

where $\langle \cdot \rangle$ denotes an average over the random variable X. Thus the expected utility is maximized in this example by providing the cancer treatment, and the value of that expected utility is -47.

We can represent the random, decision and utility variables in this problem as a simple directed graph, as shown in Figure 1.2. The symantics of such graphs will be discussed more fully in Sec-

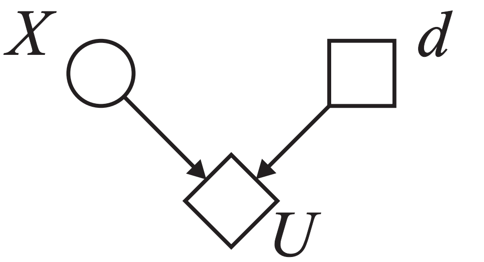

Figure 1.2: A simple decision graph corresponding to the cancer treatment problem described in the text. The circular node represents a random variable X describing whether or not the patient has cancer, the square node represents the decision variable describing whether or not the patient receives treatmtent, and the diamond node represents the utility function U. Arrows from the random and decision variables into the utility node indicate that the utility function depends on both X and d.

tion 1.2.1. We shall subsequently use such decision graphs as the basis for a graphical solution to the decision problem obtained by extending the elimination algorithm of Chapter ??.

Note that without loss of generality we can replace utility functions with cost (or loss) functions, in which the goal is to make decisions which minimize the expected cost. A cost function can always be defined as the negative of a utility, and so from now on we work exclusively with utilities.

1.1.2 Multiple Decisions

In the decision problem considered so far, there has been a single random variable X and a single decision variable d. We now turn to more general decision problems involving multiple stages of decision making which occur sequentially, together with multiple random variables. The states of some random variables may be known before the first decision is taken. After each decision the states of other random variables may be revealed and these may influence subsequent decisions. In this case there is a more complex interaction between inference and decision, and we shall see in Section 1.2 that the formalism of graphical models can be extended and exploited to provide a computationally efficient solution to the problem of making optimal decisions in such a scenario.

In order to introduce the concept of multi-stage decision problems we consider an extension of the cancer treatment example given in Section 1.1.1. We first introduce a random variable X_1, representing the presence or absence of symptoms in the patient which could be indicative of cancer. Initially the state of X_1 is revealed to the decision maker. Subsequently a decision is made, described by a decision variable d_1, as to whether to conduct a test which will provide further information on whether the patient may have cancer. There is an associated utility $U_1(d_1)$ which will have a negative value if the test is conducted corresponding to the cost of the test. After the test is conducted the state of a random variable X_2 is revealed, where X_2 represents the result of the test[1]. Subsequently a second decision is taken, represented by the decision variable d_2, as to whether to provide treatment to the patient. The final random variable X_3 describes whether the patient actually has cancer or not, and the state of this variable will be unknown at the times that decisions d_1 and d_2 must be taken. There is a second utility $U_2(X_3, d_2)$ whose value depends both on whether the patient actually has cancer and on whether treatment was administered (it might for example be given by the table in Figure 1.1). The total utility function is given by the sum $U_1 + U_2$. Note that there is a natural ordering of the random and decision variables in the form $(X_1, d_1, X_2, d_2, X_3)$ such that when decision d_i is taken only the values of random variables X_i with $i < d$ will be known. The decision graph corresponding to the extended cancer treatment example is shown in Figure 1.3.

1.1.3 The General Multi-Stage Decision Problem

We can formulate the general multi-stage decision problem as follows. Suppose we have a sequence of decision variables $d_1, d_2, \dots, d_M$ such that decision d_i is taken before decision d_{i+1}. The random variables can be partitioned into corresponding groups X_i such that after decision d_i is taken the states of the variables in group X_{i+1} are revealed. Combining the random and decision variables we have an overall decision sequence of the form $(X_1, d_1, X_2, d_2, \dots, d_M, X_{M+1})$. Note that the decisions d_i can also comprise groups of variables, although for simplicity we shall limit attention to the case of a single variable d_i (the generalization to groups of decision variables is straightforward).

[1] Note that if the decision is made not to conduct the test, then the result of the test will of course not be revealed. We return to a discussion of this asymmetry in Section 1.2.6

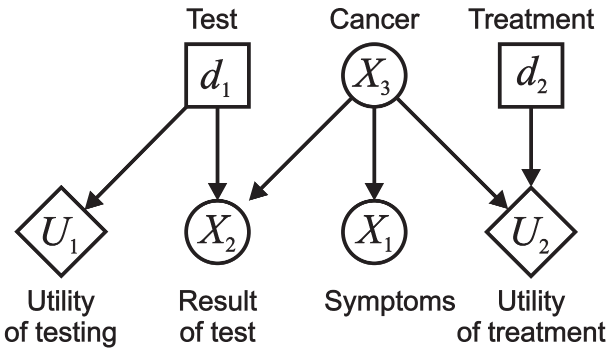

Figure 1.3: Decision graph corresponding to the extended cancer treatment example discussed in the text.

Without loss of generality we can exploit the ordering of the random variables to specify their joint distribution as a factorization in terms of conditional distributions of the form

$$P(X_{i+1}|X_1, \ldots, X_i, d_1, \ldots, d_M) \tag{1.6}$$

where $i = 1, \ldots, M$.

However, we must also impose a constraint to reflect the causality requirement that the value of a decision variable cannot influence any of the random variables which appear earlier in the decision sequence. This is equivalent to a set of conditional independence statements of the form

$$P(X_{i+1}|X_1, \ldots, X_i, d_1, \ldots, d_M) = P(X_{i+1}|X_1, \ldots, X_i, d_1, \ldots, d_i). \tag{1.7}$$

Any consistent model of a multi-stage decision problem must respect these constraints.

Thus the probabilistic model is specified by defining the following set of conditional probabilities

$$P(X_1) \tag{1.8}$$

$$P(X_2|X_1, d_1)$$

$$\cdots \tag{1.9}$$

$$P(X_{M+1}|X_1, d_1, \ldots, X_M, d_M). \tag{1.10}$$

Although these represent the most general situation, in any particular model there may be variables missing from the conditioning sets in (1.8) – (1.10) corresponding to additional conditional independence properties. When we develop the extended elimination algorithm for decision graphs in Section 1.2 we will seek to exploit these conditional independences in order to obtain efficient computation.

The decision problem specification is completed by defining the utility function, which in general can depend on all of the random and decision variables

$$U(X, d) = U(X_1, d_1, \ldots, X_M, d_M, X_{M+1}). \tag{1.11}$$

This utility function often takes the form of the sum of a number of terms, each of which depends only on a subset of the variables. Again, we seek an algorithm which can exploit this additive structure to achieve efficient computation.

1.1.4 Solution of the Multi-stage Decision Problem

Now let us consider how to solve the multi-stage decision problem in the general case. The solution will take the form of a set of tables which specify the optimal decision $\widehat{d}_i$ for all possible settings of the previous decision variables (which may have been chosen suboptimally) and of the random variables whose values will be known at the time that d_i must be chosen. Thus we seek a set of tables, called a decision *mapping*, of the form

$$\widehat{d}_1(X_1) \tag{1.12}$$

$$\widehat{d}_2(X_1, d_1, X_2) \tag{1.13}$$

$$\cdots$$

$$\widehat{d}_M(X_1, d_1, X_2, \ldots, d_{M-1}, X_M). \tag{1.14}$$

Given this set of tables any particular decision can be taken optimally since the required values of previous decision variables and random variables will be known at the time the decision must be taken.

The overall optimal decision is one in which the complete set of decision variables is chosen jointly to maximize the expected utility. To do this the first decision is made optimally using (1.12), and then this optimal decision $\widehat{d}_1$ is used to find the optimal value for the second decision variable using (1.13) and so on. Thus the overall optimum can be found from the decision mapping by substitution to give

$$\widehat{d}_1(X_1) \tag{1.15}$$

$$\widehat{d}_2(X_1, \widehat{d}_1, X_2) \tag{1.16}$$

$$\cdots$$

$$\widehat{d}_M(X_1, \widehat{d}_1, X_2, \ldots, \widehat{d}_{M-1}, X_M). \tag{1.17}$$

Now consider the problem of computing one of the tables $\widehat{d}_i(X_1, d_1, \ldots, d_{i-1}, X_i)$ in the decision mapping (1.12) – (1.14). For each setting of the known variables $(X_1, d_1, \ldots, d_{i-1}, X_i)$ we must determine the setting of the decision variable d_i such that the overall expected utility is maximized. This expected utility will involve averages over the as yet unobserved random variables $X_{i+1}, \ldots, X_{M+1}$ but will also depend on the choices made for future decisions which themselves must be taken optimally. Thus to find $\widehat{d}_i(\cdots)$ we must in general consider all possible combinations of settings of all of the decision variables, and indeed this turns out to be an NP-hard problem.

We can evaluate the decision mapping tables by a backwards induction process starting with the final decision. Thus we can first determine $\widehat{d}_M$ as a function of $(X_1, d_1, \ldots, X_M)$ by first marginalizing the utility function over X_{M+1} to define an expected utility function given by

$$\overline{U}_M(X_1, d_1, \ldots, X_M, d_M) = \sum_{X_{M+1}} P(X_{M+1}|X_1, d_1, \ldots, X_M, d_M) U(X_1, d_1, \ldots, X_M, d_M, X_{M+1}) \tag{1.18}$$

and then maximizing with respect to d_M for all possible choices of $(X_1, d_1, \ldots, d_{M-1}, X_M)$

$$\widehat{d}_M(X_1, d_1, \ldots, X_M) = \arg\max_{d_M} \overline{U}_M(X_1, d_1, \ldots, X_M, d_M). \tag{1.19}$$

Substituting $\widehat{d}_M$ into the expected utility function $\overline{U}_M$ yields a new utility function in which d_M and X_{M+1} have been eliminated

$$U_{M-1}(X_1, d_1, \ldots, X_M) = \overline{U}_M(X_1, d_1, \ldots, X_M, \widehat{d}_M(X_1, d_1, \ldots, X_M)) \tag{1.20}$$

and corresponds to a multi-stage decision problem over a reduced set of variables. Thus the above procedure can be applied recursively until the complete set of decision mapping tables has been evaluated.

The full solution to the decision problem to find the jointly optimal set of decisions is therefore given by

$$\sum_{X_1} \max_{d_1} \sum_{X_2} \cdots \max_{d_M} \sum_{X_{M+1}} P(X|d) U(X, d) \tag{1.21}$$

where we have defined

$$P(X|d) = P(X_{M+1}|X_1, \ldots, d_M) \cdots P(X_2|X_1, d_1) P(X_1). \tag{1.22}$$

Note that the quantity $P(X|d)$ can only be interpreted as the conditional distribution of X conditioned on d if the values of d are set externally. If the values of d are chosen to maximize expected utility then they will depend on the values of X and so in this case $P(X|d)$ can no longer be viewed as the conditional distribution.

Our goal is to evaluate the successive summations and maximizations given by (1.21). The values which are assigned to the decision variables will then yield the decision mapping corresponding to (1.12) – (1.14), and the final value of the quantity in (1.21) gives the corresponding maximum value of the expected utility. Our approach will generalize the elimination algorithm of Chapter ?? and will be based on the idea of commuting the summation and maximization operations through the various factors in $P(X|d)U(X, d)$ in order to achieve efficient computations which act on small groups of variables at a time. Before developing this framework, however, we consider a brute force approach, called *decision trees*, which emphasises the nature of the computations which must be performed.

1.1.5 Decision Trees

In this section we introduce a simple graphical technique, called *decision trees*, for setting out the solution to a multi-stage decision problem. Although for many complex problems decision trees can be computationally inefficient compared to the elimination algorithm developed shortly, they serve a useful role in emphasising the underlying computations which must be performed.

Consider again the extended cancer treatment problem discussed in Section 1.1.2. The decision tree is simply a graphical representation of all possible combinations of choices for the decision variables and the random variables. In the cancer problem the first variable to be revealed is the random variable X_1 corresponding to the presence or absence of symptoms. Subsequently a decision d_1 is taken as to whether a test should be conducted. As a consequence the result X_2 of the test is revealed and then a decision d_2 can be made as to whether to provide treatment. The final random variable X_3 describes whether the patient has cancer or not and is only known after both decisions have been taken. The full decision tree corresponding to this problem involves many possible paths through the states of these variables. Two such paths are shown in Figure 1.4. Note

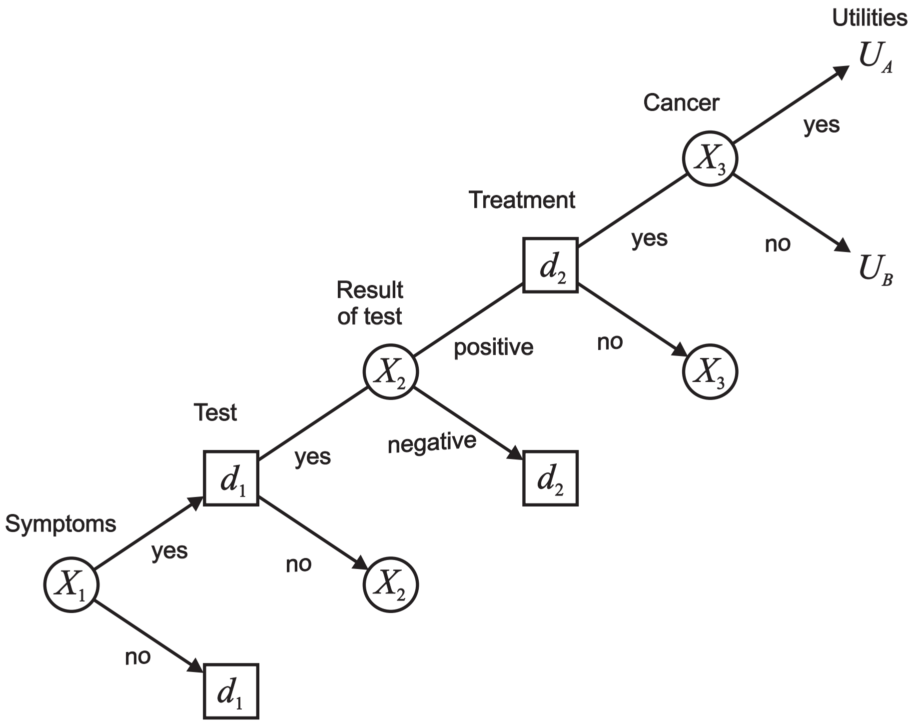

Figure 1.4: Decision tree for the extended cancer treatment example discussed in Section 1.1.2. For clarity only two of the possible paths from the root node to the utilities are shown.

that if decision d_1 is set so that no test is conducted then there will be no test result to observe. This asymmetry, which is exploited efficiently in decision trees, is discussed further in Section 1.2.5.

By following each path in the decision tree from the root to a leaf we can evaluate the corresponding utility at the leaf. It is then a simple matter to find, for any given decision node in the tree, the decision which will give rise to the maximum expected utility. Consider for instance the decision node d_2 in Figure 1.4 corresponding to the decision on whether to administer treatment in the case of a positive test result. The expected utility for the choice 'yes' for d_2 is obtained by averaging the two utilities U_A and U_B, weighted by the appropriate conditional probabilities for the states of X_3. Thus the process of solving the decision problem using the decision tree framework involves a backwards recursion starting with the final state utilities. So in Figure 1.4, we first evaluate the marginal utilities at each of the 'X_3' nodes by computing sums of utilities weighted by their corresponding probabilities (conditioned on the previous variables in the path). Next, for each decision node 'd_2' we chose the maximum expected utility. These expected utilities are then fed back to the X_2 nodes where again they are summed with weights given by the corresponding conditional probabilities, to give an expected utility at each X_2 node. The optimal decision for the d_1 node is found by choosing the maximum of the expected utilities at the X_2 nodes. Finally, the value of the maximum expected utility is obtained by summing over the X_1 variable weighted by its marginal probability.

Note that the conditional probabilities required to evaluate this decision tree are given by

$$P(X_3|X_1, X_2, d_1, d_2)$$

$$P(X_2|X_1, d_1)$$

$$P(X_1)$$

whereas we see from Figure 1.3 that the original problem is formulated most naturally using conditional probabilities in the form

$$P(X_3)$$

$$P(X_1|X_3)$$

$$P(X_2|X_3, d_1).$$

In order to apply the decision tree technique to this problem we must therefore perform some initial pre-processing using the sum and product rules of probability in order to recast the conditional probabilities in the correct form. The requirement for such pre-processing is a general feature of decision trees. As we shall see, the required manipulations are performed automatically when we make use of the elimination algorithm, and so the need to pre-process the conditional probabilities is avoided.

The principal disadvantage of decision trees, however, is that they exhaustively enumerate every possible path and so fail to take advantage of conditional independences which might exist in the model. This is resolved by moving to an elimination tree approach, discussed in Section 1.2.

1.2 Elimination Algorithm for Decision Graphs

We turn now to the development of a general framework for solving multi-stage decision problems in a computationally efficient manner. Our approach will be based on an extension of the elimination algorithm introduced in Chapter ??. Note that an alternative approach would be to extend the junction tree algorithm of Chapter ??. Although a junction tree approach may often be more efficient in terms of storage, as we discuss in Section 1.2.5, the algorithm iteself is more complex as it required additional book keeping. For simplicity we therefore limit attention to the elimination tree approach.

In Chapter ?? we viewed the elimination algorithm as an efficient procedure for evaluating conditional probabilities obtained by commuting summation (marginalization) operators with the factors comprising the joint probability distribution, so that the summations act only on small subsets of the variables. For the case of multi-stage decision problems our goal is the evaluation of the sequence of summations and maximizations defined by (1.21) in which the joint distribution $P(X|d)$ is defined by (1.22). To achieve this efficiently we again seek to commute the factors in the joint distribution through the summation and maximization operators so that the operations are performed over sets of low cardinality. The utility function itself may also comprise the sum of several terms each of which only depends on a subset of the random and decision variables, and again we wish to exploit such structure to achieve computational efficiency.

In the case of purely probabilistic models we were also able to choose an efficient order in which to marginalize, and therefore eliminate, the variables. However, a key point to emphasise in the case of decision problems is that the elimination ordering is restricted because it is not possible to permute a summation with a maximization. To see this consider the cancer example described in Section 1.1.1 whose utility function is shown in Figure 1.1. The value of the maximum expected utiliy was obtained by first averaging over the two states of the random variable X for each possible value of the decision variable d (in order to obtain the expected utility as a function of d), and then subsequently choosing the value of d for which the expected utility was greater. This gave an expected utility of -47. Now suppose that instead we perform the maximization first, and then subsequently marginalize over X. For $X =$ 'normal' the maximum value of the utility $U(X, d)$ occurs for $d =$ 'no treatment' and has the value 0, while for $X =$ 'cancer' the maximum value of the utility occurs for $d =$ 'treatment' and has the value -200. Thus the expected value of this maximum utility is given by

$$\langle U \rangle = 0.9 \times 0 + 0.1 \times (-200) = -20. \tag{1.23}$$

Thus the value of the maximum expected utility is different in the two cases, and also in the second case the optimal decision on whether to provide treatment is dependent on the state of the variable X. Making the decision before averaging over X corresponds to the case where the state of health of the patient is known before the decision to treat is taken. This of course leads to a higher expected utility since the decision is better informed.

Thus we see that summations and maximizations cannot be commuted. We are still free to choose the elimination ordering of the variables within each group of random variables X_i (and similarly within each of the d_i if these comprise more than one decision variable) but we cannot interchange the groups.

1.2.1 Definition of Decision Graphs

In order to extend the elimination algorithm to the case of decision problems we first of all generalize the concept of a directed probabilistic graph to include decision variables and utility functions, and we shall refer to these as *decision graphs*. We have encountered two examples of such graphs already in Figures 1.2 and 1.3.

We can define a decision graph in general as follows. Consider first a problem involving only random variables. The joint distribution over these variables can be characterized by a directed acyclic graph, as discussed in Chapter ??. Now suppose that in addition we have some decision variables d_i. These enter as conditioning variables in the conditional probabilities (1.6). We therefore introduce an additional node (represented as a square) for each decision variable d_i and this will be a parent of every random variable node for which d_i appears in the conditioning set of the corresponding conditional distribution. Thus decision nodes themselves have no parents and have random nodes as their children.

Next we wish to represent the utility functions. For each utility term $U_m(X, d)$ we introduce a corresponding node (represented as a diamond) whose parents are the set of random variables and decision variables on which that term depends. Thus utility nodes can have both random and decision variables as parents, but themselves have no children.

There is one final constraint on the structure of the graph, which arises from the causality requirement represented by the conditional independence conditions (1.7). Specifically we require that, ignoring the utility nodes, for each decision node, all random variables which are descendents of that decision node must appear later in the decision sequence than that decqision variable. To see that this implies the conditions (1.7) consider the joint distribution $P(X|d)$ formed by taking the product of the individual conditionals given by (1.6). For any given decision variable d_m we can marginalize the joint distribution with respect to the descendents of d_m. Now from Chapter ?? we know that if we marginalize over a random node that has no descendents, the joint distribution of the remaining variables factorizes with resepect to the graph obtained by removing that node. We can therefore work backwards from the leaf nodes and remove nodes one at a time from the graph. Once we have removed all of the descendents of d_m, the node corresponding to d_m will be isolated and so the joint distribution of the remaining variables will be independent of d_m. It is easy to see that this implies that all of the factors which comprise that joint distribution will also be independent of d_m, and so we have shown that the conditions (1.7) must hold.

Note that there is a closely related graphical representation of multi-stage decision problems called *influence diagrams*. They differ from the directed decision graphs discussed here through the addition of extra edges, called *information arcs*, leading into decision nodes which reflect the causal consistency constraint. Specifically, for any decision node, the ancestors of the node in the direced graph correspond to the variables (decision and random) whose values will be known at the time the decision is taken. The causality requirement then translates into a requirement that the directed graph be acyclic. Here we consider graphs without information arcs, but we must then maintain separately a record of the partial ordering of the variables in the problem.

1.2.2 Elimination for Decision Graphs

In order to construct the computational machinery needed to solve the decision problem, we will generalize the elimination algorithm introduced in Chapter ??. We will show how the sequential elimination of variables allows us to define a tree of eliminations sets satisfying the running intersection property, and we will then generalize the concept of the clique potentials, introduced in Chapter ??, to include utilities. As we shall see, it is appropriate to use potentials consisting of *pairs* of functions, with one member of each pair representing a probabilistic component and the other member representing a utility component.

In order to motivate the use of function pairs for potentials, consider the graph shown in Figure 1.5. This graph contains no decision nodes, only random and utility nodes, and the goal is to

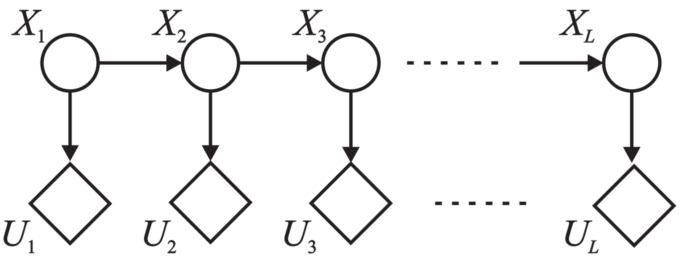

Figure 1.5: A graph having only random and utility nodes, for which the goal is to evaluate the expected utility. This example motivates the use of potentials containing pairs of functions representing the probability and utility contributions separately.

evaluate the total expected utility. This requires that we evaluate the following expression involving summation over the random variables

$$\sum_{X_1}\sum_{X_2}\dots\sum_{X_L} P(X_1)P(X_2|X_1)\dots P(X_L|X_{L-1})\left\{U_1(X_1)+U_2(X_2)+\dots+U_L(X_L)\right\}. \qquad (1.24)$$

Clearly a very bad way to solve this problem would be to evaluate the function

$$P(X_1)P(X_2|X_1)\dots P(X_L|X_{L-1})\left\{U_1(X_1)+U_2(X_2)+\dots+U_L(X_L)\right\} \tag{1.25}$$

for all combinations of variable values and then perform the L successive summations. If each of the random variables has K states then this would require evaluation and storage of a function of the joint space of all L variables, comprising K^L states, and subsequent summation over this number of states.

Another approach would be to expand out the bracket and solve L separate problems, one for each term of the utility function, and then add the results together. This again is inefficient since many of the summation operations would be common to the L sub-problems and hence there would be substantial replication of computation.

We can find a more efficient approach by exploiting the conditional independence properties of the probabilistic 'backbone' in the graph in Figure 1.5, together with the additive decomposition of the utility function. Here we consider how to set up such a computation in which we eliminate (i.e. sum over) X_1 first[2], then X_2 and so on up to X_L. If we commute the conditional probabilities through the summations as far as we can we obtain the following expression for the expected utility

$$\sum_{X_L}\dots\sum_{X_2}P(X_3|X_2)\sum_{X_1}P(X_1)P(X_2|X_1)\left\{U_1(X_1)+U_2(X_2)+\dots+U_L(X_L)\right\}. \tag{1.26}$$

Now we exploit the additive structure of the utility function. Consider first the summation over X_1, and define the following quantities

$$\gamma_1^*(X_2) = \sum_{X_1}P(X_1)P(X_2|X_1) \tag{1.27}$$

$$\eta_1^*(X_2) = \sum_{X_1}P(X_1)P(X_2|X_1)U_1(X_1) \tag{1.28}$$

where the $*$ superscript denotes that a variable has been marginalized out. We shall interpret these quantities shortly as messages passed between cliques of the elimination tree. Using (1.27) and (1.28) we can write (1.26) in the form

$$\sum_{X_L}\dots\sum_{X_2}P(X_3|X_2)\left\{\eta_1^*(X_2)+\gamma_1^*(X_2)\left[U_2(X_2)+\dots+U_L(X_L)\right]\right\}. \tag{1.29}$$

We have therefore succeeded in eliminating the variable X_1. Note that this operation required the evaluation of *two* different summations over X_1 given by (1.27) and (1.28). These quantities must be computed for all values of the variable X_2. However, because of the conditional independence property $X_3 \perp\!\!\!\perp X_1 \mid X_2$, corresponding to the absence of an edge from X_1 to X_3 in the graph in Figure 1.5, we do not need to involve the variable X_3 (or indeed any of the remaining variables) at this stage, so the summations involve storage and computation only over the variables (X_1, X_2).

For reasons which we shall discuss shortly, it is convenient to work with normalized expected

[2]Since this problem involves only random variables and no decision variables, we can consider any elimination order we wish. Other elimination orderings are considered in the exercises.

utilities. We therefore define a new quantity given by

$$\mu_1^*(X_2) = \frac{\eta_1^*(X_2)}{\gamma_1^*(X_2)}. \tag{1.30}$$

This allows us to pull out a common factor of $\gamma_1^*(X_2)$ from the braces in (1.29) go give

$$\sum_{X_L} \dots \sum_{X_2} P(X_3|X_2)\gamma_1^*(X_2) \left\{\mu_1^*(X_2) + U_2(X_2) + \dots + U_L(X_L)\right\}. \tag{1.31}$$

The key point to emphasize here is that we have to maintain separately two quantities resulting from summations over X_1. The first of these, $\gamma_1^*(X_2)$, involves only conditional probabilities, while the second, $\mu_1^*(X_2)$, takes the form of an expected utility.

When we come to eliminate variable X_2 we can work in the joint space (X_2, X_3). From (1.31) we see that this requires us to have access to the quantities $\gamma_1^*(X_2)$ and $\mu_1^*(X_2)$ separately, otherwise we would be forced to work in the larger joint space (X_1, X_2, X_3).

We can continue eliminating the variables one at a time, working our way along the chain and exploiting its conditional independence properties. The elimination of the mth variable involves a summation given by

$$\sum_{X_L} \dots \sum_{X_m} P(X_{m+1}|X_m)\gamma_{m-1}^*(X_m) \left\{\mu_{m-1}^*(X_m) + U_m(X_m) + \dots + U_L(X_L)\right\}. \tag{1.32}$$

We now define the quantities

$$\widehat{\gamma}_m(X_{m+1}, X_m) = P(X_{m+1}|X_m)\gamma_{m-1}^*(X_m) \tag{1.33}$$

$$\widehat{\mu}_m(X_m) = \mu_{m-1}^*(X_m) + U_m(X_m) \tag{1.34}$$

which we shall interpret in Section 1.2.4 as the two components of a clique potential, modified by absorption of the messages γ_{m-1}^* and μ_{m-1}^*. The total expected utility, after summing out variables X_1 to X_{m-1}, then becomes

$$\sum_{X_L} \dots \sum_{X_m} \widehat{\gamma}_m(X_{m+1}, X_m) \left\{\widehat{\mu}_m(X_m) + U_{m+1}(X_{m+1}) + \dots + U_M(X_M)\right\}. \tag{1.35}$$

Again there are two distinct summations over X_m to be performed, giving rise to the two quantities

$$\gamma_m^*(X_{m+1}) = \sum_{X_m} \gamma_m(X_{m+1}, X_m) \tag{1.36}$$

$$\mu_m^*(X_{m+1}) = \frac{\sum_{X_m} \mu_m(X_m)\gamma_m(X_{m+1}, X_m)}{\gamma_m^*(X_{m+1})} \tag{1.37}$$

which, as we shall see in Section 1.2.5, are the messages transmitted to the next clique in the elimination tree when the variable X_m is eliminated. Thus we see that, in order to maintain computational efficiency, two types of quantities (and, as we shall see, only two) need to be maintained and propagated.

Note that if there were only probability nodes and no utility (or decision) nodes, then we would only need to propagate γ variables, as we have seen in previous chapters. They are updated and marginalized using the standard elimination algorithm for probabilistic models as discussed in Chapter ??. The computational saving arising from the use of pairs of functions in the potentials stems from the additive nature of the utility function.

1.2.3 Elimination Trees

In Chapter ?? we gave an informal introduction to the elimination algorithm in the context of directed graphs of random nodes. Recall that when we eliminate a node from a graph we connect together all of the neighbours of that node. Here we revisit the elimination algorithm and show how it can be used to construct a particular form of junction tree called an *elimination tree*. We then extend this procedure to general decision graphs.

Consider first a graph consisting entirely of random nodes. As we have already seen in Chapter ?, the first step is to moralize the graph in order that each of the conditional probabilities can be associated with at least one of the cliques of the resulting elimination tree. After moralization the graph can be triangulated.

Here we define a particular algorithm for triangulation, called *one step look-ahead*, which is an alternative to the maximum cardinality search considered in Chapter ??, and we then use this to construct an elimination tree.

Algorithm 1.1 (One step look-ahead triangulation) *Consider a (directed or undirected) graph $\mathcal{G}$ with L vertices. Set the variable $i = L$, and let all the vertices initially be unnumbered. While there are still unnumbered vertices in the graph:*

1. *Choose an unnumbered vertex having the smallest number of unnumbered neighbours, and give it the label i.*
2. *Define the set C_i to consist of the vertex i together with all of its unnumbered neighbours.*
3. *Add edges to the graph as required such that the set C_i becomes fully connected.*
4. *Eliminate the vertex i from the graph and decrement i by 1.*

At each step of this algorithm the variable to be eliminated is chosen so as to give rise to the smallest elimination set C_i. As we shall see, the corresponding computation involves the joint space of the variables in the clique and so it is here that the elimination approach for decision graphs achieves its computational efficiency. This is, however, a greedy algorithm so is not guaranteed to find the optimal triangulation (recall from Chapter ?? that triangulation is NP-hard).

Next we treat the sets C_i as nodes in a graph, and add edges between pairs of nodes so as to form a tree. This is achieved using the following procedure

Algorithm 1.2 (Elimination tree construction) *For $i = 1, \ldots, L$, if C_i contains more than one vertex, add an edge between C_i and C_j, where j is the largest index of the vertices in C_i (excluding vertex i itself). Finally, add an extra node, containing no vertices, and connect it to all nodes having a single vertex.*

This procedure ensures that when we eliminate a variable, the remaining variables in that elimination set are connected to a node in which those variables also appear. A consequence of this is that the elimination tree will satisfy the running intersection property. To see this, note that for each variable there will be a particular node at which it is eliminated, and that it will not appear in any node which is nearer to the root than this node. If there is any other node in which the same variable appears then, since it will not be eliminated in that node, it must appear in the next node in the tree which is nearer the root. By induction the same variable must appear in every node on the path which connects any two nodes in which the variable appears, and hence the tree has the running intersection property. This is therefore a junction tree of sets of nodes. Note, however, although the cliques will appear as nodes of the tree, there will in general also be other nodes comprising subsets of nodes in cliques. Thus the elimination is not a junction tree of cliques.

As an illustration of this procedure, consider the directed graph of Figure ?? in Chapter ??, reproduced for convenience in Figure 1.6. Applying the triangulation algorithm 1.1 followed by

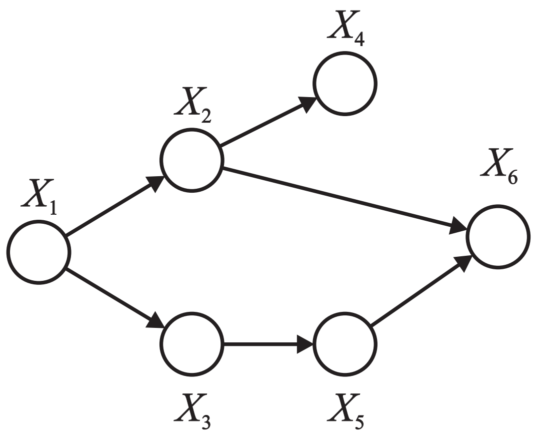

Figure 1.6: The graph introduced in Chapter ?? to illustrate the technique of elimination.

algorithm 1.2 we obtain the elimination tree shown in Figure 1.7. Note that are several possible

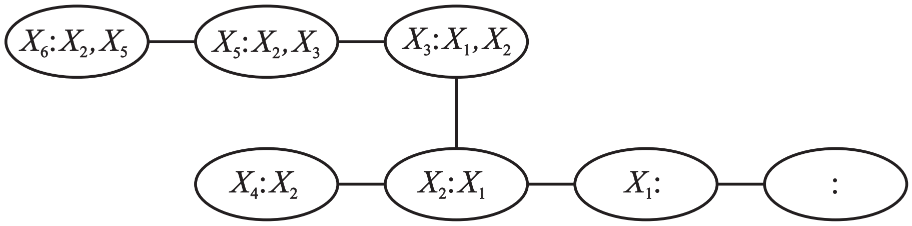

Figure 1.7: The elimination tree resulting from the application of the elimination algorithm 1.1 to the graph in Figure 1.6 followed by the tree construction procedure given by algorithm 1.2. The notation $X : Y, \ldots , Z$ associated with a node denotes that the variable X is eliminated in that node, and that the remaining variables in the node are $Y, \ldots , Z$ It is easily seen that this tree satisfies the running intersection property.

elimination orderings which are consistent with the algorithm 1.1. We choose an ordering which is the same as that considered in Chapter ??.

In order to extend the construction of an elimination graph to the framework of decision graphs we need to take account of both utility nodes and decision nodes. Each utility node is associated with an additive term in the total utility function. Recall that utility nodes have only parents and no descendents. The moralization step will add links between all pairs of parents for each utility node. After the moralization step the utility nodes can be deleted from the graph and each term in the utility function can then be associated with the corresonding complete set of nodes on the moral graph. As we saw in Section 1.1.4, the presence of decision variables places restrictions on the order in which variables may be eliminated. This can be accommodated simply by altering the triangulation algorithm 1.1 so as to consider only those elimination orderings which are consistent with the causality constraints of the particular problem being solved. More formally we have the following algorithm:

Algorithm 1.3 (One step look-ahead triangulation for decision graphs) *Consider a moralized decision graph $\mathcal{G}$ with the decision sequence $(X_1, d_1, X_2, d_2, \ldots, d_M, X_{M+1})$ having a total of L nodes after the utility nodes have been removed. Set the variable $i = L$, set $j = M + 1$, and let all the vertices initially be unnumbered. While there are still unnumbered vertices in the graph:*

1. *Choose an unnumbered vertex from the set X_j having the smallest number of unnumbered neighbours, and give it the label i. If there are no such vertices then select decision variable d_j, give it the label i, and decrement j by 1.*
2. *Define the set C_i to consist of the vertex i together with all of its unnumbered neighbours.*
3. *Add edges to the graph as required such that the set C_i becomes fully connected.*
4. *Eliminate the vertex i from the graph and decrement i by 1.*

Again we use algorithm 1.2 to construct an elimination tree of cliques, with the empty clique designated as the root.

1.2.4 Decision Potentials

We are now in a position to set up the formalism for solving decision problems through the use of elimination trees. This will involve defining decision potentials, initialization of the tree, and then message passing on the tree to determine the optimal decisions and the maximum value of the expected utility.

As we have already seen, the efficient solution of problems involving decision graphs requires that we keep separate track of two different quantities, one related to probabilities and the other to expected utilities. We therefore define a *decision potential* ϕ to consist of a pair of variables (γ, μ) where the first variable γ represents a probability and the second variable μ represents an expected utility. For problems in which there are no decision or utility nodes, the formalism will reduce to an exact inference algorithm for probabilistic graphs.

We now define the three key operations on decision potentials:

Multiplication Given two potentials $\phi_1 = (\gamma_1, \mu_1)$ and $\phi_2 = (\gamma_2, \mu_2)$ we define their product to be the potential

$$\phi_1 \phi_2 = (\gamma_1 \gamma_2, \mu_1 + \mu_2) \tag{1.38}$$

so that the probability parts combine multiplicatively while the utility parts are additive.

Sum-marginalization We define the sum-marginalization over a random variable X as

$$\mathbf{S}_X \phi = \left(\sum_X \gamma, \frac{\sum_X \gamma\mu}{\sum_X \gamma} \right) \tag{1.39}$$

Max-marginalization We define the max marginalization over a decision variable d as

$$\mathbf{M}_d \phi = \left(\gamma(d = \widehat{d}), \mu(d = \widehat{d}) \right) \tag{1.40}$$

where

$$\widehat{d} = \arg\max_{d}(\gamma\mu). \tag{1.41}$$

1.2.5 Initialization and Message Passing for the Elimination Tree

In order to construct an elimination tree for solving the decision problem we take the original directed decision graph and then moralize it by adding edges connecting parents of nodes and then dropping the directions on the edges. This ensures that there will exist a clique on the undirected graph corresponding to each of the conditional probabilities $P(X_i|X_1, d_1, \dots)$ and similarly a clique for each utility function U_m. Once this has been done the nodes corresponding to utility functions can be removed.

We now initialize the elimination tree as follows. For every clique C_i we define a decision potential $\phi_i = (\gamma_i, \mu_i)$, with the probability term γ_i initialized to unity and the utility term μ_i initialized to zero. Then, for each factor $P(X_i|X_1, d_1, \dots)$ in the joint probability $P(X|d)$ we identify a clique containing all of the variables in this factor and we multiply the probability term in that clique potential by $P(X_i|X_1, d_1, \dots)$. When this is done, the product over all decision potentials will have a probability term which is the joint probability $P(X|d)$.

Then for each utilty function U_m we identify a clique containing the arguments of U_m and then add U_m to the utility term in that clique's decision potential. When this is done the sum of all the utility terms in all of the decision potentials will equal the total utility function. Thus, from the definition of the product of decision potentials, given by (1.38), the product of all the decision potentials, given by

$$\phi_V = \prod_i \phi_i \tag{1.42}$$

will have a utility term which is equal to the total utility function. Note that there is no need to introduce separator sets and separator potentials since, as we shall see, the solution to the decision problem involves only one-way message passing (equivalent to a 'CollectEvidence' operation, with no corresponding 'DistributeEvidence').

Next we define a message passing step whereby a variable is eliminated either by summation (probability variable) or maximization (decision variable). Suppose C is an elimination set of an elimination tree, and let $C_1, \dots, C_M$ be the neighbours of C which are further from the root than C itself. Then the action of absorption consists of the evaluation of messages ϕ^*_m for each of the sets C_m and the absorbtion of these messages by multiplication into the decision potential ϕ_C for clique C. Specifically

$$\phi^*_m = \begin{cases} \mathbf{M}_d\phi_m & \text{if } C_m \text{ is the elimination set of a decision variable } d \\ \mathbf{S}_X\phi_m & \text{if } C_m \text{ is the elimination set of a random variable } X \end{cases} \tag{1.43}$$

and

$$\widehat{\phi}_C = \phi_C \phi^*_1 \dots \phi^*_M. \tag{1.44}$$

This absorbtion operation then forms the basic step of 'CollectEvidence' which is defined recursively. When CollectEvidence is called at a particular node it first calls CollectEvidence in any neighbours which are further from the root, and subsequently absorbs messages from them. The solution of the decision problem then involves a call to CollectEvidence at the root node of the elimination tree. This leads to a sequence of absorptions starting at the leaves of the tree and working inwards towards the root. Once this has terminated the root clique potential will contain the maximum expected utility, and the optimal values for all of the decision variables will have been determined during the various max-marignalizations. We give a formal proof of this result in Section 1.2.6.

One issue which is worth exploring in more detail concerns the definition of sum marginalization given by (1.39) which involved normalization of the expected utilities, and hence which requires divisions to be performed. In the general formulation of the decision problem given by (1.21) there are no division operations, so it worth looking more closely at the motivation for this definition. To do this we consider the elimination tree shown in Figure 1.8, in which the elimination ordering is $(X_1, \dots, X_L, Z)$. This graph could be considered as part of a larger elimination tree.

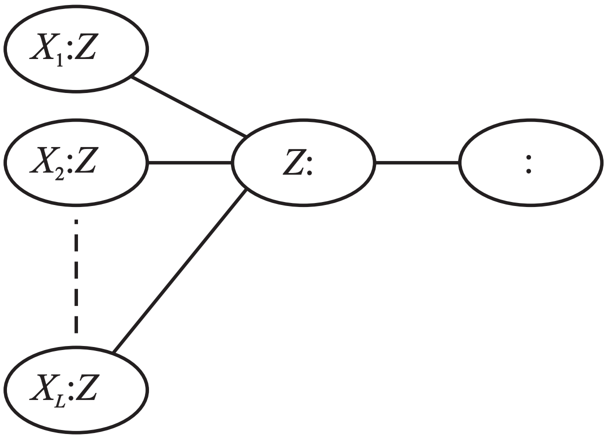

Figure 1.8: An elimination tree used to motivate the use of normalized expected utilities in the definition of message passing.

Note that in this instance there are only random variables and no decision variables. The key point is that several different messages are to be absorbed into the same node Z. Associated with each node of the tree we have initial potentials $\phi_i(X_i, Z) = (\gamma_i(X_i, Z), \mu_i(X_i, Z))$ for $i = 1, \dots, L$ and $\phi_Z(Z) = (\gamma_Z(Z), \mu_Z(Z))$. Marginalization over the random variables to yield the total expected utility corresponds to the following operations

$$\langle U \rangle = \sum_Z \sum_{X_1} \dots \sum_{X_L} \gamma_Z(Z)\gamma_1(X_1, Z)\dots\gamma_L(X_L, Z)\left\{\mu_1(X_1, Z) + \dots + \mu_L(X_L, Z) + \mu_Z(X_Z)\right\}. \tag{1.45}$$

In the approach discussed here we seek a message passing scheme such that we can absorb from the nodes sequentially while preserving all the required information using only the two local potentials γ and μ in each node of the elimination tree. To see how this is achieved, suppose we first eliminate variable X_L from (1.45). We require that the result of this be a similar elimination tree with the node $(X_L : Z)$ removed and with suitably modified potentials $\widehat{\gamma}_Z(Z)$ and $\widehat{\mu}_Z(Z)$ in node Z so that

$$\langle U \rangle = \sum_Z \sum_{X_1} \dots \sum_{X_{L-1}} \widehat{\gamma}_Z(Z)\gamma_1(X_1, Z)\dots\gamma_{L-1}(X_{L-1}, Z)\left\{\mu_1(X_1, Z) + \dots + \mu_{L-1}(X_{L-1}, Z) + \widehat{\mu}_Z(Z)\right\}. \tag{1.46}$$

By identifying corresponding terms in (1.45) and (1.46) we obtain

$$\widehat{\gamma}_Z(Z) = \gamma_Z(Z)\sum_{X_L}\gamma_L(X_L,Z) \tag{1.47}$$

$$\widehat{\mu}_Z(X_Z) = \mu_Z(Z) + \frac{\sum_{X_L}\gamma_L(X_L,Z)\mu_L(X_L,Z)}{\sum_{X_L}\gamma_L(X_L,Z)} \tag{1.48}$$

which corresponds to the definition of sum marginalization introduced in Section 1.2.

Note that we could avoid divisions by keeping track of all the incoming messages to a node, at the expense of extra storage. This would be analogous to the Shafer-Shenoy algorithm discussed in Chapter ??.

Because the elimination tree approach involves eliminating one variable at a time, the number of cliques in the tree (excluding the empty clique) will be equal to the total number of random and decision variables. Instead of using an elimination tree, we could instead have extended the junction tree framework of Chapter ?? to allow for decision variables and utility nodes. Although this can be more efficient in terms of storage, it requires additional book keeping since to construct the message from a given clique it may have to marginalize out several random and decision variables and hence must keep track of the ordering of those variables. Note that the junction tree approach would require only a 'CollectEvidence' step. Since no 'DistributeEvidence' is needed, the separator potentials play no role and can be omitted.

We saw in the case of the extended cancer example of Section 1.1.2 that there was an asymmetry in that if a decision d_1 was made not to carry out the test, then the random variable representing the result X_2 of the test would remain unobserved. Such issues can easily be handled within the elimination algorithm framework by simply expanding the state space of X_2 to include a 'no result' state. The disadvantage, however, is that the expanded state space leads to additional computation, a problem which is avoided in the decision tree approach.

1.2.6 Summary and Proof of Elimination Algorithm

So far in this chapter we have assembled and motivated all of the ingredients needed for the formulation of an elimination algorithm to solve the multi-stage decision problem. We now summarize the steps of the algorithm, and then we give a formal proof that it does indeed generate the correct solution.

Algorithm 1.4 (Elimination Algorithm for Multi-stage Decision Problems)

Consider a multi-stage decision problem represented by the decision sequence

$$(X_1, d_1, X_2, d_2, \ldots, d_M, X_{M+1}) \tag{1.49}$$

with corresponding sets of conditional probabilities

$$P(X_i|X_1,\ldots,X_{i-1},d_1,\ldots,d_{i-1}). \tag{1.50}$$

The algorithm consists of the following steps:

1. *Construct the decision graph using the formulation described in Section 1.2.1.*
2. *Moralize the decision graph and remove utility nodes.*

3. *Triangulate the graph using Algorithm 1.3 and then construct elimination tree using Algorithm 1.2.*
4. *Initialize the decision potentials on the nodes of the elimination tree as described in Section 1.2.5.*
5. *Call CollectEvidence recursively starting from the root node of the elimination tree.*
6. *Read off the decision mapping and the maximum expected utility.*

We now prove that this algorithm does indeed lead to the correct solution to the decision problem. To do this we first introduce some notation. First of all we expand out the decision sequence so that each of the probabilistic variables is represented individually

$$(X_1^1, \ldots, X_1^{k_1}, d_1, X_2^1, \ldots, X_2^{k_2}, d_2, \ldots, X_{M+1}^{k_{M+1}}) \tag{1.51}$$

such that the variables are eliminated in the reverse order to that given. Next we define

$$(C_1^1, \ldots, C_1^{k_1}, C_{d_1}, C_2^1, \ldots, C_2^{k_2}, C_{d_2}, \ldots, C_{M+1}^{k_{M+1}}) \tag{1.52}$$

to be the corresponding sequence of elimination sets obtained by running algorithm 1.3. We also define the subsets

$$T_i^j = \left\{C_1^1, \ldots, C_1^{k_1}, C_{d_1}, C_2^1, \ldots, C_2^{k_2}, d_2, \ldots, C_i^j\right\} \quad j > 0 \tag{1.53}$$

$$T_i^0 = \left\{C_1^1, \ldots, C_1^{k_1}, C_{d_1}, C_2^1, \ldots, C_2^{k_2}, d_2, \ldots, d_{i-1}\right\}. \tag{1.54}$$

We shall also use $\Phi[T_i^j]$ to denote the product of all the decision potentials in the subset T_i^j. Finally we denote

$$P_i^j = \sum_{X_i^{j+1}} \cdots \sum_{X_{M+1}^{k_{M+1}}} P(X|d) \tag{1.55}$$

$$V_i^j = \sum_{X_i^{j+1}} \ldots \max_{d_i} \ldots \sum_{X_{M+1}^{k_{M+1}}} P(X|d)U(X,d) \tag{1.56}$$

so that P_i^j involves only the subsequence of sum marginalizations and represents a conditional probability, whereas V_i^j involves the complete sequence of sum and max marginalizations and represents a partially maximized expected utility. Armed with this notation, we are now ready to proceed with the proof.

The key idea is to show that, after each elimination of a (probability or decision) variable, the product of the decision potentials over the remaining cliques (corresponding to variables that have yet to be eliminated) takes the form

$$\Phi[T_i^j] = (P_i^j, V_i^j / P_i^j). \tag{1.57}$$

Note that $P_1^0 = 1$ since, from (1.55) this is just the sum over all probability variables of the conditional distribution $P(X|d)$, and that V_1^0 is the globally maximum expected utility. Thus (1.57) implies that, once all of the variables are eliminated, the decision potential for the null clique in the elimination tree (the root) will be $(P_1^0, V_1^0/P_1^0)$ and so will contain the maximum expected utility. Furthermore, we shall show that each of the max marginalizations is equivalent to the corresponding max marginalization occuring in the general formulation (1.21) and so each of the decision variables will be set to its optimal value.

We now prove by induction that (1.57) holds after each variable is eliminated. When the elimination tree is initialized, before any variables have been eliminated, we have

$$\Phi[T_{M+1}^{k_{M+1}}] = (P_{M+1}^{k_{M+1}}, V_{M+1}^{k_{M+1}}/P_{M+1}^{k_{M+1}}) \tag{1.58}$$

where $P_{M+1}^{k_{M+1}} = P(X|d)$ and $V_{M+1}^{k_{M+1}} = P(X|d)U(X,d)$, and so this provides a starting point for the inductive proof.

Now suppose that (1.57) holds for some particular values of i and j, and that the next variable to be eliminated is a probabilistic variable, X_i^j. By definition we have

$$\Phi[T_i^j] = \Phi[T_i^{j-1}]\phi_{C_i^j}. \tag{1.59}$$

We now operate on both sides of this equation with the sum-marginalization operator for variable X_i^j. On the left hand side of (1.59) we have

$$\begin{aligned} \mathbf{S}_{X_i^j}\Phi[T_i^j] &= \left(\sum_{X_i^j} P_i^j, \frac{\sum_{X_i^j} P_i^j \left(V_i^j/P_i^j\right)}{\sum_{X_i^j} P_i^j}\right) &(1.60)\\ &= (P_i^{j-1}, V_i^{j-1}/P_i^{j-1}). &(1.61)\end{aligned}$$

Now consider the sum marginalization operator acting on the right hand side of (1.59). We can temporarily de-clutter the notation by writing $X \equiv X_i^j$, $\Phi \equiv \Phi[T_i^{j-1}]$ and $\phi \equiv \phi_{C_i^j}$. Noting that Φ does not depend on X we then have

$$\begin{aligned} \mathbf{S}_{X_i^j}(\Phi\phi) &= \left(\sum_X \gamma_\Phi\gamma_\phi, \frac{\sum_X \gamma_\Phi\gamma_\phi(\mu_\Phi + \mu_\phi)}{\sum_X \gamma_\Phi\gamma_\phi}\right) \\ &= \left(\gamma_\Phi \sum_X \gamma_\phi, \frac{\gamma_\Phi\mu_\Phi \sum_X \gamma_\phi + \gamma_\Phi \sum_X \gamma_\phi\mu_\phi}{\gamma_\Phi \sum_X \gamma_\phi}\right) \\ &= \left(\gamma_\Phi \sum_X \gamma_\phi, \mu_\Phi + \frac{\sum_X \gamma_\phi\mu_\phi}{\sum_X \gamma_\phi}\right) \\ &= \Phi\mathbf{S}_{X_i^j}\phi. &(1.62)\end{aligned}$$

Thus from (1.59), (1.61) and (1.62) we have

$$\Phi[T_i^{j-1}]\mathbf{S}_{X_i^j}\phi_{C_i^j} = (P_i^{j-1}, V_i^{j-1}/P_i^{j-1}). \tag{1.63}$$

The left hand side of (1.63) represents the computation of a message in the clique C_i^j and the absorption of that message in the neighbouring clique nearer the root. From (1.63) we see that the resulting product of decision potentials over the remaining cliques indeed has the form (1.57). This completes the inductive step over i for a given value of j in the case where we absorb a random variable. For a given value of i we can apply this result repeatedly until we have eliminated all of the random variables in the group X_i. At this stage we have the result

$$\Phi[T_i^0] = (P_i^0, V_i^0/P_i^0) \tag{1.64}$$

Next we come to the elimination of the decision variable d_{i-1}. By definition we have

$$\Phi[T_i^0] = \Phi[T_{i-1}^{k_{i-1}}]\phi_{C_{d_{i-1}}}. \tag{1.65}$$

We now operate on both sides of this equation with the max-marginalization operator for the decision variable d_{i-1}. On the left hand side of (1.65) we can use the definition of the max marginalization operator, together with (1.55) and (1.56), to obtain

$$\mathbf{M}_{d_{i-1}}\Phi[T_i^0] = (P_{i-1}^{k_{i-1}}, V_{i-1}^{k_{i-1}}/_{i-1}^{k_{i-1}}) \tag{1.66}$$

where we have used the causal consistency requirement which implies that P_i^0 is independent of d_{i-1} and hence that $P_{i-1}^{k_{i-1}} = P_i^0$. To see this note that P_i^0 is the product of conditional probabilities of the form

$$P_i^0 = P(X_1^1|\cdots)\ldots P(X_{i-1}^{k_i-1})|\cdots) \tag{1.67}$$

and then make use of (1.7).

For the max-marginalization operator acting on the right hand side of (1.65) we note that $\Phi[T_{i-1}^{k_{i-1}}]$ is independent of d_{i-1} since it involves the product of clique potentials over variables later in the elimination sequence. Furthermore, the probability term γ in $\phi_{C_{d_{i-1}}}$ is also independent of d_{i-1} since it involves products of conditional distributions only up to

$$P(X_{i-1}|X_1, \ldots, X_{i-2}, d_1, \ldots, d_{i-2}) \tag{1.68}$$

which, from the conditional independence properties (1.7) implied by the causal consistency requirement, is independent of d_{i-1}. Again we can temporarily de-clutter the notation by writing $\Phi \equiv \Phi[T_{i-1}^{k_{i-1}}]$ and $\phi \equiv \phi_{C_{d_{i-1}}}$. We then have

$$\begin{aligned}
\mathbf{M}_{d_{i-1}}(\Phi\phi) &= \mathbf{M}_{d_{i-1}}(\gamma_\Phi\gamma_\phi, \mu_\Phi + \mu_\phi) \\
&= (\gamma_\Phi\gamma_\phi, \mu_\Phi + \widehat{\mu}_\phi) \\
&= (\gamma_\Phi, \mu_\Phi)(\gamma_\phi, \widehat{\mu}_\phi) \\
&= (\gamma_\Phi, \mu_\Phi)\mathbf{M}_{d_{i-1}}(\gamma_\phi, \mu_\phi) \\
&= \Phi\mathbf{M}_{d_{i-1}}\phi
\end{aligned} \tag{1.69}$$

where $\widehat{\mu}_\phi = \mu_\phi(d_{i-1} = \widehat{d}_{i-1})$ and $\widehat{d}_{i-1} = \arg\max_{d_{i-1}}(\gamma_\phi \mu_\phi)$. Combining (1.65), (1.66) and (1.69) we have

$$\Phi[T_{i-1}^{k_{i-1}}]\mathbf{M}_{d_{i-1}}\phi_{C_{d_{i-1}}} = (P_{i-1}^{k_{i-1}}, V_{i-1}^{k_{i-1}}/P_{i-1}^{k_{i-1}}). \tag{1.70}$$

The operation on the left hand side of (1.70) represents the evaluation of the message resulting from the elimination of the decision variable d_{i-1} and the absorption of that message into the decision potential of the next clique in the elimination tree. Again we see that this takes the general form (1.57).

Thus we have reduced the index i by 1, and so this completes the inductive step over i, and hence the result (1.57) is proved for all permissible values of i and j.

1.3 Reinforcement Learning

1.4 Historical Remarks and Bibliography

Exercises

1.1 ($\star\star$) Apply the general elimation framework summarized in Section 1.2.6 to the graph of Figure 1.5. Do this by considering an elimination ordering $(X_m, X_{m+1}, \ldots, X_L, X_{m-1}, X_{m-2}, \ldots, X_1)$ which starts and some node X_m part way along the chain. Show that after the call to CollectEvidence the root node of the elimination tree contains the expected utility in the utility part of its decision potential.

1.2 ($\star$) Consider the application of the elimination algorithm to the two-stage cancer problem represented by the graph in Figure 1.3. Show that the restriction on the elimination ordering arising from causality means that no computational saving compared to brute force evaluation is possible for this problem.

1.3 ($\star\star\star$) Show that the elimination algorithm 1.1 adds edges to the original graph $\mathcal{G}$ such as to traingulate the graph.

1.4 ($\star\star$) Show that the elimination algorithm 1.1 generates a graph whose cliques are contained within the elimination sets C_i.

1.5 ($\star$) Set out a formal proof of the running intersection property for the elimination algorithm 1.2 as set out in Section 1.2.3.

1.6 ($\star$) Verify that when multiple messages are absorbed into a clique of an elimination tree, the order in which the messages are received is unimportant.

1.7 ($\star$) Apply the elimination algorithm summarized in Section 1.2.6 to the graph given in Figure 1.5. First of all write down a suitable elimination tree for this problem. Then show that, after absorbtion of all the messages, the utility part of the potential in the final node contains the total expected utility.