Graphical Models for Partially Sequential Decision Problems

Thomas D. Nielsen
Department of Computer Science
Aalborg University

August 1, 2001

Preface

This Ph.D.-dissertation is the result of my Ph.D. study at the Department of Computer Science, Aalborg University, Denmark. It concerns the representation, solution and analysis of decision problems formulated in the influence diagram framework as well as other more general modeling languages.

The dissertation consists of ten chapters that are divided into four parts:

Part I deals with partially specified decision problems, and the representation of asymmetric decision problems.

Part II concerns the evaluation of asymmetric decision problems represented in the framework of so-called asymmetric influence diagrams.

Part III addresses issues concerning the analysis of decision problems.

Part IV presents an operational approach to rational decision making based on non-expected utility.

Both Part I and Part II include a chapter describing relevant results published by other authors. Each chapter contains an introductory section, and a section describing terms, notation and existing results relevant for that chapter. Parts of the dissertation have been published either as technical reports, in conference proceedings or have been submitted for publication, as indicated in the abstract of each chapter.

The study has been partly theoretical and partly practical. The results of the practical part are documented in form of the test results presented in chapter 9 and a manual, which appears in Appendix B.

Acknowledgements

The study has been funded by CIT project number 87.2 - “Automated Decision Support for Customer Support Operations - Phase 2”.

I would like to thank my advisor Professor Finn V. Jensen for encouraging me in the study and for many valuable discussions and comments. I would also like to thank the Decision Support Systems group at Aalborg University for providing a very stimulating environment. A special thank to Kristian G. Olesen and Helge Langseth for making these years very memorable.

From January 2000 to July 2000 I was a visiting scholar at Université Pierre et Marie Curie, Paris 6, France. I wish to thank Professor Jean-Yves Jaffray for many stimulating discussions and an inspiring environment. I would also like to thank Pierre-Henri Wuillemin for making me feel very welcome during my stay.

Last but not least I would like to thank Rikke Linding for keeping up with me over these last years.

Aalborg, Denmark — Thomas D. Nielsen
August 2001

Table of Contents

Chapter 1

Introduction

A bad decision can have a good outcome.
A good outcome of a bad decision does
not make the decision better.
- Ross Shachter

Graphical modeling for decision support systems is getting more and more widespread. First of all, graphical models are potentially powerful because they translate a complex problem into an easily understandable form (Almond 1995). Moreover, they provide an appealing way to think of and communicate on the underlying structure of the domain in question.

The influence diagram framework (Howard and Matheson 1981) serves as an effective modeling tool for symmetric decision problems with several decisions; the power of an influence diagram, both as an analysis tool and a communication tool, lies in its ability to concisely and precisely describe the structure of a decision problem (Smith et al. 1993). However, influence diagrams cannot efficiently represent the class of asymmetric decision problems. In order to represent such types of decision problems using the influence diagram framework, the problems must be "symmetrized" by e.g. adding artificial states to capture the asymmetry. Unfortunately, this "symmetrization" tends to obscure the structure of the decision problem, and may impose a significant computational overhead when solving the decision problem, see e.g. (Smith et al. 1993; Qi et al. 1994; Covaliu and Oliver 1995; Shenoy 2000).

When representing a decision problem using an influence diagram, a linear temporal ordering over the decisions is required; this requirement ensures that the decision problem is well-defined. However, the structure of a decision problem often renders certain decisions conditionally independent and in these situations it is unnecessary to impose such an ordering. For instance, if no information is gathered between two decisions, then these decisions can be taken independently of each other. This type of temporal independence is recognized in the framework of valuation based systems (Shenoy 1992), however, temporal independence may be more complicated, e.g. observing a variable A immediately before taking a decision D need not have any impact on this particular decision, see e.g. (Shachter 1999). The latter case also indicates that not all observed variables necessarily have an impact on a particular decision variable; hence, these variables can be considered irrelevant for that decision. Similarly, future variables may also be irrelevant, and for both communicational as well as computational reasons it is important to identify such types of irrelevance. For instance, by identifying the parts of a decision problem that are relevant for a certain decision (which may be of particular importance) we can focus our efforts on these parts when eliciting the utilities and the probabilities. This approach can also be supplemented with an analysis of how sensitive the model is to variations in the utilities and the probabilities; e.g., during elicitation we may focus on values for which the model is particularly sensitive, see e.g. (Bielza et al. 2000).

When solving a decision problem we seek an optimal strategy for the decisions involved. This strategy is typically computed using expected utility theory, i.e., a strategy that maximizes the expected utility is termed an optimal strategy. Unfortunately, it has turned out that expected utility theory has a poor performance when considered as a descriptive model, see e.g. (Allais 1979). From a prescriptive point of view we cannot simply disregard this shortcoming as the task is to propose a strategy which a decision maker is willing to undertake. However, the decision maker is usually not prepared to blindly follow a strategy before he has gained sufficient confidence in the model used. But, since expected utility theory may not be able to simultaneously account for all of his choices, the decision maker is likely to reject the proposed strategy.

In this dissertation we address the representation, solution, and analysis of decision problems in relation to the issues above. First of all, we consider influence diagrams with only a partial temporal ordering over the decisions (partial influence diagrams), and we present a set of necessary and sufficient conditions for a partial influence diagram to be well-defined. As a generalization of the partial influence diagram, we propose another framework, termed asymmetric influence diagrams, for representing and solving asymmetric decision problems; this framework primarily encodes information about asymmetry at the qualitative level, and this information can therefore be read directly from the model. Concerning the analysis of decision problems, we present a method for determining the parts of an influence diagram that are relevant for a particular decision variable. In this context we also give two methods for determining how sensitivity an influence diagram is to variations in the utilities and the probabilities. Finally, we consider rational decision making based on non-expected utility theory, and we show how to define a procedure that selects a compromise strategy which is non-dominated.

1.1 Outline of the dissertation

In Part I we consider the representation of decision problems. Chapter 2 presents some terms and notation used throughout the dissertation, together with relevant results published by other authors. In Chapter 3 we present a set of necessary and sufficient conditions ensuring that an influence diagram with only a partial temporal ordering over the decisions is well-defined. In Chapter 4 we propose a framework (termed asymmetric influence diagrams) as a modeling language for asymmetric decision problems.

Part II concerns the evaluation of decision problems. In Chapter 5 we briefly review some existing evaluation methods for solving symmetric decision problems. These methods form the basis for Chapter 6, in which we propose a method for solving asymmetric decision problems formulated in the asymmetric influence diagram framework.

Part III addresses issues concerning the analysis of influence diagrams. In Chapter 7 we propose a method for decomposing an influence diagram into a collection of smaller influence diagrams. Each of the smaller influence diagrams contains exactly the variables relevant for a particular decision variable, and a solution to the original decision problem can therefore be found by solving the smaller decision problems independently. Chapter 8 describes two methods for doing sensitivity analysis in influence diagrams, with emphasis on uncertain utility parameters.

Part IV considers rational decision making based on non-expected utility theories. Particularly, in Chapter 9 we propose an operational approach to rational decision making based on rank dependent utility (Quiggin 1982).

Finally, in Chapter 10 we make some concluding remarks about the work documented in this dissertation. Topics for future research are also outlined.

Part I

Representing decision problems

Chapter 2

Preliminaries and notation

In this chapter we introduce some terms and notation that will be used in the remainder of this dissertation. For ease of reference, we will also briefly review some existing results published by other authors.

2.1 Graphs

A *graph* $G = (\mathcal{U}, \mathcal{E})$ consists of a finite set of nodes $\mathcal{U}$ and a finite set of edges $\mathcal{E}$. An edge is a pair of nodes (X, Y), where $X, Y \in \mathcal{U}$ and $X \neq Y$; if X and Y are ordered, then the edge is said to be *directed*; otherwise it is *undirected*. A directed edge will be referred to as an *arc*. If all edges in $\mathcal{E}$ are directed, then G is said to be a *directed graph*. Conversely, if all edges are undirected, then G is an *undirected graph*; in this dissertation we shall only consider these two types of graphs. A *subgraph* G' of a graph $G = (\mathcal{U}, \mathcal{E})$ induced by a set of nodes $\mathcal{U}' \subseteq \mathcal{U}$ is the graph $G' = (\mathcal{U}', \mathcal{E}')$, where $\mathcal{E}' = \{(X, Y) | \{X, Y\} \subseteq \mathcal{U}' \text{ and } (X, Y) \in \mathcal{E}\}$. A subgraph $G' = (\mathcal{U}', \mathcal{E}')$ is said to be *complete* if $(X, Y) \in \mathcal{E}'$ for all $X, Y \in \mathcal{U}'$. A maximal complete subgraph is called a *clique*.

A *path* from a node X to a node Y in a graph $G = (\mathcal{U}, \mathcal{E})$ is a sequence $X = X_0, X_1, \ldots, X_n = Y$ of distinct nodes such that (X_{i-1}, X_i) is an edge in $\mathcal{E}$ for all $1 \leq i \leq n$; the *length* of a path is the number of edges it contains. The path is a *directed path* if (X_{i-1}, X_i) is an arc directed from X_{i-1} to X_i, $\forall 1 \leq i \leq n$. An undirected graph is said to be a *tree* if each pair of distinct nodes is connected by exactly one path. A sequence $X = X_0, X_1, \ldots, X_n = Y$ of distinct nodes is said to be a *trail* from X to Y if either $(X_{i-1}, X_i) \in \mathcal{E}$ or $(X_i, X_{i-1}) \in \mathcal{E}$, for all $1 \leq i \leq n$.

A *cycle* is a path with the exception that $X_0 = X_n$, and a *directed cycle* is a directed path with $X_0 = X_n$. A directed graph with no directed cycles is said to be a *directed acyclic graph*. A *rooted tree* is a directed acyclic graph $G = (\mathcal{U}, \mathcal{E})$ obtained from a tree $T = (\mathcal{U}, \mathcal{E}')$ by designating a unique node $R \in \mathcal{U}$ as root and directing all edges away from R.

Given an arc (X, Y) from X to Y, the node X is said to be a *parent* of Y and Y is said to be a *child* of X. The set of parents for a node Y is denoted by $pa(Y)$, and the set of children for X is denoted by $ch(X)$. The set of nodes from which there exists a directed path to a node X is termed the *ancestors* of X (denoted $an(X)$), and the set of nodes to which there exists a directed path from X is referred to as the *descendants* of X (denoted $de(X)$). In an undirected graph $G = (\mathcal{U}, \mathcal{E})$, two nodes X and Y are said to be *adjacent* if $(X, Y) \in \mathcal{E}$. The set

$$adj(X) = \{Y \in \mathcal{U} | (X, Y) \in \mathcal{E}\}$$

is the nodes adjacent to X in G; the adjacency set of a node X also defines the edges *incident* to X.

2.1.1 Triangulation

Let $G = (\mathcal{U}, \mathcal{E})$ be an undirected graph. Then G is said to be *triangulated* (or *chordal*) if every cycle of length greater than 3 has a *chord*; a chord in a cycle is an edge between two non-consecutive nodes in the cycle. A set of *fill-in edges* $\mathcal{T}$ is said to be a *triangulation* of G if the graph $G^t = (\mathcal{U}, \mathcal{E} \cup \mathcal{T})$ is triangulated. An algorithm for determining whether or not an undirected graph $G = (\mathcal{U}, \mathcal{E})$ is triangulated was proposed by Tarjan and Yannakakis (1984); the algorithm has time complexity $O(|\mathcal{U}| + |\mathcal{E}|)$.

A triangulation $\mathcal{T}$ is *minimal* if for any proper subset $\mathcal{T}' \subset \mathcal{T}$ the graph $G = (\mathcal{U}, \mathcal{E} \cup \mathcal{T}')$ is not triangulated; a fill-in edge $T \in \mathcal{T}$ is *redundant* if $\mathcal{T} \backslash \{T\}$ is a triangulation of G. $\mathcal{T}$ is said to be *minimum* if, for any other triangulation $\mathcal{T}'$, it holds that $|\mathcal{T}| \leq |\mathcal{T}'|$.

A node X in G is said to be *eliminated* if i) the nodes adjacent to X is made a complete subgraph of G and ii) X, together with the edges incident to X, are removed. If α is an *elimination ordering* of the nodes in $\mathcal{U}$ (i.e., α is a bijection $\alpha : \mathcal{U} \leftrightarrow \{1, 2, \ldots, |\mathcal{U}|\}$), then the set of fill-in edges $\mathcal{T}_\alpha$ produced by eliminating the nodes according to α is a triangulation of G (see e.g. (Rose et al. 1976)):

$$\begin{aligned} \mathcal{T}_\alpha = \{(X, Y) | (X, Y) \notin \mathcal{E} \text{ and there is a path } X = X_0, \ldots, X_n = Y \text{ s.t.} \\ \alpha(X_i) < \alpha(X) \text{ and } \alpha(X_i) < \alpha(Y), \forall 0 < i < n\}. \end{aligned} \tag{2.2}$$

A graph G is triangulated if and only if there exists an elimination ordering α s.t. the nodes can be eliminated according to α without producing any fill-in edges; α is said to be a *perfect elimination ordering* for G. The problem of finding a "good" triangulation therefore reduces to the problem of finding a "good" elimination ordering of the nodes. Unfortunately, the problem of finding an optimal triangulation for an undirected graph is NP-complete (Yannakakis 1981), see also (Arnborg et al. 1987; Wen 1990).

Different heuristic triangulation methods have been proposed in the literature, see e.g. (Kjærulff 1993) for an overview. Typical rules (optimality criteria) guiding these methods include *minimum fill-in* (as few fill-in edges as possible), *minimum size* (the smallest clique size) and *minimum weight* (the clique with least "weight").[1] One commonly used heuristic algorithm is based on one-step look ahead: Successively choose the next node to be eliminated as the node that minimizes the score (the optimality criterion). An extensive evaluation of different heuristic triangulation algorithms can be found in (Kjærulff 1993).

When referring to the triangulation of a directed acyclic graph $\mathsf{G} = (\mathcal{U}, \mathcal{E})$ we consider its underlying *moral graph*. The moral graph G^{m} of G is obtained by *moralization*:

1. Add an undirected edge for all pairs of non-adjacent nodes with a common child.
2. Drop the direction of all the arcs.

Thus, if $\mathcal{M}$ is the set of edges added during moralization and $\mathcal{T}$ is a triangulation of G^{m}, then the triangulated graph G^{t} of G is given by $\mathsf{G}^{\mathsf{t}} = (\mathcal{U}, \mathcal{E} \cup \mathcal{M} \cup \mathcal{T})$.

2.1.2 Cluster trees and junction trees

A *cluster tree* T for an undirected graph $\mathsf{G} = (\mathcal{U}, \mathcal{E})$ is a tree, where each node $\mathcal{C}$ is a subset (cluster) of the variables in $\mathcal{U}$ and the union of all clusters is the set $\mathcal{U}$. An edge between two nodes $\mathcal{C}_i$ and $\mathcal{C}_j$ in T is labeled with a *separator* that consists of the intersection $\mathcal{C}_i \cap \mathcal{C}_j$. A cluster tree T is said to be a *junction tree* (or *join tree*) if it satisfies the *junction tree property*, i.e., for each pair of nodes $\mathcal{C}_l^*$ and $\mathcal{C}_k^*$ in T, all nodes on the path between $\mathcal{C}_l^*$ and $\mathcal{C}_k^*$ contain the nodes $\mathcal{C}_l^* \cap \mathcal{C}_k^*$. The junction tree property also ensures that the intersection $\mathcal{C}_l^* \cap \mathcal{C}_k^*$ will be a subset of every separator on the path between $\mathcal{C}_l^*$ and $\mathcal{C}_k^*$.

The nodes in a junction tree for an undirected graph G are the cliques in G, and there exists a junction tree T of cliques for G if and only if G is triangulated (Cowell et al. 1999). All junction trees over a triangulated graph have the same set of separators (Jensen and Jensen 1994).

[1]These heuristic rules correspond to the optimality criteria where we seek the minimum number of fill-in edges, the minimum sum of the clique sizes and the triangulated graph with minimum weight, respectively.

From a triangulated graph G^t the cliques can easily be identified: Let α be a perfect elimination ordering for the nodes $\mathcal{U}$ in G^t, and let $\mathcal{C}_i$ be the set of nodes containing node $\alpha^{-1}(i)$ and all the nodes adjacent to $\alpha^{-1}(i)$ at the time when it is eliminated. Then every clique in G^t corresponds to a set $\mathcal{C}_i$, for some $i = 1, \ldots, |\mathcal{U}|$.

A sequence $(\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k)$ of sets is said to have the *running intersection property* if:

$$\forall 1 \leq j \leq k : \mathcal{C}_j \cap (\mathcal{C}_1 \cup \cdots \cup \mathcal{C}_{j-1}) \subseteq \mathcal{C}_i \text{ for some } i < j.$$

If the cliques $(\mathcal{C}_1^*, \mathcal{C}_2^*, \ldots, \mathcal{C}_k^*)$ of a triangulated graph G^t have been ordered to satisfy the running intersection property, then a junction tree for G^t can be constructed by: For each $2 \leq j \leq k$, add an edge between $\mathcal{C}_i^*$ and $\mathcal{C}_j^*$, where $1 \leq i < j$ such that $\mathcal{C}_j^* \cap (\mathcal{C}_1^* \cup \cdots \cup \mathcal{C}_{j-1}^*) \subseteq \mathcal{C}_i^*$.

Alternatively, the junction tree can be constructed from the *junction graph* J for the triangulated graph G^t; the nodes in J correspond to the cliques in G^t, and for each pair of nodes in J with a non-empty intersection there is an edge (separator) labeled with that intersection. A junction tree T of G^t is then a *spanning tree* of J with maximal weight (Jensen and Jensen 1994); the weight of an edge is the number of nodes in its label, and the weight of a junction tree is the sum of the weights of its separators.

2.2 Bayesian networks

A *Bayesian network* (Pearl 1988; Jensen 1996) is a compact representation of a joint probability distribution over a set of variables. Formally, a discrete Bayesian network $BN = (G, \mathcal{R})$ consists of a directed acyclic graph $G = (\mathcal{U}, \mathcal{E})$ and a set of *conditional probability distributions* $\mathcal{R}$. Each node $X \in \mathcal{U}$ (termed a *chance node*) corresponds to a discrete *chance variable* X with a finite number of mutually exclusive and exhaustive states; the *state space* of a variable X is denoted by $sp(X) = (x_1, x_2, \ldots, x_l)$. For each node X in a Bayesian network $BN = (G, \mathcal{R})$ there is a conditional probability distribution $P(X|pa(X)) \in \mathcal{R}$, where $pa(X)$ is the set of parents for X in G. The *domain* of a potential $\phi = P(X|pa(X)) \in \mathcal{R}$ is given by $dom(\phi) = \{X\} \cup pa(X)$.

The set $\mathcal{R}$ of conditional probability distributions corresponds to a multiplicative factorization of the joint probability distribution over $\mathcal{U}$, i.e.,:

$$P(\mathcal{U}) = \prod_{X \in \mathcal{U}} P(X|pa(X)). \tag{2.4}$$

Since there is a one-to-one correspondence between a variable in a Bayesian network $BN = (G, \mathcal{R})$ and a node in G, we shall use the concepts of node and variable interchangeably. This correspondence is also preserved when referring to the elimination of nodes and variables, i.e., when eliminating a variable X we combine all potentials in $\mathcal{R}$ with the variable X in the domain, and then eliminate X by marginalization (usually summation); the elimination of X from $\mathcal{R}$ is reflected by the elimination of X in G.

Probabilistic inference is one of the cornerstones in the Bayesian network framework; this task, however, has been proven to be NP-hard (Cooper 1990). Probabilistic inference can be done by local calculations in the underlying junction tree representation of the Bayesian network, see (Lauritzen and Spiegelhalter 1988; Jensen et al. 1990).

2.3 Influence diagrams

An *influence diagram* can be seen as a Bayesian network augmented with decision nodes and value nodes. Thus, an influence diagram consists of a directed acyclic graph $G = (\mathcal{U}, \mathcal{E})$, where the nodes $\mathcal{U}$ can be partitioned into three disjoint subsets: chance nodes $\mathcal{U}_C$, *decision nodes* $\mathcal{U}_D$ and *value nodes* $\mathcal{U}_V$.

The chance nodes (drawn as circles) correspond to chance variables, and represent events which are not under the direct control of the decision maker. The decision nodes (drawn as squares) correspond to *decision variables* and represent actions under the direct control of the decision maker. In an influence diagram, the decision nodes are ordered totally, thereby indicating the order in which the decisions are made (the ordering of the decision nodes is traditionally represented by a directed path which includes all decision nodes). We shall assume that no *barren nodes* are specified by an influence diagram, since they have no impact on the decisions (Shachter 1986); a chance node or a decision node is said to be barren if it does not precede any other node, or if all its descendants are barren.

Each chance variable and decision variable X is associated with a finite state space $\mathrm{sp}(X)$ which denotes the set of possible outcomes/decision options for X. For a set $\mathcal{U}'$ of variables we define the state space as $\mathrm{sp}(\mathcal{U}') = \times\{\mathrm{sp}(X)|X \in \mathcal{U}'\}$, where $A \times B$ denotes the Cartesian product of A and B. The uncertainty associated with each chance variable C is represented by a conditional probability potential $\phi_C = P(C|\mathrm{pa}(C)) : \mathrm{sp}(\{C\} \cup \mathrm{pa}(C)) \to [0; 1]$; the set of all conditional probability potentials specifies a multiplicative factorization of $P(\mathcal{U}_C|\mathcal{U}_D)$:

$$P(\mathcal{U}_C|\mathcal{U}_D) = \prod_{C \in \mathcal{U}_C} P(C|\mathrm{pa}(C)).$$

The set of value nodes (drawn as diamonds) defines a set of *utility functions* with the restriction that value nodes have no descendants. Each utility function indicates the local utility for a given configuration of the variables in its *domain*; the domain of a utility function ψ_V is denoted by $\mathrm{dom}(\psi_V) = \mathrm{pa}(V)$, where V is the value node associated with ψ_V. The total utility is most commonly calculated as the sum or the product of the local utilities, see (Tatman and Shachter 1990); in the remainder of this dissertation we assume that the total utility is the sum of the local utilities. Analogously to the concepts of variable and node, we shall sometimes use the terms value node and utility function interchangeably.

A *realization* of an influence diagram I is an attachment of probability potentials and utility functions to the appropriate variables in I, i.e., a realization of an influence diagram is a set $\mathcal{R} = \{P(C|pa(C)) : C \in \mathcal{U}_C\} \cup \{\psi_V(pa(V)) : V \in \mathcal{U}_V\}$. So, a realization specifies the quantitative part of the model whereas the graph constitutes the qualitative part. Note that we shall sometimes refer to the graphical structure as an influence diagram, without having specified a realization.

The arcs in an influence diagram can be partitioned into three disjoint subsets, corresponding to the type of node they go into. Arcs into value nodes represent functional dependences by indicating the domain of the associated utility function. Arcs into chance nodes, termed *dependency arcs*, represent probabilistic dependence, whereas arcs into decision nodes, termed *informational arcs*, imply information precedence; if there is an arc from a node X to a decision node D then the state of X is known when decision D is made. In the remainder of this dissertation we shall use $\bar{I}$ to denote the graph obtained from the influence diagram I by removing all informational arcs. Note that all intermediate nodes in a directed path in $\bar{I}$ are chance nodes.

Let $\mathcal{U}_C$ be the set of chance variables and let $\mathcal{U}_D = \{D_1, D_2, \ldots, D_n\}$ be the set of decision variables. Assuming that the decision variables are ordered by their index, then the set of informational arcs induces a partitioning of $\mathcal{U}_C$ into a collection of disjoint subsets $\mathcal{C}_0, \mathcal{C}_1, \ldots, \mathcal{C}_n$. The set $\mathcal{C}_j$ denotes the chance variables observed between decision D_j and D_{j+1} thus, the variables in $\mathcal{C}_j$ occur as parents for D_{j+1}. This induces a *partial order* $\prec$ on $\mathcal{U}_C \cup \mathcal{U}_D$:

$$\mathcal{C}_0 \prec D_1 \prec \mathcal{C}_1 \prec \cdots \prec D_n \prec \mathcal{C}_n. \tag{2.6}$$

The set of variables known to the decision maker when deciding on D_j is called the *informational predecessors* of D_j and is denoted $\text{pred}(D_j)$. By assuming that the decision maker remembers all previous observations and decisions we have $\text{pred}(D_i) \subseteq \text{pred}(D_j)$, (for $i \leq j$), and in particular, $\text{pred}(D_j)$ is the set of variables that occur before D_j under $\prec$. This property is known as *no-forgetting* and from this we can assume that an influence diagram does not specify any redundant no-forgetting arcs, i.e., $pa(D_i) \cap pa(D_j) = \emptyset$ if $D_i \neq D_j$. Nilsson and Lauritzen (2000) propose a generalization of the influence diagram, termed the limited memory influence diagram, in which the no-forgetting assumption is relaxed.

One of the classical influence diagram representations is of the oil wildcatter's problem given by Raiffa (1968).

Example 2.3.1. **[Oil wildcatter]** An oil wildcatter has to decide whether or not to drill. He is uncertain whether the hole is dry, wet, or soaking in oil. The wildcatter has the option of making seismic soundings which may help determine the geological structure of the site. The result will either have a closed, open or diffused reflection pattern which indicates much oil, some oil, or almost no oil, respectively.

An influence diagram representation of this decision problem is given in Figure 2.1.

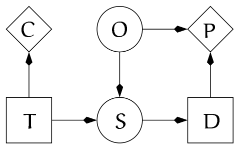

Figure 2.1: An influence diagram representation of the oil wildcatter's problem.

The decision node T represents the decision whether or not to perform seismic soundings. The chance node S represents the outcome of this test (if the test is performed), and the utility function associated with the value node C encodes the cost of performing the test. The decision node D is the decision whether or not to drill for oil, and the chance node O represents the amount of oil in the hole. The utility function associated with the value node P specifies the profit of drilling as a function of the amount of oil in the hole.

The directed path from T to D indicates that we decide on T before deciding on D. The informational arc from S to D specifies that the test result is known before deciding on D. On the other hand, as there is no informational arc from O to either of the decision nodes, the state of O is observed (possibly never) after deciding on D. □

2.4 Independence relations

The independence relations specified by a directed acyclic graph can be determined using the directional-separation (*d-separation*) criterion:

Definition 2.4.1. Let G be a directed acyclic graph. If $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{Z}$ are disjoint subsets of the nodes in G, then $\mathcal{X}$ and $\mathcal{Z}$ are said to be *d-separated* given $\mathcal{Y}$ if there does not exist a trail between a node in $\mathcal{X}$ and a node in $\mathcal{Z}$ s.t.:

- for every intermediate node Y in a converging connection (head-to-head), either $\mathsf{Y} \in \mathcal{Y}$ or Y has a descendant in $\mathcal{Y}$.

- for every intermediate node Y in a serial (head-to-tail) or diverging (tail-to-tail) connection, $\mathsf{Y} \notin \mathcal{Y}$.

If $\mathcal{X}$ and $\mathcal{Z}$ are not d-separated given $\mathcal{Y}$, then we say that $\mathcal{X}$ and $\mathcal{Z}$ are *d-connected* given $\mathcal{Y}$. Each trail satisfying the conditions above is called *active*; otherwise it is said to be *blocked*. To simplify notation we will sometimes use: $\mathcal{X}$ is d-separated from $\mathcal{Z}$ given $\mathcal{Y}$ as a shorthand for $\mathcal{X}$ is d-separated from $\mathcal{Z}$ given $\mathcal{Y} \setminus \mathcal{X}$, if $\mathcal{Y} \cap \mathcal{X} \neq \emptyset$.

Lauritzen et al. (1988) propose an algorithm for verifying whether two sets of nodes $\mathcal{X}$ and $\mathcal{Z}$ are d-separated given another set of nodes $\mathcal{Y}$ in a directed acyclic graph G, see also (Lauritzen et al. 1990). The algorithm works on the moral graph of the subgraph for G, formed by removing all nodes which are not an ancestor of a node in $\mathcal{X} \cup \mathcal{Y} \cup \mathcal{Z}$.

Algorithm 2.4.1. Let $\mathsf{G} = (\mathcal{U}, \mathcal{E})$ be a directed acyclic graph, and let $\mathcal{X}$, $\mathcal{Y}$ and $\mathcal{Z}$ be disjoint sets of nodes. To determine whether $\mathcal{X}$ is d-separated from $\mathcal{Z}$ given $\mathcal{Y}$, do:

1. Construct the subgraph G' of G induced by the set of nodes $\mathcal{X} \cup \mathcal{Y} \cup \mathcal{Z} \cup \mathrm{an}(\mathcal{X} \cup \mathcal{Y} \cup \mathcal{Z})$.
2. Let G^{m} be the moral graph for G'.
3. **If** all paths connecting $\mathcal{X}$ and $\mathcal{Z}$ in G^{m} intersect $\mathcal{Y}$ **then**

   $\mathcal{X}$ is d-separated from $\mathcal{Z}$ given $\mathcal{Y}$

   else

   $\mathcal{X}$ is not d-separated from $\mathcal{Z}$ given $\mathcal{Y}$.

□

The time complexity of the algorithm is $\mathrm{O}(|\mathcal{U}|^2)$.

Geiger et al. (1990a) describe an algorithm for detecting the nodes that are d-separated from a set of nodes $\mathcal{J}$ given another set of nodes $\mathcal{L}$. The algorithm performs a constant number of operations for each arc encountered; hence, the time complexity of the algorithm is $\mathrm{O}(|\mathcal{E}|)$.

Algorithm 2.4.2. Let $\mathsf{G} = (\mathcal{U}, \mathcal{E})$ be a directed acyclic graph and let $\mathcal{J}$ and $\mathcal{L}$ be disjoint sets of nodes. The set $\mathcal{D}$ of nodes d-separated from $\mathcal{J}$ given $\mathcal{L}$ can be determined as follows:

1. Construct the table:

$$\text{descendent}[\mathsf{X}] = \begin{cases} \mathtt{true} & \text{if } \mathsf{X} \in \mathcal{L} \text{ or } \mathsf{X} \text{ has a descendant in } \mathcal{L} \\ \mathtt{false} & \text{otherwise.} \end{cases}$$

2. Construct the directed graph $\mathsf{G}' = (\mathcal{U}, \mathcal{E}')$, where:

$$\mathcal{E}' = \mathcal{E} \cup \{(\mathsf{Y}, \mathsf{X}) | (\mathsf{X}, \mathsf{Y}) \in \mathcal{E}\}.$$

3. Use Algorithm 2.4.3 to find the set of reachable nodes $\mathcal{D}'$ to which there is a legal path from $\mathcal{J}$ in G'; a pair of arcs $((\mathsf{X}, \mathsf{Y}), (\mathsf{Y}, \mathsf{Z}))$ is legal if and only if $\mathsf{X} \neq \mathsf{Z}$ and either:
   a) Y is a head-to-head node on the trail $(\mathsf{X}, \mathsf{Y}, \mathsf{Z})$ in G and descendant$[\mathsf{Y}] =$ $\mathtt{true}$ or
   b) Y is not a head-to-head node on the trail $(\mathsf{X}, \mathsf{Y}, \mathsf{Z})$ in G and $\mathsf{Y} \notin \mathcal{L}$.
4. Return $\mathcal{D}$, where $\mathcal{D} = \mathcal{U} \backslash (\mathcal{D}' \cup \mathcal{J} \cup \mathcal{L})$.

□

The following algorithm is used by Algorithm 2.4.2 to determine the set of "reachable" nodes.

Algorithm 2.4.3. Let $\mathsf{G} = (\mathcal{U}, \mathcal{E})$ be a directed graph, let $\mathcal{F}$ be a set of illegal pairs of links and let $\mathcal{J}$ be a set of nodes. To determine the set of nodes $\mathcal{D}'$ reachable from $\mathcal{J}$ via a legal path, do:

1. Set $\mathcal{D}' = \emptyset$ and add a new node S to $\mathcal{U}$.
2. **For** each $\mathsf{X} \in \mathcal{J}$
   a) Add (S, X) to $\mathcal{E}$ and label it with 1.
   b) Set $\mathcal{D}' := \mathcal{D}' \cup \{\mathsf{X}\}$.
3. Label all other arcs with "undefined".
4. Set $\mathtt{i} := 1$.
5. **Repeat**
   a) **For** each arc (X, Y) which is not labeled "undefined"
      If (X, Y) is adjacent to an arc (Z, X) labeled with $\mathtt{i}$ s.t. $((\mathsf{Z}, \mathsf{X}), (\mathsf{X}, \mathsf{Y}))$ is a legal pair **then**
         i) Label (X, Y) with $\mathtt{i} + 1$.
         ii) Set $\mathcal{D}' := \mathcal{D}' \cup \{\mathsf{Y}\}$.
   b) Set $\mathtt{i} := \mathtt{i} + 1$.

   Until no such legal pair of arcs exists.
6. Return $\mathcal{D}'$.

□

Notice that the algorithm can easily be modified to determine the nodes d-connected to $\mathcal{J}$ given $\mathcal{L}$ by simply returning the reachable nodes $\mathcal{D}'$ found in step 3 of Algorithm 2.4.2.

Another recently proposed algorithm for detecting d-separation is the *Bayes-ball* algorithm by Shachter (1998). The algorithm is actually based on *D-separation*, which can be seen as a generalization of the d-separation criterion in the sense that it takes deterministic relations into account (Geiger et al. 1990b); the time complexity of the algorithm is $\mathsf{O}(|\mathcal{U}| + |\mathcal{E}|)$.

Chapter 3

Partial influence diagrams

When formulating a decision problem as an influence diagram, a linear temporal ordering of the decision variables is required; this constraint ensures that the decision problem is well-defined. However, the structure of a decision problem often renders certain decisions conditionally independent, and in such situations it is unnecessary to impose a linear temporal ordering on the decisions. In this chapter we work with partial influence diagrams, i.e., influence diagrams with only a partial temporal ordering specified. We present a set of necessary and sufficient conditions ensuring that a partial influence diagram is well-defined. These conditions are used as a basis for an algorithm for determining whether or not a partial influence diagram is well-defined.

The content of this chapter is based on (Nielsen and Jensen 1999b) and (Nielsen and Jensen 2001b).

3.1 Introduction

The influence diagram framework (Howard and Matheson 1981) serves as a powerful modeling tool for symmetric decision problems with several decisions. However, influence diagrams require a linear temporal ordering of the decisions, and this is often felt as an unnecessary constraint. For instance, if no information is gathered between two decisions, then these decisions can be taken independently of each other. This type of very obvious temporal independence is recognized in the framework of valuation based systems (Shenoy 1992), but temporal independence may be more complicated. For example, observing a variable A immediately before taking a decision D need not have any impact on this particular decision. Hence, we look for an operational characterization of temporal independence in influence diagrams.

The advantages of having an operational characterization of temporal independence are twofold. First of all, when solving an influence diagram we basically eliminate the variables in reverse temporal order, see e.g. (Shachter 1986; Shenoy 1992; Jensen et al. 1994; Zhang 1998); eliminating a variable produces a table (function) over all non-eliminated neighbours. However, the reverse temporal order of elimination has a tendency of creating very large tables (usually much larger than for Bayesian networks of the same complexity). Thus, if we could relax the temporal order to a partial order, we would have more freedom when looking for a good elimination ordering. To put it in another way: When a decision variable D is eliminated we determine a policy for D given its past. Hence, if we can reduce the past to contain only the variables required for taking that decision, then we have reduced the domain of the policy.

The second advantage concerns the modeling process. That is, will we allow two decisions to be taken independently of each other? Or, do we allow an observation to be made independently of a certain decision? Hence, we work with partial specifications of influence diagrams, and would therefore like to know whether or not this partial structure is ambiguous, and if it is ambiguous we would like the system to give suggestions for further specification of the temporal order. An unambiguous partial influence diagram is said to represent a well-defined decision problem.

In this chapter we give a set of operational rules for determining whether or not a partial influence diagram represents a well-defined decision problem. These rules form the basis of an algorithm for answering this question. The algorithm can furthermore be used to pinpoint where the model should be changed in order to make it unambiguous.

3.2 Preliminaries

Solving an influence diagram amounts to computing a policy for the decisions involved. A policy can be seen as a prescription of responses to earlier observations and decisions, and the set of policies for all the decision variables constitutes a strategy for the influence diagram.

Definition 3.2.1. A *strategy* for an influence diagram I is a set of functions $\bar{\Delta} = \{\bar{\delta}_{\mathsf{D}} | \mathsf{D} \in \mathcal{U}_{\mathsf{D}}\}$, where $\bar{\delta}_{\mathsf{D}}$ is a *policy* given by:

$$\bar{\delta}_{\mathsf{D}} : \mathrm{sp}(\mathrm{pred}(\mathsf{D})) \rightarrow \mathrm{sp}(\mathsf{D}).$$

The set of policies for a decision variable D is denoted by $\bar{\delta}^*_{\mathsf{D}}$, and the set of all strategies is denoted by $\bar{\Delta}^*$.

A policy $\bar{\delta}_D$ describes the relationship between the decision variable D and its predecessors. This relationship can equivalently be expressed in terms of the conditional probability potential $P_{\bar{\delta}_D}(D|\text{pred}(D))$:

$$P_{\bar{\delta}_D}(d|\bar{r}) = \begin{cases} 1 & \text{if } \bar{\delta}_D(\bar{r}) = d, \\ 0 & \text{otherwise.} \end{cases}$$

For notational convenience we shall use $P_{\bar{\delta}}(D|\text{pred}(D))$ to denote the conditional probability potential specified above.

Given an influence diagram I and a realization specified by $\{P(C|pa(C)) : C \in \mathcal{U}_C\} \cup \{\psi_V(pa(V)) : V \in \mathcal{U}_V\}$, the *expected utility function* for the last decision D_n under the strategy $\bar{\Delta}$ is defined as (see also (Zhang 1994)):

$$\begin{aligned} \rho^{\bar{\Delta}}_{D_n}(\text{pred}(D_n)) = \sum_{D_n} \sum_{\mathcal{C}_n} & P_{\bar{\delta}}(D_n|\text{pred}(D_n))P(\mathcal{C}_n|\text{pred}(D_n), D_n) \\ & \sum_{V \in \mathcal{U}_V} \psi_V(pa(V)), \end{aligned} \tag{3.3}$$

where $\mathcal{C}_n$ is the set of variables observed (possibly never) after decision D_n is taken; cf. 2.6.

The expected utility function for decision $D_k (k < n)$ under $\bar{\Delta}$ is defined recursively (Equation 3.3 terminates the recursion):

$$\rho^{\bar{\Delta}}_{D_k}(\text{pred}(D_k)) = \sum_{D_k} \sum_{\mathcal{C}_k} P_{\bar{\delta}}(D_k|\text{pred}(D_k))P(\mathcal{C}_k|\text{pred}(D_k), D_k)\rho^{\bar{\Delta}}_{D_{k+1}}. \tag{3.4}$$

By expanding Equation 3.4 the expected utility of I under the strategy $\bar{\Delta}$ is defined as:

$$\begin{aligned} \rho^{\bar{\Delta}} &= \sum_{\mathcal{U}_D} \sum_{\mathcal{C}_0} \cdots \sum_{\mathcal{C}_n} P(\mathcal{C}_0) \prod_{i=1}^{n} P(\mathcal{C}_i|\text{pred}(D_i), D_i)P_{\bar{\delta}}(D_i|\text{pred}(D_i)) \\ &\qquad \sum_{V \in \mathcal{U}_V} \psi_V(pa(V)) \qquad (3.5) \\ &= \sum_{D \in \mathcal{U}_D} \sum_{C \in \mathcal{U}_C} \prod_{D \in \mathcal{U}_D} P_{\bar{\delta}}(D|\text{pred}(D)) \prod_{C \in \mathcal{U}_C} P(C|pa(C)) \sum_{V \in \mathcal{U}_V} \psi_V(pa(V)). \end{aligned}$$

The *maximum expected utility* of I is therefore given by:

$$\rho = \max_{\bar{\Delta} \in \bar{\Delta}^*} \rho^{\bar{\Delta}}.$$

A strategy $\bar{\Delta} = (\bar{\delta}_{D_1}, \ldots, \bar{\delta}_{D_n})$ that maximizes the expected utility of I is termed an *optimal strategy* for I (denoted by $\Delta = (\delta_{D_1}, \ldots, \delta_{D_n})$), i.e., a strategy $\bar{\Delta}$ is an optimal strategy if $\rho^{\bar{\Delta}} = \rho$; the policy δ_{D_i} is termed an *optimal policy* for D_i, $\forall 1 \leq i \leq n$.

From Equation 3.4 we have that if the policy $\bar{\delta}_{D_i}$ specifies an alternative that maximizes the expected utility for decision D_i (for all $1 \leq i \leq n$), then the strategy $\bar{\Delta} = (\bar{\delta}_{D_1}, \ldots, \bar{\delta}_{D_n})$ is an optimal strategy and, in particular, $\bar{\delta}_{D_i}$ is an optimal policy. The optimal policy for decision D_n can now be written as:

$$\delta_{D_n}(\text{pred}(D_n)) = \arg\max_{D_n} \sum_{\mathcal{C}_n} P(\mathcal{C}_n | \mathcal{C}_0, D_1, \ldots, \mathcal{C}_{n-1}, D_n) \sum_{V \in \mathcal{U}_V} \psi_V(pa(V)), \tag{3.7}$$

and similarly for the maximum expected utility for D_n:

$$\rho_{D_n}(\text{pred}(D_n)) = \max_{D_n} \sum_{\mathcal{C}_n} P(\mathcal{C}_n | \mathcal{C}_0, D_1, \ldots, \mathcal{C}_{n-1}, D_n) \sum_{V \in \mathcal{U}_V} \psi_V(pa(V)). \tag{3.8}$$

In general, the maximum expected utility for decision D_k is given by:

$$\rho_{D_k}(\text{pred}(D_k)) = \max_{D_k} \sum_{\mathcal{C}_k} P(\mathcal{C}_k | \mathcal{C}_0, D_1, \ldots, \mathcal{C}_{k-1}, D_k) \rho_{D_{k+1}},$$

and the optimal policy is:

$$\delta_{D_k}(\text{pred}(D_k)) = \arg\max_{D_k} \sum_{\mathcal{C}_k} P(\mathcal{C}_k | \mathcal{C}_0, D_1, \ldots, \mathcal{C}_{k-1}, D_k) \rho_{D_{k+1}}. \tag{3.10}$$

By continuously expanding Equation 3.10, we get the following expression for the optimal policy for decision D_k:

$$\begin{aligned}\delta_{D_k}(\text{pred}(D_k)) = \arg\max_{D_k} \sum_{\mathcal{C}_k} \cdots \max_{D_n} \sum_{\mathcal{C}_n} \\ P(\mathcal{C}_k, \ldots, \mathcal{C}_n | \mathcal{C}_0, \ldots, \mathcal{C}_{k-1}, D_1, \ldots, D_n) \sum_{V \in \mathcal{U}_V} \psi_V(pa(V)).\end{aligned} \tag{3.11}$$

Not all the variables observed (i.e. $\text{pred}(D_k)$) do necessarily influence the optimal policy for D_k, hence, we introduce the notion of a required variable:

Definition 3.2.2. Let I be an influence diagram and let D be a decision variable in I. The variable $X \in \text{pred}(D)$ is said to be *required* for D if there exists a realization of I and a configuration $\bar{y}$ over $dom(\delta_D) \setminus \{X\}$ s.t. $\delta_D(x_1, \bar{y}) \neq \delta_D(x_2, \bar{y})$, where $dom(\delta_D)$ is the domain of δ_D and x_1 and x_2 are states of X. The set of variables required for D is denoted $\text{req}(D)$.

Finally, Equation 3.11 conveys that in order to determine an optimal strategy for an influence diagram we have to perform a series of alternating max-marginalizations and sum-marginalizations to eliminate the variables. The order in which these eliminations are performed must respect the partial order (see

Page 12) induced by the influence diagram; thus, we define a *legal elimination ordering* as a bijection $\alpha : \mathcal{U}_C \cup \mathcal{U}_D \leftrightarrow \{1, 2, \ldots, |\mathcal{U}_C \cup \mathcal{U}_D|\}$, where $X \prec Y$ implies $\alpha(X) < \alpha(Y)$.[1] Note that a legal elimination ordering is not necessarily unique, since the chance variables in the sets $\mathcal{C}_j$ can be commuted. Even so, any two legal elimination orderings result in the same optimal strategy since the decision variables are totally ordered and sum-operations commute; the total ordering of the decision variables ensures that the relative elimination order for any pair of variables of opposite type is invariant under the legal elimination orderings (this is needed since a max-operation and a sum-operation do not commute in general).

3.3 Unambiguous representation of decision problems

In the section above we described the evaluation of an influence diagram w.r.t. the maximum expected utility principle. The underlying assumption was a total ordering of the decision variables ensuring that the optimal strategy is independent of the legal elimination orderings.

However, it may not be necessary to have a total ordering of the decision variables. For instance, if $\mathcal{C}_k = \emptyset$ then D_k and D_{k+1} can be commuted. This can also be seen from the optimal policy for a decision variable (Equation 3.11), where the elimination order for any two neighbouring variables of opposite type may be permuted if the variables do not occur in the same function; we assume that the decision maker is indifferent between any two optimal strategies, i.e., the decision maker does not prefer one optimal strategy over another optimal strategy.

3.3.1 Two examples

Before we proceed with the description of the partial influence diagram, we present the following examples illustrating the occurrence of temporal (in)dependence between decision variables.

Example 3.3.1. [**Stud farm**] This example is a revised version of the stud farm problem given by Jensen (1996). The stallion Brian(B) has sired Dorothy(D) with the mare Ann(A) and sired Eric(E) with the mare Cecily(C). Dorothy and Fred(F) are the parents of Henry(H), and Eric has sired Irene(I) with Gwenn(G). Ann is the mother of both Fred and Gween, but their fathers are not related in any way. The colt John(J) with the parents Henry and Irene has been born

[1] Variables with higher numbers are eliminated before variables with lower numbers.

recently; unfortunately, it turns out that John suffers from a life threatening hereditary disease carried by a recessive gene. The disease is so serious that John is displaced instantly and, as the stud farm wants the gene out of production, Henry and Irene are taken out of breeding. Now, it should be decided whether or not Fred and/or Gwen should be taken out of production (see Figure 3.1).

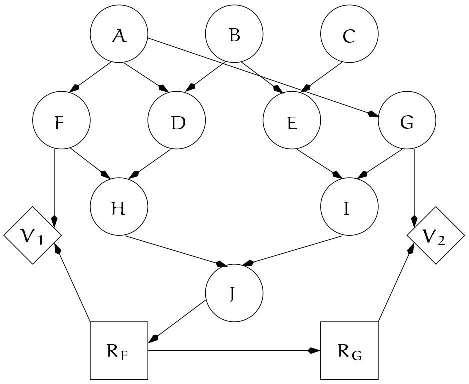

Figure 3.1: The geneological structure of the horses in the stud farm. The decision variables represent the decisions of whether or not to take Fred and/or Gwen out of production.

In the influence diagram shown in Figure 3.1, the chance variable J(John) is the only variable observed prior to deciding on Fred(R_F). Furthermore, the remaining chance variables are observed (possibly never) after deciding on Gwen (R_G).

The maximum expected utility of the decision whether or not to remove Fred is given by:

$$\rho_{R_F}(J) = \max_{R_F} \max_{R_G} \left(\sum_{\mathcal{U}_C \setminus \{J\}} P(\mathcal{U}_C \setminus \{J\} | J)(\psi_{V_1}(F, R_F) + \psi_{V_2}(G, R_G)) \right),$$

where $\mathcal{U}_C = \{A, B, C, D, E, F, G, H, I, J\}$. Since max-operations commute it follows that the ordering of the decision variables is of no importance. That is, we may decide on Gwen before deciding on Fred, assuming that J is observed prior to the decision first made. Hence, the informational arc between the decision variables is redundant, if the chance variable J occurs as a parent of both decision variables. □

As opposed to the example above, the following example presents an influence diagram in which the decision nodes can not be commuted.

Example 3.3.2. Consider the influence diagram I depicted in Figure 3.2. When deciding on D_1 we wish to maximize the expected utility w.r.t. ψ_V; A has not been observed and may therefore influence the decision made. Now assume that we decide on D_2 before deciding on D_1. In this scenario the chance variable A is observed before D_1 is decided upon hence, the optimal policies for D_1 (and therefore the maximum expected utility of I) relative to those scenarios may differ, i.e., the informational arc between D_1 and D_2 is not redundant.

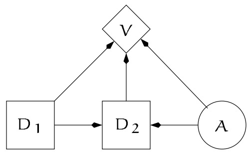

Figure 3.2: The figure represents an influence diagram in which the informational arc (D_1, D_2) cannot be reversed without affecting the maximum expected utility of I.

This can also be seen from the maximum expected utility for I relative to those scenarios. The maximum expected utility relative to the influence diagram depicted in Figure 3.2 is given by:

$$\rho = \max_{D_1} \sum_{A} \max_{D_2} P(A)\psi_V(D_1, A, D_2).$$

The maximum expected utility, after reversing the informational arc, is given by:

$$\rho = \sum_{A} \max_{D_2} \max_{D_1} P(A)\psi_V(D_1, A, D_2).$$

From these expressions it immediately follows that for some realizations of I the maximum expected utility is dependent on whether or not A has been observed before D_1. $\square$

3.3.2 Partial influence diagrams and decision scenarios

Given that a total ordering of the decision variables may not be needed, we define a *partial influence diagram* as a directed acyclic graph consisting of decision nodes, chance nodes and value nodes. Value nodes have no children, but chance nodes may have several decision nodes as children and no ordering is imposed on the decisions.

Analogously to the influence diagram, we define a realization of a partial influence diagram as an attachment of potentials to the appropriate variables, i.e., the chance variables are associated with conditional probability potentials and the value nodes are associated with utility functions.

Since the semantics of a partial influence diagram correspond to the semantics of an influence diagram, a partial influence diagram induces a partial order $\prec$ on the nodes $\mathcal{U}_C \cup \mathcal{U}_D$, as defined by the transitive closure of the following relation:

- $Y \prec D_i$, if (Y, D_i) is an arc in I ($D_i \in \mathcal{U}_D$).
- $D_i \prec Y$, if $(D_i, X_1, X_2, \ldots, X_m, Y)$ is a directed path in I ($Y \in \mathcal{U}_C \cup \mathcal{U}_D$ and $D_i \in \mathcal{U}_D$).
- $D_i \prec A$, if $A \not\prec D_j$ for all $D_j \in \mathcal{U}_D$ ($A \in \mathcal{U}_C$ and $D_i \in \mathcal{U}_D$).
- $D_i \prec A$, if $A \not\prec D_i$ and $\exists D_j \in \mathcal{U}_D$ s.t. $D_i \prec D_j$ and $A \prec D_j$ ($A \in \mathcal{U}_C$ and $D_i \in \mathcal{U}_D$).

In what follows we say that two different nodes X and Y in a partial influence diagram I are *incompatible* if $X \not\prec Y$ and $Y \not\prec X$. Note that a chance node A is incompatible with a decision node D if there exists a decision node D' s.t. D and D' are incompatible, (A, D') is an informational arc in I and $A \notin \text{pred}(D)$. This situation is illustrated in Figure 3.3, where the chance variable F is incompatible with the decision variables D_3 and D_4; the model was originally presented by Jensen et al. (1994) where the ordering of the decision variables corresponded to their index.

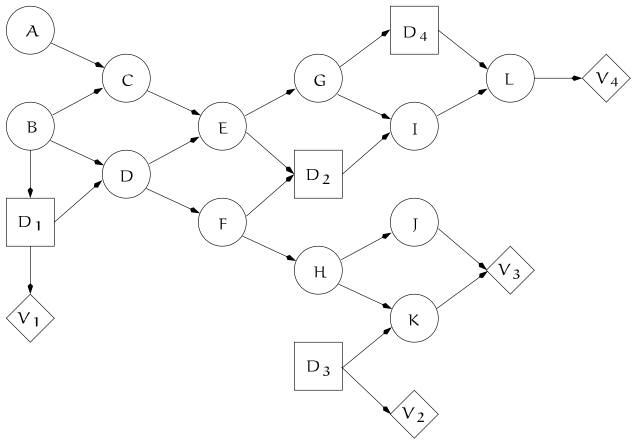

Figure 3.3: The figure illustrates a partial influence diagram which specifies the partial order $\{B\} \prec D_1 \prec \{E, F, G, D_2, D_4\} \prec \mathcal{C}_4$, $\{B\} \prec D_1 \prec \{E, F\} \prec D_2 \prec \mathcal{C}_4$, $\{B\} \prec D_1 \prec \{G\} \prec D_4 \prec \mathcal{C}_4$ and $D_3 \prec \mathcal{C}_4$ ($\mathcal{C}_4$ denotes the chance variables observed (possibly never) after deciding on all the decisions). Thus, D_2, D_3 and D_4 are pairwise incompatible, whereas e.g. D_1 and D_4 are not. Notice that F is incompatible with D_3 and D_4.

As for the traditional influence diagram we seek to identify an optimal strategy when evaluating a partial influence diagram. However, since a partial influence diagram may specify incompatible variables, we define a total order for a partial influence diagram to describe the relative temporal order of incompatible variables:

Definition 3.3.1. Let I be a partial influence diagram and let $\mathcal{U}' = \mathcal{U}_D \cup \mathcal{U}_C$ denote the set of decision variables and chance variables in I. A *total ordering* of I is a bijection $\beta : \mathcal{U}' \leftrightarrow \{1, 2, \ldots, |\mathcal{U}'|\}$. A total ordering of I is said to be an *admissible total ordering* if $X \prec Y$ implies that $\beta(X) < \beta(Y)$, where $\prec$ is the partial order induced by I.

In what follows $<_\beta$ will be used to denote the total ordering β s.t. $X <_\beta Y$ if and only if $\beta(X) < \beta(Y)$ (the index β will be omitted if this does not introduce any confusion). E.g. $E < F < D_2 < B < D_1 < G < D_4 < D_3 < \mathcal{C}_4$ is a total ordering of the partial influence diagram in Figure 3.3, but it is not admissible since it contradicts $D_1 \prec D_2$. Notice, that an admissible total ordering of a partial influence diagram I implies that I can be seen as an influence diagram assuming that redundant no-forgetting arcs have been removed.

Based on the informational predecessors for a decision variable (induced by a total ordering), we define a strategy relative to a total ordering:

Definition 3.3.2. A *strategy relative to a total order* $<$ is a set of functions $\bar{\Delta}^< = \{\bar{\delta}_D^< | D \in \mathcal{U}_D\}$, where $\bar{\delta}_D^<$ is a *policy relative to* $<$ given by:

$$\bar{\delta}_D^< : \mathrm{sp}(\mathrm{pred}(D)^<) \to \mathrm{sp}(D),$$

and $\mathrm{pred}(D)^< = \{X | X < D\}$; the index $<$ in $\mathrm{pred}(D)^<$ will be omitted if this does not introduce any confusion. The set of all policies for a decision variable D relative to $<$ is denoted $\bar{\delta}_D^{<*}$ and the set of all strategies relative to $<$ is denoted $\bar{\Delta}^{<*}$.

Given a realization of a partial influence diagram I, the *expected utility* of I w.r.t. $<$ under the strategy $\bar{\Delta}^<$ is defined as for the influence diagram by letting $<$ represent the temporal ordering of the variables. The *maximum expected utility* of I w.r.t. $<$ is therefore given by:

$$\rho = \max_{\bar{\Delta}^< \in \bar{\Delta}^{<*}} \rho^{\bar{\Delta}^<}.$$

A strategy $\bar{\Delta}^< = (\bar{\delta}_{D_1}^<, \ldots, \bar{\delta}_{D_n}^<)$ relative to $<$ is termed an *optimal strategy relative to* $<$ (denoted $\Delta^< = (\delta_{D_1}^<, \ldots, \delta_{D_n}^<)$), if the strategy maximizes the expected utility of I w.r.t. $<$; a policy $\delta_D^<$ contained in an optimal strategy relative to $<$ is termed an *optimal policy relative to* $<$. Since a total order for a partial influence diagram need not be admissible, we define an *admissible optimal* *strategy* for a realization of a partial influence diagram as an optimal strategy relative to an admissible total order.

As for the influence diagram, an optimal policy for a decision variable does not necessarily depend on all the variables observed. Hence, we say that an observed variable X is *required* for a decision variable D w.r.t. $<$ if there is a realization of I s.t. the optimal policy for D relative to $<$ is a non-constant function over X.

Before characterizing the class of partial influence diagrams that can be considered well-defined, we need the following definition of similar strategies:

Definition 3.3.3. Let I be a partial influence diagram and let $<'$ and $<''$ be two admissible total orderings of I. Let $\Delta^{<'}$ and $\Delta^{<''}$ be admissible optimal strategies for I relative to $<'$ and $<''$, respectively. Then $\Delta^{<'}$ and $\Delta^{<''}$ are said to be *similar* if $\rho^{\Delta^{<'}} = \rho^{\Delta^{<''}}$.

Based on this definition we have the following characterization of a well-defined decision scenario:

Definition 3.3.4. A realization of a partial influence diagram I is said to *define a decision scenario* if all admissible optimal strategies for I are similar. A partial influence diagram is said to *define a decision scenario* if all its realizations define a decision scenario.

This definition characterizes the class of partial influence diagrams which can be considered *well-defined* since the set of admissible total orderings for a partial influence diagram I corresponds to the set of legal elimination orderings for I (note that the traditional influence diagram defines a decision scenario). However, it also conveys the problem of only having a partial temporal ordering of the decision variables; the relative temporal order of a chance variable (eliminated by summation) and a decision variable (eliminated by maximization) may vary under different admissible orderings and summation and maximization do not commute in general.

In correspondance with the permutations of chance variables in any legal elimination ordering for an influence diagram, we define the following relation for any admissible total ordering:

Definition 3.3.5. Let $<$ be an admissible total order, and let X and Y be two neighbouring variables under $<$, fulfilling one of the following three conditions:

- X and Y are both chance variables.
- X and Y are both decision variables.
- X and Y are incompatible.

The ordering $<'$ obtained from $<$ by permuting X and Y according to the rules above is said to be *C-equivalent* to $<$, denoted $< \equiv_C <'$.

Proposition 3.3.1. The transitive closure of $\equiv_C$ is an equivalence relation.

Theorem 3.3.1. All admissible orderings of a partial order $\prec$ are C-equivalent.

Proof. Induction in the number of elements in the partial order.

If the partial order consists of one element, then the theorem is trivial. Assume the theorem to hold for all partial orders over n elements. Consider a partial order over $n+1$ elements, and let $X_1 < \cdots < X_{n+1}$ be an admissible total ordering. It is sufficient to prove that any admissible ordering $Y_1 <' \cdots <' Y_{n+1}$ is C-equivalent to an admissible ordering starting with X_1. If this hold, then by removing X_1 we get a partial order over n elements and two admissible orderings $X_2 < \cdots < X_{n+1}$ and $Y_2 <' \cdots <' Y_{n+1}$. From the induction hypothesis we have that they are C-equivalent, and accordingly $X_1 < \cdots < X_{n+1}$ and $Y_1 <' \cdots <' Y_{n+1}$ are C-equivalent.

To prove the claim, consider the position of X_1 in the $<'$ ordering, and assume that $X_1 = Y_{i+1}$. As $X_1 < Y_i$ and $Y_i <' X_1$, the two elements must be either chance variables or incompatible (both $<$ and $<'$ are admissible). Therefore Y_i and X_1 can be permuted in $<'$, and we can continue so until X_1 is at the first position. $\square$

So we are looking for a set of necessary and sufficient conditions ensuring that all admissible orderings yield the same strategies. Actually, we will look for conditions ensuring that orderings C-equivalent with an admissible ordering yield similar strategies; this is a bit broader as we allow permutation of two neighbouring decision variables. From Theorem 3.3.1 we infer that we can narrow down the scope to neighbouring variables of opposite type. Hence, we look for a necessary and sufficient set of conditions granting commutation of two incompatible neighbouring variables of opposite type.

Definition 3.3.6. Let I be a partial influence diagram and let A be a chance variable incompatible with a decision variable D in I. Then A is said to be *significant* for D if there is a realization and an admissible total order $<$ for I s.t.

- A occurs immediately before D under $<$.
- The optimal strategy relative to $<$ is different from the one achieved by permuting A and D in $<$.

A chance variable is said to be *significant for* D *w.r.t.* $<$ if the above conditions are satisfied w.r.t. $<$.

Notice that if a chance variable A is significant for a decision variable D w.r.t. $<$, then A is required for D w.r.t. $<$.

Based on the definitions above we present the following theorem which characterizes the constraints necessary and sufficient for a partial influence diagram to define a decision scenario.

Theorem 3.3.2. The partial influence diagram I defines a decision scenario if and only if for each decision variable D there does not exist a chance variable A significant for D.

Proof. If I does not define a decision scenario, then there exists a realization and two admissible orderings $<$ and $<'$ s.t. the optimal strategy relative to $<$ is different from the optimal strategy relative to $<'$. From Theorem 3.3.1 we have that $<'$ can be obtained from $<$ by successively commuting neighbouring variables; however, at some point the variables X and Y are to be commuted yielding different optimal strategies. Consequently X and Y must be of opposite type, i.e., if Y is a chance variable and X is a decision variable, then Y is significant for X.

Conversely, if A is significant for D, then from Definition 3.3.4 and Definition 3.3.6 we have that I does not define a decision scenario. □

So we have reduced the task to the following: Let I be a partial influence diagram, and let A be a chance variable incompatible with a decision variable D. Is A significant for D?

As mentioned previously, a chance variable A incompatible with a decision variable D is significant for D w.r.t. an admissible total order $<$ if A is required for D w.r.t. $<$. Thus, we actually need to determine whether A is required for D since an admissible total order for a partial influence diagram I implies that I can be seen as an influence diagram. Shachter (1998) presents an algorithm for determining the so-called *requisite information* for a decision variable in an influence diagram. Unfortunately, the algorithm does not find the minimal set of variables being required for a particular decision variable as shown by the following example.

Example 3.3.3. When running the algorithm *Decision Bayes-ball* (Shachter 1998) on the influence diagram depicted in Figure 3.4, the chance variable B is marked as requisite for decision D_1. However, B is not required for the optimal policy for D_1, i.e., the elimination order for B relative to D_1 is of no importance. □

The following method, which corresponds to iteratively replacing decisions by their policies, has the same drawback. For an influence diagram we start off with the moral graph, i.e., informational arcs are removed, edges are added

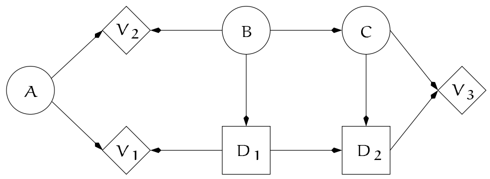

Figure 3.4: The algorithm Decision Bayes-ball marks the chance variable B as requisite for D_1.

between nodes with a common child and finally value nodes are removed together with the directions on the arcs. When eliminating a decision variable D the resulting adjacency set $\mathrm{adj}(D)$ is a subset of pred(D). This set of neighbours is characterized as the set of variables connected in the moral graph to D through a path with no intermediate variable in pred(D). As $\mathrm{adj}(D)$ contains all the information relevant for determining the optimal policy for D, it is a candidate for the required past. However $\mathrm{adj}(D)$ may contain variables irrelevant for D as can be seen from the influence diagram depicted in Figure 3.4; B is contained in the neighbouring set for D_1 as the elimination of A produces a fill-in edge between B and D_1.

So, neither Decision Bayes-ball nor the elimination method presented above is fine-grained enough to detect all independencies. The problem is that $\mathrm{adj}(D)$ may contain variables relevant for the maximum expected utility for a decision variable D; the maximum expected utility for D may cover utility functions having no influence on the optimal policy for D, see also (Fagiuoli and Zaffalon 1998; Zhang 1998). This means that we need to characterize and identify the utility functions on which the optimal policy for D depends:

Definition 3.3.7. The utility function ψ is *relevant* for D w.r.t. the admissible total order $<$ for I, if there exists two realizations $\mathcal{R}_1$ and $\mathcal{R}_2$ of I which only differ on ψ s.t. the optimal policies for D relative to $<$ are different in $\mathcal{R}_1$ and $\mathcal{R}_2$.

Thus, we need to determine the structural constraints necessary and sufficient for a utility function to be relevant for a decision variable, and based on this characterization we shall define the constraints necessary and sufficient for a chance variable to be significant for a given decision variable.

3.3.3 Relevant utility functions

The optimal policy for a decision variable D is based on the assumption that we always adhere to the maximum expected utility principle. Hence, if deciding on D can influence a future decision D′, then the utility functions relevant for D′ may be relevant for D also. From this observation together with the expression for the optimal policy for D (see equation 3.11), we present the following metarules. For notational convenience we shall sometimes treat uninstantiated decision variables as chance variables with an even prior distribution. Moreover, since a utility function is termed relevant w.r.t. an admissible total order we will mainly consider influence diagrams in the section.

Metarule 1. ψ_V is relevant for D in I if there is a realization of I s.t. D has an impact on the expected utility for ψ_V.

Metarule 2. ψ_V is relevant for D in I if there is a realization of I and a future decision D′ s.t. D has an impact on D′ and ψ_V is relevant for D′.

Metarule 3. If neither of the metarules above can be applied, then ψ_V is not relevant for D.

The following examples present a set of influence diagrams where we identify the utility functions relevant for a given decision variable. The properties relating to these examples will be generalized to arbitrary influence diagrams, which will serve as a basis for determining the structural constraints necessary for a utility function to be relevant for a given decision variable.

Example 3.3.4. Consider the influence diagram depicted in Figure 3.5, and assume that the conditional probability potentials are specified s.t. the state of a variable corresponds to the state of its parent.

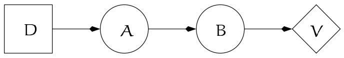

Figure 3.5: The figure represents an influence diagram, where the utility function ψ_V is relevant for the decision variable D.

It is easy to specify two realizations of ψ_V s.t. the optimal policies relative to those realizations differ, i.e., ψ_V is relevant for D. □

Now, assume an arbitrary influence diagram I in which there exists a directed path from a decision node D to a value node V in $\bar{I}$, and assume a realization $\mathcal{R}$ of I. Since the conditional probability potentials associated with the variables on the path from D to V can be specified s.t. deciding on D has an impact on the expected utility for ψ_V, it follows that ψ_V is relevant for D.

Rule 3.3.1. Let I be a partial influence diagram, and let $\bar{I}$ denote I without informational arcs. The utility function ψ_V is relevant for the decision variable D if there exists a directed path from D to V in $\bar{I}$ (see Figure 3.6).

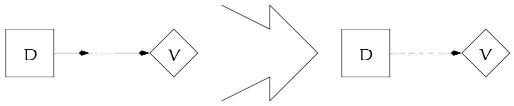

Figure 3.6: The figure illustrates Rule 3.3.1, where the directed path from the decision node D to the value node V is represented by $D \rightarrow \cdots \rightarrow V$ and the relevance of ψ_V w.r.t. D is represented by $D \dashrightarrow V$.

This rule is equivalent to Metarule 1, as can be seen from the mathematical expression corresponding to this metarule: ψ_V is relevant for D if $P(\mathrm{dom}(\psi_V)|D, \mathrm{pred}(D))$ is a function of D, where $\mathrm{dom}(\psi_V)$ is the chance variables in the domain of ψ_V (uninstantiated decision variables are treated as chance variables). The conditional probability potential $P(\mathrm{dom}(\psi_V)|\mathrm{pred}(D), D)$ is a function of D if D is d-connected to a variable $A \in \mathrm{dom}(\psi_V)$ given $\mathrm{pred}(D)$. However, this implies that there exists a directed path from D to V in $\bar{I}$. Conversely, if there exists a directed path from D to V in $\bar{I}$, then D is d-connected to a variable $A \in \mathrm{dom}(\psi_V)$ given $\mathrm{pred}(D)$.

The following examples illustrate, that in order to identify all the utility functions relevant for a given decision variable D, it is in general not sufficient only to consider those utility functions to which there exist a directed path from D in $\bar{I}$.

Example 3.3.5. When deciding on D_4 in the influence diagram depicted in Figure 3.7 we want to maximize ψ_{V_2}. As the decision variable D_2 is d-connected to C given $\mathrm{pred}(D_4)$, it follows that the decision made w.r.t. D_2 may change our belief in C (when deciding on D_4) and thereby influence D_4 through ψ_{V_2} (D_2 is required for D_4). Hence, ψ_{V_2} is relevant for D_2, which is also true for ψ_{V_1} as $D_2 \in \mathrm{dom}(\psi_{V_1})$. Moreover, knowledge of A may likewise change our belief in C when deciding on D_4, and since A is influenced by D_1 it follows that D_1 has an impact on D_2 since knowledge of D_1 can be taken into account when deciding on D_2 (D_1 is required for D_2). Thus, ψ_{V_2} is relevant for both D_1 and D_2 conveying that ψ_{V_1} is relevant for D_1 also.

This can also be seen by considering the utility function specified in table 3.1 together with the functions $P(C) = (0.5; 0.5)$, $\psi^1_{V_1}(D_2) = (0; 3)$ and $\psi^2_{V_1}(D_2) = (3; 0)$ (we assume that the state of A corresponds to the state of D_1 and that $P(B|D_2, A, C)$ is specified s.t. the state of C is revealed if the states of A, B and

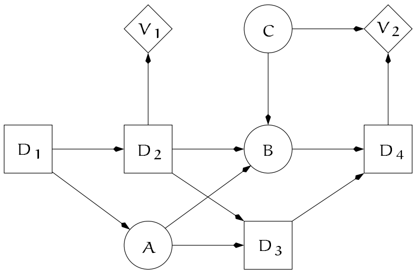

Figure 3.7: The figure represents an influence diagram where both ψ_{V_1} and ψ_{V_2} are relevant for D_1.

D_2 are the same, and no knowledge is gained on C if this is not the case).

$\psi_{V_2}(C, D_4)$	$C = c_1$	$C = c_2$
$D_4 = d_1$	10	0
$D_4 = d_2$	0	9

Table 3.1: The utility function $\psi_{V_2}(C, D_4)$.

These functions define two realizations of I that only differ on ψ_{V_1}, and when evaluating I w.r.t. these realizations we obtain two different optimal policies for D_1, i.e., $\delta_{D_1} = d_1$ if $\psi_{V_1} = \psi^1_{V_1}$ and $\delta_{D_1} = d_2$ if $\psi_{V_1} = \psi^2_{V_1}$. From these policies it can be seen that the utility function ψ_{V_1} influences the optimal policy for D_1; hence, ψ_{V_1} is relevant for D_1. □

The example above can be seen as an instance of Metarule 2. Consider an arbitrary influence diagram I and a realization of I in which the conditional probability potential for any intermediate variable in a converging connection is specified as in the example above. From this structure we may deduce that, if ψ_V is relevant for a future decision D' and D is required for D', then the utility function ψ_V is relevant for D; D has an impact an D'.

Example 3.3.6. When deciding on D_4 in the influence diagram depicted in Figure 3.8, we seek to maximize ψ_{V_2}. By the arguments given in the example above, it follows that both ψ_{V_1} and ψ_{V_2} are relevant for D_2. Additionally, knowledge of B may change our belief in E when deciding on D_4, and since B is influenced by A, which in turn is influenced by D_1, it follows that D_1 has an impact on D_2 (A is required for D_2). Thus, both ψ_{V_1} and ψ_{V_2} are relevant for D_1.

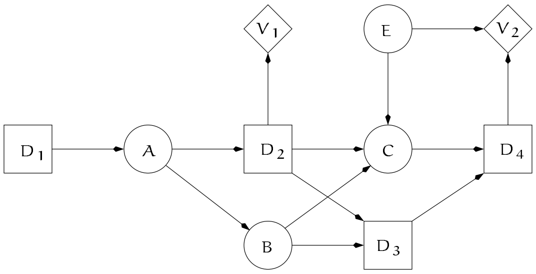

Figure 3.8: The figure represents an influence diagram, in which D_1 may influence the chance variable A required for D_2, indicating that both ψ_{V_1} and ψ_{V_2} are relevant for D_1.

This can also be seen by assuming the two realizations consisting of the utility functions $\psi^1_{V_1}(D_2) = (0;3)$ and $\psi^2_{V_1}(D_2) = (3;0)$ together with functions corresponding to the ones specified in Example 3.3.4 and Example 3.3.5. When evaluating I w.r.t. these realizations we obtain two optimal strategies which differ on D_1, i.e., $\delta_{D_1} = d_1$ if $\psi_{V_1} = \psi^1_{V_1}$ and $\delta_{D_1} = d_2$ if $\psi_{V_1} = \psi^2_{V_1}$. From these policies it can be seen that the utility function ψ_{V_1} may influence the optimal policy for D_1; thus, ψ_{V_1} is relevant for D_1. □

The structural properties relating to the example above can, as for the previous example, be generalized to an arbitrary influence diagram. Thus, based on the example above and the deductions made w.r.t. Example 3.3.5, we present the following rule.

Rule 3.3.2. Let I be a partial influence diagram and let $<$ be an admissible total order for I. The utility function ψ_V is relevant for the decision variable D w.r.t. $<$ if there exists a decision variable D' s.t.

i) $D < D'$ and ψ_V is relevant for D' w.r.t. $<$, and

ii) either

  a) D is required for D' w.r.t. $<$ or

  b) there exists a directed path in $\bar{I}$ from D to a chance variable $X \in \mathrm{pred}(D')$, and X is required for D' w.r.t. $<$.

The rule is illustrated in Figure 3.9.

This rule is equivalent to Metarule 2, since the rule covers all the cases where D can have an impact on a future decision D'; D has an impact on D' if and only if D is required for D' or D influences a variable which is required for D'.

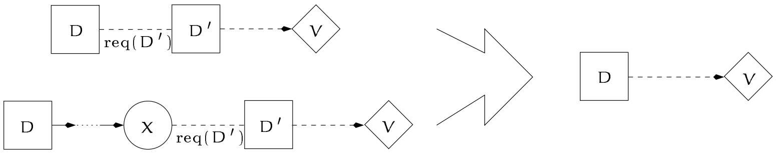

Figure 3.9: The figure illustrates Rule 3.3.2, where $\mathsf{D} \underset{\mathrm{req}(\mathsf{D}')}{-\!-\!-} \mathsf{D}'$ indicates that D is required for D'; similarly, $\mathsf{X} \underset{\mathrm{req}(\mathsf{D}')}{-\!-\!-} \mathsf{D}'$ indicates that X is required for D'.

Note that Rule 3.3.2 is not a complete structural rule as it refers to the term "required", which has not yet been characterized structurally. This is done in the following section, where it is also shown that Rule 3.3.1 and Rule 3.3.2 are sound and complete.

3.3.4 Required variables

Having established a method to identify the relevant utility functions for a decision variable, one might think that the required variables could be identified in the following way: Before constructing the moral graph, remove all utility functions not relevant for D and then eliminate the variables as described in Section 3.2. However, the resulting adjacency set $\mathrm{adj}(\mathsf{D})$ may still contain variables which are not required as can be seen from Figure 3.10; A is not required for D_1, but $\mathsf{A} \in \mathrm{adj}(\mathsf{D}_1)$ when D_1 is eliminated.

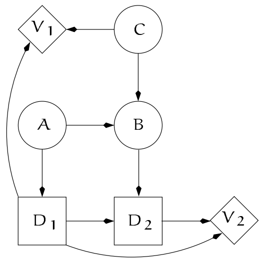

Figure 3.10: The figure illustrates an influence diagram, where the chance node A is not required for D_1, but $\mathsf{A} \in \mathrm{adj}(\mathsf{D}_1)$ when D_1 is eliminated.

A variable X is required for a decision variable D if X is observed before D and the state of X may influence the optimal policy for D. Since the optimal policy for D is dependent on the assumption that we always adhere to the maximum expected utility principle it follows that X is required for D if i) X has an impact on D, or ii) X has an impact on a future decision variable D' on which D also has an impact. Hence, analogously to the metarules specifying the utility functions being relevant for a decision variable, we present three metarules concerning the variables required for a given decision variable; according to the definition of a required variable we assume an admissible total order $<$ where a variable X occurs before a decision variable D under $<$.

Metarule 4. X is required for D if there is a realization s.t. when deciding on D the state of X has an impact on the expected utility for a utility function ψ relevant for D w.r.t. $<$.

Metarule 5. X is required for D if there is a realization and a future decision D' s.t. X has an impact on D', and there exists a utility function ψ_V relevant for both D and D' w.r.t. $<$.

Metarule 6. If neither of the metarules above can be applied, then X is not required for D.

The following examples present a set of partial influence diagrams in which some of the required variables are identified. The properties described by these examples will be generalized to arbitrary partial influence diagrams; in the examples we assume that chance variables with no parents are given an even prior distribution.

Example 3.3.7. Consider the partial influence diagram I depicted in Figure 3.11. The utility function ψ_{V_1} is relevant for D_1 for any admissible total ordering of I, and since ψ_{V_1} is functionally dependent on the chance variable A it follows that A is required for D_1 w.r.t. an admissible total ordering with $A < D_1$.

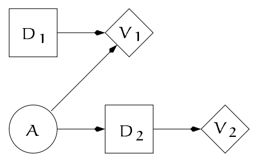

Figure 3.11: The figure depicts a partial influence diagram in which the chance variable A is required for D_1 if $A < D_1$.

□

The example above can be generalized to an arbitrary partial influence diagram I, assuming that I contains a decision variable D and a variable X. If there exists an admissible total ordering $<$ s.t. X occurs before D under $<$ and X is d-connected to a value node V given $\text{pred}(D)$, then X is required for D under $<$ if ψ_V is relevant for D w.r.t. $<$; we can specify a realization of I s.t. the state of X has an impact on the expected utility for ψ_V; note that informational arcs are disregarded when testing for d-connectivity.

Rule 3.3.3. Let I be a partial influence diagram and let $<$ be an admissible total order for I. The variable X is required for the decision variable D w.r.t. $<$ if $X < D$ and there exists a utility function ψ_V relevant for D w.r.t. $<$ s.t. X is d-connected to V given $\mathrm{pred}(D)$ (see Figure 3.12).

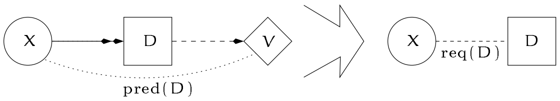

Figure 3.12: The figure illustrates Rule 3.3.3, where $X \twoheadrightarrow D$ represents $X < D$, and $X \cdot_{\mathrm{pred}(D)} \cdot V$ indicates that X is d-connected to V given $\mathrm{pred}(D)$.

This rule is equivalent to Metarule 4, as can be seen by expressing the metarule mathematically: X is required for D if $P(\mathrm{dom}(\psi_V)|D, \mathrm{pred}(D))$ is a function of X, and ψ_V is a utility function relevant for D. The probability potential $P(\mathrm{dom}(\psi_V)|D, \mathrm{pred}(D))$ is a function of X if and only if X is d-connected to V given $\mathrm{pred}(D)$.

The following examples show, that in order to identify all the variables required for a decision variable D, it is in general not sufficient only to consider the variables which directly influence the decision made w.r.t. D (see Metarule 5).

Example 3.3.8. In the partial influence diagram I depicted in Figure 3.13, the chance variable A is required for D_3 since ψ_V is relevant for D_3 and A is d-connected to V given $\mathrm{pred}(D_3)$. As ψ_V is relevant for D_1 also, it follows that if $D_1 < D_3$ then A and D_1 may both have an impact on D_3. Additionally, if A is observed prior to D_1 then the state of A can be taken into account when deciding on D_1, i.e., A is required for D_1 if $A < D_1$.

This can also be seen by considering the utility function specified by Table 3.2, and by assuming that $P(B|A, C)$ has the properties of $P(B|D_2, A, C)$ specified in Example 3.3.5.

$\psi_V(D_1, C, D_3)$	$C = c_1$	$C = c_2$
$D_3 = d_1$	$(10; 2)$	$(1; 0)$
$D_3 = d_2$	$(0; 1)$	$(2; 10)$

Table 3.2: The utility function $\psi_V(D_1, C, D_3)$.

The optimal policy for D_1 relative to $A < D_1 < D_2 < B < D_3 < C$ is given by

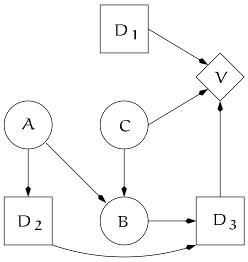

Figure 3.13: The figure depicts a partial influence diagram in which the variable A is required for D_1 if $A < D_1$.

$\delta_{D_1}(a_1) = d_2$ and $\delta_{D_1}(a_2) = d_1$ which indicate, that D_1 is dependent on the state of A (A is required for D_1). □

In the example above, the conditional probability potential for B is specified s.t. the state of C is revealed if the state of A corresponds to the state of B, and no knowledge is gained on C if the state of A does not correspond to the state of B. Furthermore, the utility function ψ_V (relevant for both D_1 and D_3) is specified s.t. the state of C and the decision made w.r.t. D_1 influences D_3. Thus, A is required for D_3 and therefore required for D_1 also.

Analogously to the previous examples, we may generalize this example by considering an arbitrary partial influence diagram I and an admissible total order for I, where a variable X occurs before a decision variable D; assume that X is required for a future decision D', which has a relevant utility function in common with D. By specifying the realization of I according to the example above, it follows that the state of X may influence the decision made w.r.t. D'. Moreover, since D and D' have a relevant utility function in common and X is required for D', we have that X is required for D also; the state of X must be taken into account when deciding on D.

Example 3.3.9. Consider the partial influence diagram I depicted in Figure 3.14. Observing the chance variable A may reveal the state of X, and when deciding on D_4 the observation of X may change our belief in the state of C. Since ψ_V is relevant for both D_1 and D_4 the decision made w.r.t. D_1 has an impact on D_4, and since the state of A may influence the decision made w.r.t. D_4, and thereby the optimal policy for D_1, it follows that the state of A is relevant when deciding on D_1. That is, A is required for D_1.

This can also be seen by considering a realization of I, where $P(X|A)$ is a deterministic function and $\psi_V(D_1, C, D_4)$ and $P(B|X, C)$ correspond to $\psi_V(D_1, C, D_3)$ and $P(A|D_1, B)$ respectively (see Example 3.3.5 and Table 3.2). When evaluating I w.r.t. $A < D_1 < D_2 < X < D_3 < B < D_4 < C$ the optimal policy for D_1 is

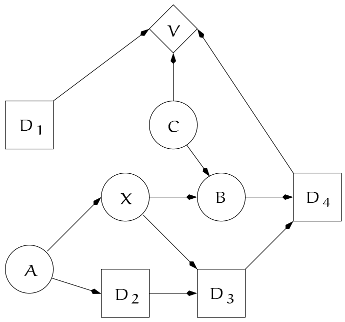

Figure 3.14: The figure depicts a partial influence diagram in which the chance variable A is required for D_1 if $A < D_1 < X$.

given by $\delta_{D_1}(a_1) = d_2$ and $\delta_{D_1}(a_2) = d_1$, i.e., A is required for D_1. $\square$

In the example above, the state of X is determined by the state of A, whereas the state of C is determined by the state of X and B, i.e., as for the previous examples, the state of C is revealed if the state of X corresponds to the state of B, whereas no knowledge is gained on C if the state of X does not correspond to the state of B. Now, consider an arbitrary partial influence diagram I, and let X denote a variable which occurs before the decision variable D under $<$, and assume that X is d-connected to a chance variable $Y \in \text{pred}(D')$ given $\text{pred}(D)$. If Y is required for D' ($D < D'$) and D' has a relevant utility function in common with D w.r.t. $<$, then we can specify a realization, as described in the example above, s.t. the optimal policy for D is dependent on the state of X. That is, X is required for D w.r.t. $<$.

Rule 3.3.4. Let I be a partial influence diagram and let D be a decision variable in I. Then the variable X is required for D w.r.t. the admissible total order $<$ (where $X < D$) if there exists a utility function ψ_V relevant for D w.r.t. $<$ and:

i) there exists a decision variable D' ($D < D'$) s.t. ψ_V is relevant for D' w.r.t. $<$.

ii) X is required for D' w.r.t. $<$ or X is d-connected to a chance variable $Y \in \text{pred}(D') \setminus \text{pred}(D)$ given $\text{pred}(D)$ and Y is required for D' w.r.t. $<$.

The rule is illustrated in Figure 3.15.

This rule is equivalent to Metarule 5, since the observation of X can have an impact on a future decision D' if and only if X is required for D' or X influences a variable required for D'.

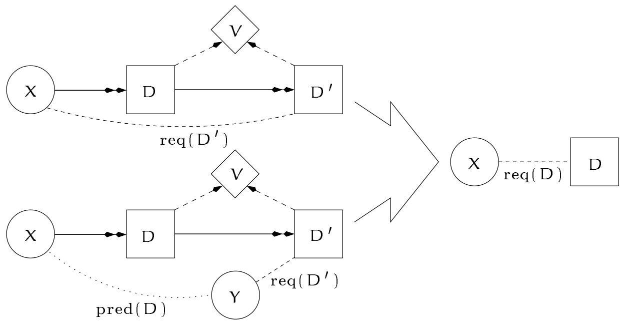

Figure 3.15: The figure illustrates Rule 3.3.4.

The rules 3.3.2 and 3.3.4 represent a set of simultaneous recursive structural constraints. The recursion terminates because it moves forward in the temporal ordering for each "call".

Based on the previous rules we present the following rule characterizing the chance variables being significant for a given decision variable; this rule is apparent from Rule 3.3.3 and Rule 3.3.4.

Rule 3.3.5. Let I be a partial influence diagram and let D be a decision variable in I incompatible with a chance variable A. Then A is significant for D if there exists a utility function ψ_{V} relevant for D w.r.t. an admissible total ordering $<$, where A occurs immediately before D, and:

i) A is d-connected to V given $\mathrm{pred}(\mathsf{D})$ or

ii) there exists a decision variable D' ($\mathsf{D} < \mathsf{D}'$) s.t. ψ_{V} is relevant for D' w.r.t. $<$, and:

   a) A is required for D' w.r.t. $<$ or

   b) A is d-connected to a chance variable $\mathsf{X} \in \mathrm{pred}(\mathsf{D}')\backslash\mathrm{pred}(\mathsf{D})$ given $\mathrm{pred}(\mathsf{D})$ and X is required for D' w.r.t. $<$.

Before giving a proof of soundness and completeness for the rules above we shall briefly consider another algorithm for identifying required variables; this algorithm was found independently by Shachter (1999). The algorithm is based on the notion of a chance variable policy for a decision variable (see Section 3.2 and Nilsson and Jensen (1998) for further discussion on chance variable policies):

Definition 3.3.8. Let D be a decision variable and let $\delta_{\mathsf{D}}(\mathrm{req}(\mathsf{D}))$ be an optimal policy for D, where $\mathrm{req}(\mathsf{D})$ are the variables required for D. The conditional probability potential $P(D|\mathrm{req}(D))$ defined as:

$$P(d|\bar{r}) = \begin{cases} 1 & \text{if } \delta_D(\bar{r}) = d, \\ 0 & \text{otherwise,} \end{cases}$$

is said to be the *chance variable policy* for D.

The algorithm by Shachter (1999) works by visiting each of the decision variables in reverse temporal order. When a decision variable is visited the variables required for that decision are identified and the decision variable is replaced by a chance variable.

Algorithm 3.3.1. Let I be an influence diagram and let $D_1, D_2, \ldots, D_n$ be the decision variables in I ordered by index. To determine the variables required for D_i ($\forall 1 \leq i \leq n$) do:

1. Set $i := n$.
2. **For** each decision variable not considered ($i > 0$)
   a) Let $\mathcal{V}_i$ be the set of value nodes to which there exists a directed path from D_i in I.
   b) Let $\mathrm{req}(D_i)$ be a set of nodes s.t. $X \in \mathrm{req}(D_i)$ if and only if $X \in \mathrm{pred}(D_i)$ is d-connected to a node in $\mathcal{V}_i$ given $\mathrm{pred}(D_i) \setminus \{X\}$ ($\mathrm{req}(D_i)$ is the set of variables required for D_i).
   c) Replace D_i with a chance variable representation of the policy for D_i.
   d) Set $i := i - 1$.

□

Shachter (1999) based Algorithm 3.3.1 on the *Bayes-Ball* algorithm (Shachter 1998) for detecting D-separation (Geiger et al. 1990b), see also Section 2.4. However, the chance variable policies inserted during the transformation does not introduce any independencies relevant for our context so we only need to consider d-separation.

The time complexity of Algorithm 3.3.1 is based on the time complexity of the algorithm for detecting d-separation. For instance, if Algorithm 2.4.2 by Geiger et al. (1990a) is used, then the time complexity is $O(|\mathcal{U}_D| \cdot |\mathcal{E}|)$ as Algorithm 2.4.2 runs in time $O(|\mathcal{E}|)$.

Shachter (1999) does not prove the correctness of the algorithm hence, the remainder of this section is devoted to this proof and the proof of the rules given previously.

Lemma 3.3.1. Let I be an influence diagram and let $D_1, D_2, \ldots, D_n$ be the decision variables in I ordered by index. Then the utility function ψ_V is relevant for D_n if and only if there exists a directed path from D_n to the value node V in I.

Proof. By Equation 3.7 the optimal policy for D_n is given by:

$$\begin{aligned}
\delta_{D_n} &= \arg\max_{D_n} \sum_{\mathcal{C}_n} P(\mathcal{C}_n|\mathrm{pred}(D_n), D_n) \sum_{V \in \mathcal{U}_V} \psi_V(pa(V)) \\
&= \arg\max_{D_n} \sum_{\mathcal{C}_n} P(\mathcal{C}_n|\mathrm{pred}(D_n), D_n)\psi_V(pa(V)) \\
&\qquad + \sum_{\mathcal{C}_n} P(\mathcal{C}_n|\mathrm{pred}(D_n), D_n) \sum_{V' \in \mathcal{U}_V \setminus \{V\}} \psi_{V'}(pa(V')).
\end{aligned}$$

Since

$$\begin{aligned}
&\sum_{\mathcal{C}_n} P(\mathcal{C}_n|\mathrm{pred}(D_n), D_n)\psi_V(pa(V)) \\
&\qquad = \sum_{\mathcal{C}_n \cap pa(V)} \psi_V(pa(V)) \sum_{\mathcal{C}_n \setminus pa(V)} P(\mathcal{C}_n|\mathrm{pred}(D_n), D_n) \\
&\qquad = \sum_{\mathcal{C}_n \cap pa(V)} \psi_V(pa(V)) P(\mathcal{C}_n \cap pa(V)|\mathrm{pred}(D_n), D_n)
\end{aligned} \tag{3.18}$$

we have that ψ_V is relevant for D_n if and only if D_n is d-connected to a variable $X \in pa(V) \cap \mathcal{C}_n$ given $\mathrm{pred}(D_n)$ (or $D_n \in pa(V)$); otherwise, we have $P(\mathcal{C}_n \cap pa(V)|\mathrm{pred}(D_n), D_n) = P(\mathcal{C}_n \cap pa(V)|\mathrm{pred}(D_n))$. However, this implies that D_n is d-connected to V given $\mathrm{pred}(D_n)$ also, which is the case if and only if there exists a directed path from D_n to V in I. If the path is not directed it would contain a converging connection with an intermediate variable Y, and since D_n is the last decision variable in I neither Y nor any of its descendants can be in $\mathrm{pred}(D_n)$. Thus, if the path is not directed, D_n would be d-separated from V given $\mathrm{pred}(D_n)$. ☐

Lemma 3.3.2. Let I be an influence diagram and let $D_1, D_2, \ldots, D_n$ be the decision variables in I ordered by index. Then the variable $X \in \mathrm{pred}(D_n)$ is required for D_n if and only if there exists a utility function ψ_V relevant for D_n s.t. X is d-connected to V given $\mathrm{pred}(D_n)$.

Proof. The proof is trivial if no utility functions are relevant for D_n. If there exists at least one utility function relevant for D_n, then the proof follows the proof of Lemma 3.3.1 except that we consider X instead of D_n. ☐

In what follows we will use I_{D_i} to denote the influence diagram produced from I by iteratively replacing the decision variables succeeding D_i with their chance variable policies.

Lemma 3.3.3. Let I be an influence diagram and let $D_1, D_2, \ldots, D_n$ be the decision variables in I ordered by index. Then $\delta_{D_i} = \delta'_{D_i}, \forall 1 \leq i \leq n-1$, where δ_{D_i} and δ'_{D_i} are the optimal policies for D_i in I and $I_{D_{n-1}}$, respectively.

Proof. By Equation 3.4 it is sufficient to show that the maximum expected utility function ρ_{D_n} for decision D_n in I is equivalent to the utility function obtained by marginalizing out the variables $\mathcal{C}_n \cup \{D_n\}$ in $I_{D_{n-1}}$.

From Equation 3.8 we have that the maximum expected utility for decision D_n is given by:

$$\rho_{D_n}(\text{pred}(D_n)) = \max_{D_n} \sum_{\mathcal{C}_n} P(\mathcal{C}_n | \mathcal{C}_0, D_1, \ldots, \mathcal{C}_{n-1}, D_n) \sum_{V \in \mathcal{U}_V} \psi(pa(V)).$$

If $P(D_n|\text{pred}(D_n))$ is the conditional probability potential representing the optimal policy for D_n, then by Equation 3.3 we have:

$$\rho_{D_n}(\text{pred}(D_n)) = \sum_{D_n} \sum_{\mathcal{C}_n} P(D_n|\text{pred}(D_n)) P(\mathcal{C}_n|\text{pred}(D_n), D_n) \sum_{V \in \mathcal{U}_V} \psi(pa(V)). \tag{3.20}$$

As δ_{D_n} is a constant function over the variables $\text{pred}(D_n) \backslash \text{req}(D_n)$ we have that $P(D_n|\text{pred}(D_n)) = P(D_n|\text{req}(D_n))$. Thus, Equation 3.20 reduces to:

$$\rho_{D_n}(\text{pred}(D_n)) = \sum_{D_n} \sum_{\mathcal{C}_n} P(D_n|\text{req}(D_n)) P(\mathcal{C}_n|\text{pred}(D_n), D_n) \sum_{V \in \mathcal{U}_V} \psi(pa(V)),$$

which is the utility function obtained by marginalizing out the variables $\mathcal{C}_n \cup \{D_n\}$ in $I_{D_{n-1}}$. □

Based on the lemmas above, we have the following theorem.

Theorem 3.3.3. Let I be an influence diagram and let D be a decision variable in I. If Algorithm 3.3.1 is invoked on I, then:

1. there is a directed path from D to a value node V in I_D if and only if ψ_V is relevant for D.
2. $X \in \text{req}(D)$ if and only if X is required for D.

Proof. Assume that no decision variable succeed D under $\prec$. Then the utility function ψ_V is relevant for D if and only if there exists a directed path from D to V (see Lemma 3.3.1). Analogously, the variable $X \in \text{pred}(D)$ is required for

D, if and only if X is d-connected to a value node V given $\text{pred}(\mathsf{D})$, where ψ_{V} is relevant for D (see Lemma 3.3.2). If D is succeeded by other decision variables, then we can apply Lemma 3.3.3 iteratively until D is the last decision variable hence, the "if" part follows immediately. The "only if" part follows by observing that the chance variable representation of an optimal policy for a decision variable D_i contains exactly the variable required for D_i, i.e., there does not exist a variable $\mathsf{Y} \in \text{req}(\mathsf{D}_i)$ s.t. $\mathsf{P}(\mathsf{D}_i|\text{req}(\mathsf{D}_i)) = \mathsf{P}(\mathsf{D}_i|\text{req}(\mathsf{D}_i)\setminus\{\mathsf{Y}\})$; substituting a decision variables with its chance variable policy does not introduce any "false" structural dependences. □

We now have the following corollaries as a consequence of Theorem 3.3.3; we shall use $\text{req}(\mathsf{D})^<$ to denote the set $\text{req}(\mathsf{D})$ identified by Algorithm 3.3.1 when the algorithm is invoked on a partial influence diagram given the total ordering $<$.

Corollary 3.3.1. Let $\mathcal{I}$ be a partial influence diagram and let D be a decision variable in $\mathcal{I}$. If Algorithm 3.3.1 is invoked on $\mathcal{I}$ w.r.t. an admissible total order $<$, then

- there is a directed path from D to a value node V in $\mathcal{I}_{\mathsf{D}}$ if and only if ψ_{V} is relevant for D w.r.t. $<$.
- $\mathsf{X} \in \text{req}(\mathsf{D})^<$ if and only if X is required for D w.r.t. $<$.

Corollary 3.3.2. Let $\mathcal{I}$ be a partial influence diagram and let D be a decision variable in $\mathcal{I}$ incompatible with a chance variable A. Then A is significant for D if and only if there exists an admissible total order $<$ s.t. A occurs immediately before D under $<$ and $\mathsf{A} \in \text{req}(\mathsf{D})^<$ when Algorithm 3.3.1 is invoked on $\mathcal{I}$ w.r.t. $<$.

Theorem 3.3.2 and Corollary 3.3.2 constitute an operational characterization of temporal independence. The following corollaries establish the soundness and completeness of the the rules given previously.

Corollary 3.3.3. Let $\mathcal{I}$ be a partial influence diagram and let D_i be a decision variable in $\mathcal{I}$. Then the utility function ψ_{V} is relevant for D_i w.r.t. the admissible total order $<$ if and only if Rule 3.3.1 or Rule 3.3.2 can be applied.

Proof. The proof exploits that there is a one to one correspondance between the rules and Algorithm 3.3.1.

Let D'_j denote the chance variable in $\mathcal{I}_{\mathsf{D}_i}$ corresponding to decision variable D_j in $\mathcal{I}$, $\forall j > i$; it is sufficient to consider the case where $\mathcal{I}$ is an influence diagram.

If ψ_{V} is relevant for D_i, then there exists a directed path from D_i to V in $\mathcal{I}_{\mathsf{D}_i}$ (Theorem 3.3.3). If the path does not contain a chance variable D'_j, then there is a path from D_i to V in $\bar{I}$ (Rule 3.3.1). On the other hand, if the path contains such a chance variable consider the D'_j closest to D_i on the path. From the construction of I_{D_i}, there also exists a directed path from D_j to V in I_{D_j}; hence, ψ_V is relevant for D_j. Moreover, from the construction of I_{D_i} we also have that either i) $D_i \in pa(D'_j)$ or ii) there exists a directed path from D_i to a variable $Y \in pa(D'_j)$ in $\bar{I}$. In case (i) D_i is required for D_j and in case (ii) Y is required for D_j; both of these cases are covered by Rule 3.3.2.

Conversely, if Rule 3.3.1 can be applied, then there exists a directed path from D_i to a value node V in $\bar{I}$. This path also exists in I_{D_i}; hence, ψ_V is relevant for D_i. Assume that Rule 3.3.2 can be applied, and assume there exists a future decision variable D_j for which the utility function ψ_V is relevant. If D_i is required for D_j, then there exists a directed path from D_i to V in I_{D_i} (since $D_i \in pa(D'_j)$) hence ψ_V is relevant for D_i. Similarly, if there is a directed path from D_i to a chance variable $X \in \text{req}(D_j)$ in $\bar{I}$, then there also exists a directed path from D_i to V in I_{D_i}; hence, ψ_V is relevant for D_i. □

Corollary 3.3.4. Let I be a partial influence diagram and let D_i be a decision variable in I. Then the variable X is required for D_i w.r.t. the admissible total order $<$ if and only if Rule 3.3.3 or Rule 3.3.4 can be applied.

Proof. The proof follows the proof the Corollary 3.3.3. □

Corollary 3.3.5. Let I be a partial influence diagram and let D_i be a decision variable incompatible with a chance variable A in I. Then A is significant for D_i if and only if Rule 3.3.5 can be applied.

3.4 Algorithms

To determine whether or not a partial influence diagram is well-defined we should in principle investigate all admissible total orderings $<$ with a pair of incompatible neighbouring variables of opposite type. However, the set of admissible orderings to investigate can be reduced substantially. E.g. if (A, D) is an incompatible pair, then the ordering of the predecessors of A and D is irrelevant. Also, if all pairs of incompatible successor nodes have been investigated and found commutable irrelevant, then we can take any admissible order of the successors of A and D. Thus, we start off with the pairs of incompatible variables with highest index and work ourself backwards in the partial temporal ordering. In this section we propose an algorithm based on this principle to determine whether or not a partial influence diagram defines a decision scenario.

The following algorithm is used by Algorithm 3.4.2 to determine the subsets of variables to investigate according to the description above. Given a set of decision variables and a partial ordering, the algorithm iteratively removes those decision variables which are ordered w.r.t. all the other decision variables currently in the set (an informal description of the algorithm is given below).

Algorithm 3.4.1. Let $\mathcal{U}'_{\mathrm{D}}$ be a set of decision variables and let $\prec$ be a partial order over $\mathcal{U}'_{\mathrm{D}}$. The smallest set of mutually incompatible nodes with the same set of informational predecessors and successors can be determined as follows:

1. Construct the table:
$$\mathtt{precedence}[\mathrm{D}, \mathrm{D}'] = \begin{cases} 1 & \text{if } \mathrm{D} \prec \mathrm{D}' \text{ or } \mathrm{D} = \mathrm{D}' \\ 0 & \text{otherwise.} \end{cases}$$

2. **For** each $\mathrm{D}' \in \mathcal{U}'_{\mathrm{D}}$
   Set $\#_{\mathrm{D}'} := \sum_{\mathrm{D} \in \mathcal{U}'_{\mathrm{D}}} \mathtt{precedence}[\mathrm{D}, \mathrm{D}']$.
3. **Repeat**
   For each $\mathrm{D} \in \mathcal{U}'_{\mathrm{D}}$
   If $\#_{\mathrm{D}} = |\mathcal{U}'_{\mathrm{D}}|$ **then**
   Set $\mathcal{U}'_{\mathrm{D}} := \mathcal{U}'_{\mathrm{D}} \setminus \{\mathrm{D}\}$.
   Until no such D exists.
4. **For** each $\mathrm{D} \in \mathcal{U}'_{\mathrm{D}}$
   If $\mathtt{precedence}[\mathrm{D}, \mathrm{D}'] = 0$ $(\forall \mathrm{D}' \neq \mathrm{D})$ **then**
   Set $\mathcal{N} := \mathcal{N} \cup \{\mathrm{D}\}$.
5. **For** each $\mathrm{D} \in \mathcal{U}'_{\mathrm{D}} \setminus \mathcal{N}$
   a) Determine $f(\mathrm{D}) = \prod_{\mathrm{D}' \in \mathcal{N}} \mathtt{precedence}[\mathrm{D}, \mathrm{D}']$.
   b) **If** $f(\mathrm{D}) = 0$ **then**
   Set $\mathcal{N} := \mathcal{N} \cup \{\mathrm{D}\}$.
6. Return $\mathcal{N}$.

□

The value $\#_{\mathrm{D}'}$ calculated in step (2) is the number variables preceding (and including) D'. If $\#_{\mathrm{D}'} = |\mathcal{U}'_{\mathrm{D}}|$, then all variables in $\mathcal{U}'_{\mathrm{D}}$ precede D'. Hence we remove D' from $\mathcal{U}'_{\mathrm{D}}$ (step (3)) since we are only interested in sets of incompatible variables; this is done iteratively until no more variables can be removed. In step (4) we determine the set $\mathcal{N}$ of incompatible variables which have no successors in the reduced set $\mathcal{U}'_{\mathrm{D}}$. Step (5) identifies the set of variables which precede at least one variable in $\mathcal{N}$ but not all of them; $f(\mathrm{D}) = 0$ if there exists a variable $\mathrm{D}' \in \mathcal{N}$ which does not succeed D.

Algorithm 3.4.2. Let $\mathrm{I} = (\mathcal{U}, \mathcal{E})$ be a partial influence diagram which induces the partial order $\prec$ and let $\mathcal{U}_{\mathrm{D}}$ be the decision variables in I. To determine whether or not I defines a decision scenario do:

1. Set $\mathcal{T} := \mathcal{U}_D$.

2. **Repeat**

   a) Invoke Algorithm 3.4.1 on $(\mathcal{T}, \prec)$ and let $\mathcal{N}$ be the set of decision variables returned.

   b) Set $\mathcal{M} := \mathcal{N} \cup (\cup_{D \in \mathcal{N}} pa(D) \setminus \cap_{D \in \mathcal{N}} pa(D))$.

   c) **For** each admissible ordering of $\mathcal{M}$
      For each decision variable D in $\mathcal{M}$
         If there exists a chance variable significant for D **then**
            I does not define a decision scenario.

   d) Set $\mathcal{T} := \mathcal{T} \setminus \mathcal{N}$.

   Until $\mathcal{T} = \emptyset$.

3. I defines a decision scenario.

□

Example 3.4.1. Consider the partial influence diagram I depicted in Figure 3.3. In order to determine whether or not I defines a decision scenario we apply the algorithm above which initially invokes Algorithm 3.4.1 on $\mathcal{U}_D = \{D_1, D_2, D_3, D_4\}$.

In step (1) and (2) of Algorithm 3.4.1 we construct the precedence table illustrated in Table 3.3.

$\preceq$	D_1	D_2	D_3	D_4
D_1	1	1	0	1
D_2	0	1	0	0
D_3	0	0	1	0
D_4	0	0	0	1
$\#_D$	1	2	1	2

Table 3.3: The precedence table constructed in step (1) and (2) of Algorithm 3.4.1.

No decision variables are removed in step (3) since $\mathcal{U}_D$ does not contain a decision variable succeeding all the other variables. The set of incompatible decision variables with no successors is given by $\mathcal{N} = \{D_2, D_3, D_4\}$ (step (4)), and in step (5) the set $\mathcal{N}$ is extended with D_1 since $f(D_1) = 0$. Finally $\mathcal{N} = \{D_1, D_2, D_3, D_4\}$ is returned.

In step (2b) of Algorithm 3.4.2 we determine the set $\mathcal{M}$:

$$\begin{aligned}\mathcal{M} &= \mathcal{N} \cup (\cup_{D\in\mathcal{N}} \mathrm{pa}(D) \setminus \cap_{D\in\mathcal{N}} \mathrm{pa}(D)) = \{D_1, D_2, D_3, D_4\} \cup (\{B, E, F, G\} \setminus \emptyset) \\ &= \{B, D_1, E, F, D_2, G, D_4, D_3\},\end{aligned} \tag{3.22}$$

which is the set of variables to be investigated in step (2c).

Consider the admissible ordering $B < D_3 < D_1 < E < F < G < D_2 < D_4$. The relevant utility functions for decision variable D_3 w.r.t. this ordering can be determined from Rule 3.3.1 and Rule 3.3.2, and consists of all the utility functions. Now, by Rule 3.3.5 the chance variable B is significant for D_3; hence, I does not define a decision scenario.

To resolve this ambiguity, assume there exist an arc from F to D_3; F is observed immediately before D_3 thereby rendering $B < D_3 < D_1 < E < F < G < D_2 < D_4$ inadmissible since we have a directed path from D_1 to D_3. By running Algorithm 3.4.2 we get $\mathcal{M} = \{E, F, D_2, G, D_4, D_3\}$ where $F < D_3 < E < G < D_2 < D_4$ is an admissible ordering. However, this implies that G is significant for D_2. In order to compensate for this ambiguity we can insert an informational arc from D_2 to D_4 thereby producing a partial influence diagram which defines a decision scenario. □

Having presented an algorithm for investigating partial influence diagrams we would like to answer the following question: Given an ambiguous partial influence diagram, what is the optimal insertion of *fill-ins* making the partial influence diagram unambiguous? As implied by the example above there does not necessarily exist a unique solution to this problem. However, an obvious candidate is to insert fill-ins producing a well-defined partial influence diagram where the maximum expected utility is greater than (or equal to) the maximum expected utility for any other well-defined partial influence diagram which can be generated from the original one (for any realization of the partial influence diagram). This candidate is easily constructed since we only have to insert fill-ins s.t. chance variables are observed as early as possible in the decision process. For instance, in Figure 3.3 the fill-ins $((D_2, D_3), (G, D_2))$ would produce a well-defined partial influence diagram having a maximum expected utility greater than (or equal to) that of any other well-defined partial influence diagram produced from the partial influence diagram in Figure 3.3 by inserting fill-ins (see Figure 3.16). Note that we might have inserted the arc (F, D_3) instead of (D_2, D_3), thereby also obtaining a well-defined partial influence diagram.

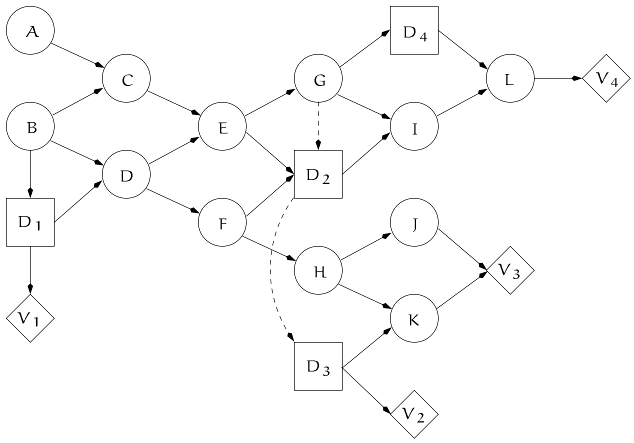

Figure 3.16: The partial influence diagram depicted in Figure 3.3 after fill-ins (drawn as dashed arcs) have been inserted, thereby making the partial influence diagram unambiguous.

3.5 Discussion

The influence diagram framework is a popular tool for representing and solving decision problems. However, the influence diagram requires a linear temporal ordering on the decisions even though the structure of a decision problem often renders certain decisions independent.

In this chapter we have defined the partial influence diagram as a generalization of the traditional influence diagram by allowing a partial ordering of the decision variables. Because the solution to a decision problem may be dependent on the temporal ordering of the decisions, we specified the class of partial influence diagrams whose solution is independent of the legal elimination orderings, i.e., the class of partial influence diagrams that represents a well-defined decision problem. We presented two characterizations of this class; one based on an iterative transformation of the original influence diagram (Shachter 1999), and the other based on a set of rules that works directly on the graphical structure of the model. These characterizations formed the basis for an algorithm for determining whether or not a partial influence diagram is well-defined.

Chapter 4

Asymmetric influence diagrams

This chapter deals with the representation of asymmetric Bayesian decision problems. We present a formal framework, termed *asymmetric influence diagrams*, that is based on partial influence diagrams and allows an efficient representation of asymmetric decision problems. As opposed to existing frameworks, the asymmetric influence diagram primarily encodes asymmetry at the qualitative level and it can therefore be read directly from the model.

4.1 Introduction

The power of an influence diagram, both as an analysis tool and as a communication tool, lies in its ability to concisely and precisely describe the structure of a decision problem (Smith et al. 1993). However, influence diagrams cannot efficiently represent the so-called asymmetric decision problems; decision problems are usually asymmetric in the sense that the set of possible outcomes of a chance variable may vary depending on the conditioning states, and the set of legitimate decision options of a decision variable may vary depending on the different information states (Qi et al. 1994).

To represent an asymmetric decision problem as an influence diagram, the problem must be "symmetrized" by adding artificial states and by assuming degenerate probability distributions and/or utility functions (Smith et al. 1993). Unfortunately, this "symmetrization" has several drawbacks. First of all, adding artificial states tends to obscure the structure of the decision problem as the possible scenarios in the decision problem are no longer concisely represented.

This also implies that the set of information states for a decision variable is increased, and many of these new configurations will be impossible, i.e., they have zero probability. Consequently, traditional evaluation algorithms may perform redundant calculations, as they compute an optimal policy for each information state, see e.g. (Shachter 1986; Shenoy 1992; Ndilikilikesha 1994; Jensen et al. 1994). Secondly, if the legitimate decision options for a decision variable vary w.r.t. the different informational states, we need degenerate utility functions having large negative values associated with the illegal decision options; conventional evaluation algorithms determine an optimal policy for a decision variable based on its entire state space.

Various frameworks have been proposed as alternatives to the influence diagram when dealing with asymmetric decision problems. Covaliu and Oliver (1995) extend the influence diagram with another diagram, called a *sequential decision diagram*. The sequential decision diagram is used for describing the asymmetric structure of the problem, as complementary to the influence diagram which is used for specifying the probability model; the functional and numerical information from these two diagrams are combined in a so-called *formulation table* similar to that of Kirkwood (1993). Smith et al. (1993) introduce the notion of *distribution trees* within the framework of influence diagrams. The use of distribution trees allows the possible outcomes of an observation to be specified, as well as the legitimate decision options for a decision variable, see also (Qi et al. 1994). However, as the distribution trees are not part of the influence diagram, the structure of the decision problem cannot be deduced directly from the model. Moreover, the sequence of decisions and observations is predetermined, i.e., previous observations and decisions cannot influence the temporal order of future observations and decisions. Finally, distribution trees have a tendency of creating large conditionals during the evaluation since they encode both numeric information and information about asymmetry. To overcome this problem, Shenoy (2000) presents the *asymmetric valuation network* as an extension of the *valuation network* for modeling symmetric decision problems (Shenoy 1992). The asymmetric valuation network uses *indicator functions* to encode asymmetry, thereby separating it from the numeric information. However, asymmetry is still not represented directly in the model and, as in (Smith et al. 1993), the sequence of observations and decisions is predetermined. Further details and comparisons of these methods can be found in (Bielza and Shenoy 1999).

In this chapter we present the *asymmetric influence diagram* framework for representing asymmetric decision problems. The asymmetric influence diagram is based on the partial influence diagram, and encodes structural asymmetry at the qualitative level through the use of labels associated with nodes and informational arcs; structural asymmetry has to do with the occurrence of variables in different scenarios as opposed to functional asymmetry which has to do with the possible outcomes/decision options of the variables. As a modeling language, the syntactical rules of the asymmetric influence diagram allow decision problems to be described in an easy and concise manner. Furthermore, as we shall see in Chapter 6, its semantic specification supports an efficient evaluation algorithm.

4.2 Preliminaries

Decision trees constitute the traditional framework for representing and solving decision problems, see (von Neumann and Morgenstern 1944) and (Raiffa 1968). A decision tree is a rooted tree $\mathsf{T} = (\mathcal{U}, \mathcal{E})$, where non-leaf nodes are either decision nodes or chance nodes, and the leaves are value nodes. The arcs in a decision tree are associated with labels: Arcs from a decision node are labeled with its legitimate decision options, and arcs from a chance node are labeled with its possible outcomes.

The past of a node X (denoted $\mathsf{past}(\mathsf{X})$) is the configuration specified by the labels associated with the arcs on the path from the root to X; if X is a value node, then $\mathsf{past}(\mathsf{X})$ is said to be a *scenario*. Similar to influence diagrams we assume no-forgetting, i.e., when taking decision D, the decision maker remembers all previous decisions and observations ($\mathsf{past}(\mathsf{D})$).

The quantitative part of a decision tree consists of probabilities and utilities. Each arc from a chance node A is associated with a probability $\mathsf{P}(\mathsf{A} = \mathsf{a} | \mathsf{past}(\mathsf{A}))$, where $\mathsf{A} = \mathsf{a}$ is the label of the arc in question. Thus, the probabilities associated with the arcs from a chance node specifies the probabilities of the achievable events conditioned on the events and decisions belonging to the path from the root to A. Finally, each value node V is associated with the utility $\psi_{\mathsf{V}}(\mathsf{past}(\mathsf{V}))$ of the scenario $\mathsf{past}(\mathsf{V})$.

Note that if only a causal model of the probabilities is available we may need to compute the probabilities in the decision tree using Bayes theorem. Alternatively, we can use the decision tree to encode the structure and the utilities of the decision problem, and then use a set of Bayesian networks for calculating the probabilities; this approach is somewhat similar to that of Call and Miller (1990) who combine decision trees with influence diagrams. In the remainder of this dissertation we assume that a decision tree only encodes the possible scenarios of a decision problem.

The decision tree allows a natural representation of asymmetric decision problems, but this is also one of its main weaknesses; representing each scenario explicitly causes the size of the decision tree to grow exponentially in the number of variables. Notice that the size of the decision tree can be reduced by collapsing identical subtrees, a procedure also know as *coalescence* (Olmsted 1983). However, a prerequisite for automated coalescence is that the complete (uncoalesced) decision tree is constructed.

In what follows we give two examples of asymmetric decision problems represented as decision trees; these examples will be used throughout this chapter to illustrate the proposed framework. The first example is the used car buyer problem by Howard (1962), and the second example is a modified version of the reactor problem; a statement of the original problem was given by Covaliu and Oliver (1995), and the modified version was proposed by Bielza and Shenoy (1999).

Example 4.2.1. [The used car buyer problem] Joe is considering buying (B) a used car for \$1000; the market price of similar cars with no defects is \$1100. Joe is uncertain whether this particular car (S) is a "peach" (p) or a "lemon" (l). Out of the ten major subsystems in the car, a peach has a serious defect in only one subsystem, whereas a lemon has a serious defect in six subsystems; the cost of repairing one defect is \$40 and the cost of repairing six defects is \$200.

For an additional \$60, Joe can buy the car with an "anti-lemon guarantee" (g); the anti-lemon guarantee will cover the full repair cost if the car is a lemon and cover half the repair cost otherwise.

Before buying the car, Joe can decide (T_1) to have the car examined by a mechanic. The mechanic offers three decision options: Test the steering subsystem alone at a cost of \$9 (t_1); test the fuel and electrical subsystems for a total cost of \$13 (t_2); do a two-test sequence in which Joe can authorize a second test T_2 after the result of the first test T_1 is known (t_3). In the latter case, the mechanic will first test the transmission subsystems at a cost of \$10 and report the result to Joe. If Joe approves (c), the mechanic will proceed to test the differential subsystems at an additional cost of \$4. All tests are guaranteed to find a defect in the subsystems if a defect exists.

The coalesced decision tree representation of the used car buyer problem is illustrated in Figure 4.1[1], and the Bayesian network used for calculating the probabilities is depicted in Figure 4.2; $P(R_1|S)$ and $P(R_2|S, R_1)$ are given in Table 4.1, and $P(S) = (0.8(p); 0.2(l))$.

$P(R_1\|S)$	$R_1 = d_1$	$R_1 = \neg d_1$
$S = p$	0.1	0.9
$S = l$	0.6	0.4

$P(R_2\|S, R_1)$	$R_1 = d_1$	$R_1 = \neg d_1$
$S = p$	$(0; 1)$	$(1/9; 8/9)$
$S = l$	$(5/9; 4/9)$	$(6/9; 3/9)$

Table 4.1: The conditional probability tables for $P(R_1|S)$ and $P(R_2|S, R_1)$.

□

[1] Observe that the arcs emanating from e.g. T_1 is also associated with utilities which reflect that the total utility decomposes additively; we assume that the utilities are linear in the monetary values.

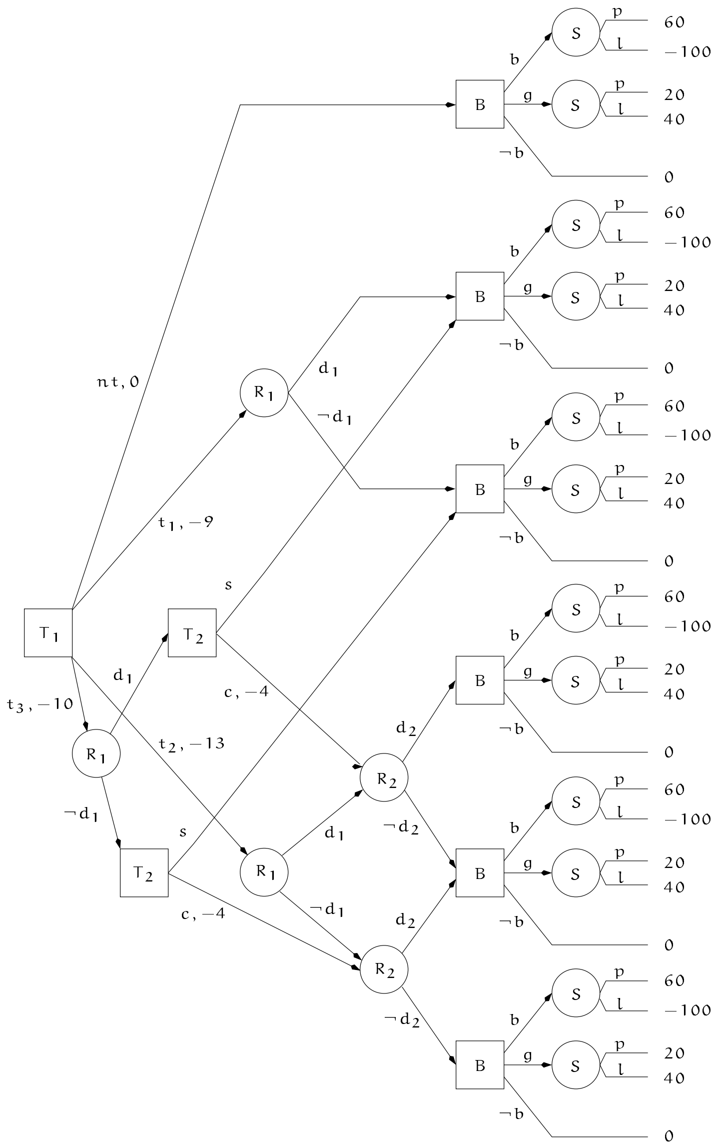

Figure 4.1: A coalesced decision tree representation of the used car buyer problem.

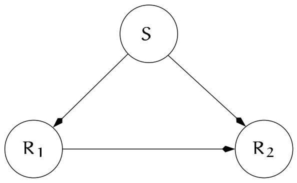

Figure 4.2: A Bayesian network for calculating the probabilities in the used car buyer problem.

Example 4.2.2. [The reactor problem] An electric utility firm is considering building a reactor, and must decide (B) whether to build an advanced reactor (a), a conventional reactor (c) or no reactor at all (n). If an advanced reactor (A) is built, the profit is larger than for a conventional reactor (C) assuming that no accidents occur. However, past experience indicates that an advanced reactor is more probable of having accidents than a conventional reactor. If the firm builds a conventional reactor, the profits are \$8B if it is a success (cs) or -\$4B if there is a failure (cf). On the other hand, the profits of building an advanced reactor are \$12B if it is a success (as), -\$6B if there is a limited accident (al), and -\$10B if there is a major accident (am).

Before deciding on what reactor to build, the firm has the option of having a test (T) performed on the components of the advanced reactor. The results (R) of the test $(T = t)$ can be classified as either bad (b), good (g) or excellent (e); the cost of performing the test is \$1B. If the test results are bad, the nuclear regulatory commission will not allow an advanced reactor to be built.

The coalesced decision tree representation of the reactor problem is depicted in Figure 4.3, and the associated Bayesian network is depicted in Figure 4.4; $P(R|A)$ is given in Table 4.2, $P(A) = (0.66(as); 0.244(al); 0.096(am))$ and $P(C) = (0.98(cs); 0.02(cf))$.

$P(R\|A)$	$R = b$	$R = g$	$R = e$
$A = as$	0	0.182	0.818
$A = al$	0.288	0.565	0.147
$A = am$	0.313	0.437	0.25

Table 4.2: The conditional probability table for $P(R|A)$.

□

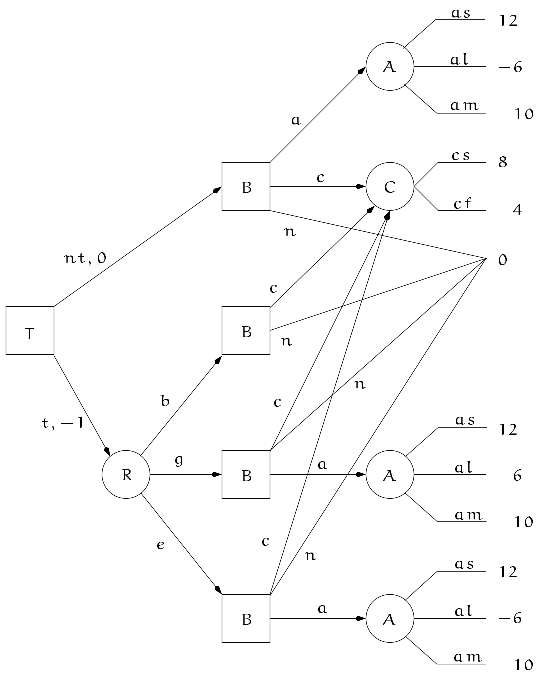

Figure 4.3: A coalesced decision tree representation of the reactor problem.

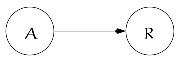

Figure 4.4: A Bayesian network for calculating the probabilities in the reactor problem.

Notice that both of these examples can be considered asymmetric, since the number of scenarios in their (uncoalesced) decision tree representations are less than the Cartesian product of the state spaces of all the chance and decision variables. For instance, the uncoalesced decision tree representation of the reactor problem contains 21 scenarios but the cardinality of the Cartesian product of the chance variables and decision variables is $2 \cdot 3 \cdot 3 \cdot 2 \cdot 3 = 108$. Observe that the coalesced decision tree has only 12 leaves.

4.3 Asymmetric influence diagrams

Qi et al. (1994) state that decision problems are usually asymmetric in the sense that the set of possible outcomes of a chance variable may vary depending on the conditioning states, and the set of legitimate decision options of a decision variable may vary depending on the different information states. Equivalently, in (Bielza and Shenoy 1999; Shenoy 2000) a decision problem is characterized as being asymmetric if, in its decision tree representation, the number of scenarios is less than the cardinality of the Cartesian product of the state spaces of all chance and decision variables. However, both of these characterizations fail to recognize decision problems in which the relative temporal order of two variables vary w.r.t. to previous observations and decisions; this is for example very common in troubleshooting problems. Thus, we define an asymmetric decision problem as follows:

Definition 4.3.1. A decision problem is said to be *asymmetric* if, in its decision tree representation, either:

- the number of scenarios is less than the cardinality of the Cartesian product of the state spaces of all chance and decision variables or
- there exists two scenarios in which the relative temporal order of two variables differ.

In order to deal with such asymmetric decision problems we introduce the *asymmetric influence diagram*. An asymmetric influence diagram is a labeled directed graph $\mathsf{G} = (\mathcal{U}, \mathcal{E}, \mathcal{L})$, where the nodes $\mathcal{U}$ can be partitioned into four disjoint subsets: *test-decision nodes* ($\mathcal{U}_{\mathrm{T}}$), *action-decision nodes* ($\mathcal{U}_{\mathrm{A}}$), chance nodes ($\mathcal{U}_{\mathrm{C}}$) and value nodes ($\mathcal{U}_{\mathrm{V}}$); we will sometimes omit the distinction between test-decisions and action-decisions by simply referring to a node in $\mathcal{U}_{\mathrm{D}} = \mathcal{U}_{\mathrm{T}} \cup \mathcal{U}_{\mathrm{A}}$ as a decision node.

The chance nodes and value nodes are similar to the chance nodes and value nodes which occur in a partial influence diagram. The decision nodes correspond to decision variables and represent actions under the direct control of the decision maker. A test-decision (drawn as a triangle) is a decision to look for more evidence, whereas an action-decision (drawn as a rectangle) is a decision to change the state of the world. We do not distinguish between actions which force a change of state for some of the variables (*intervening actions*) and actions of which the impact is not part of the model (*non-intervening actions*) (Jensen 1996).

The arcs $\mathcal{E}$ in an asymmetric influence diagram can be partitioned into four disjoint subsets. An arc into a value node or a chance node is semantically defined as in a partial influence diagram if it does not emanate from a test-decision node. Arcs into decision nodes, termed informational arcs, imply a possible information precedence; if there is an arc from a node X to a decision node D, then the state of X *may* be known when decision D is made. This redefinition is needed since we deal with asymmetric decision problems, i.e., the set of variables observed immediately before decision D is taken may depend on previous decisions and observations.

If there exists an arc, termed a *test arc*, from a test-decision node D to a chance node X, then the state of D determines whether or not X is eventually observed; having an arc from a test-decision node to a chance node represents a logical relation and does not imply probabilistic dependence. Note that in the trivial case, where X is observed no matter the state of D, the arc (D, X) implies information precedence only. If there exists an arc $(\mathsf{D}, \mathsf{D}')$ from a test-decision node D to another decision node D', then $(\mathsf{D}, \mathsf{D}')$ implies information precedence ($(\mathsf{D}, \mathsf{D}')$ is an informational arc); however, $(\mathsf{D}, \mathsf{D}')$ is termed a test arc if the state of D determines whether or not D' is eventually made. Whether we are referring to an informational arc or a test arc is conveyed by the *label* associated with D'; in the remainder of this chapter we let $\bar{\mathrm{I}}$ denote the graph obtained from I by removing all test arcs and informational arcs.

The asymmetry of a decision problem is graphically represented in the asymmetric influence diagram by a set of *restriction arcs* and by a set of labels $\mathcal{L}$. The set of restriction arcs (drawn as dashed arcs) is a subset of the informational arcs. A restriction arc (X, D) indicates that the set of legitimate decision options for D may vary depending on the state of X, in which case we say that X is *restrictive* w.r.t. D (or X is restricting D). The set of labels $\mathcal{L}$ is associated with a subset of the nodes and informational arcs. A label specifies under which conditions the associated node or informational arc appears in the decision problem. The following rules informally summarize the semantics of labels when specifying asymmetry in asymmetric influence diagrams; in the remainder of this chapter they will be referred to as rule (i)-(iii), respectively:

i) Let X be a node labeled with f_X and let $\mathcal{Y}$ be the variables observed before X is observed (or decided upon). If f_X is unsatisfied w.r.t. the state configuration $\mathcal{Y} = \bar{\mathsf{y}}$ observed, then X is not included in the scenario.

ii) Let (X, D) be an informational arc labeled with $f_{(X,D)}$ and let $\mathcal{Y}$ be the variables observed before X is observed (or decided upon). If $f_{(X,D)}$ is unsatisfied w.r.t. the state configuration $\mathcal{Y} = \bar{y}$ observed, then (X, D) is not included in the scenario.

iii) If there exists a directed path from a node X to a node Y in $\bar{I}$, then whenever X is not included in the scenario Y is not included either.

Based on the rules above we require that if the label of a chance node or value node Z is a function of a node X, then there must exist an arc from X to Z (See Figure 4.5).

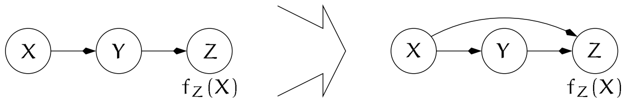

Figure 4.5: There must exist a directed arc from X to Z, since the label f_Z is a function of X.

The use of labels for specifying temporal dependence in examplified in Figure 4.6. The informational arc from Y to D is associated with the label $T = t$ which implies that the associated informational arc is only part of the decision problem if $T = t$. That is, the chance variable Y is only observed before D if we first decide on t w.r.t. the test-decision T; otherwise it is observed (possibly never) after deciding on D.

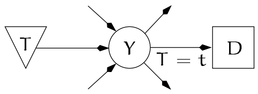

Figure 4.6: The figure illustrates the use of labels when specifying temporal dependence; the variable Y is only observed before D if we decide on t w.r.t. the test-decision T.

Before we give a more formal description of the asymmetric influence diagram, we present the following examples illustrating the use of asymmetric influence diagrams for modeling asymmetric decision problems.

Example 4.3.1. [The dating problem] Joe has to decide whether or not to ask a girl he has recently met on a date. If Joe decides not to ask her out, he can choose either to stay at home and watch TV or visit a night club; before taking that decision Joe observes what programs will be on TV that night. The pleasure of staying at home is influenced by his liking of the program watched, whereas the pleasure of going to a night club is dependent on the comfort of going to that night club and the entrance fee; the comfort depends on whether Joe likes the night club and whether he meets any friends there.

If Joe decides to ask the girl out, her response will depend on her feelings for him. If she declines the date, Joe can decide to go to a night club or stay at home and watch TV; we assume that the two "staying at home scenarios" are the same. If she accepts to go on a date with him, Joe will ask her whether she wants to go to a restaurant or to the movies. The choice of movie (decided by Joe) may influence her mood which in turn may influence Joe's satisfaction concerning the evening. Similarly, the choice of menu (decided by Joe) might influence Joe's satisfaction.

This decision problem is represented by the asymmetric influence diagram depicted in Figure 4.7. The variable *Date?* is represented as a test-decision since it has no impact on the state of *Accept?*.

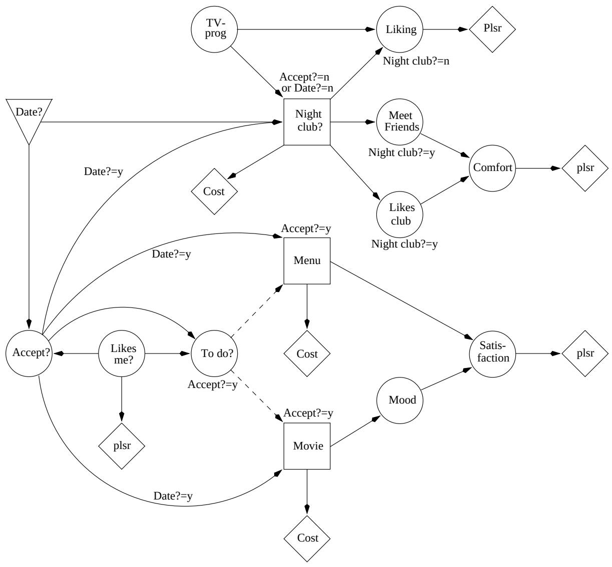

Figure 4.7: An asymmetric influence diagram representing the dating problem.

The decision *Night Club?* is made if Joe initially chooses not to ask the girl for a date; the label associated with *Night Club?* specifies a logical-or and is therefore satisfied by the state configuration *Date?=n*. If Joe chooses to go to a night club the chance variables *Meet Friends* and *Likes Club* influence Joe's comfort which in turn influences the pleasure of going to that club; the chance variable *Liking* is excluded from the decision problem since *Liking* is only included if *Night Club?=n* (rule (i)). Notice that this property could also be modelled by introducing an artificial state in the variable *Liking*. However, as noted previously, having artificial states tends to obscure the asymmetric structure of the decision problem and is, in general, computationally demanding.

Since *Date?=n*, the informational arcs from *Accept?* are excluded (rule (ii)) meaning that her potential response is never observed; *Accept?* is still part of the decision problem as opposed to e.g. *Liking* if *Night Club?*$\neq$ *n*. Now, as *Accept?* is never observed the variables only labeled by the state of *Accept?* are removed (rule (i)), together with all their successors (rule (iii)). The resulting decision problem is depicted in Figure 4.8; the variables *Accept* and *Likes me?* are included in the figure for the purpose of the asymmetric influence diagram being a tool for communication.

If Joe, on the other hand, chooses to ask the girl for a date he will observe her response (*Accept?*). If she declines the invitation, he can choose either to go to a night club or stay at home. If she accepts the invitation, the variable *To do?* will be observed which can restrict the possible decision options for *Menu* and *Movie*.

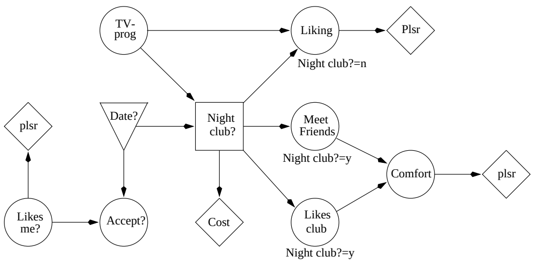

Figure 4.8: The dating problem if *Date?=n*.

There is a small technical problem with the variable *Satisfaction*, which has the mutually exclusive variables *Menu* and *Movie* as predecessors. As no descendant of an excluded variable can be included (rule (iii)), we would unintentionally exclude *Satisfaction* whenever *Menu* or *Movie* are included. This problem can be solved by duplicating *Satisfaction* and its descendants or by adding an extra state (*no-decision*) to the variables *Menu* and *Movie*. In order to minimize redundancy in the representation we have chosen the latter. □

Example 4.3.2. An asymmetric influence diagram representation of the used car buyer problem is shown in Figure 4.9. The decision variable B represents the decision of whether or not to buy the car. The decision variables T_1 and T_2 represent the decisions of having the car examined by a mechanic; these decisions basically determine whether or not more evidence should be retrieved hence, they are represented as test-decisions. The label associated with T_2 specifies that this decision is only part of the decision problem if Joe initially decides to have the mechanic perform the two-test sequence ($\mathsf{T}_1 = \mathsf{t}_3$).

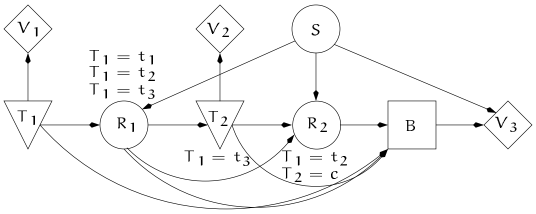

Figure 4.9: An asymmetric influence diagram representing the used car buyer problem.

The chance variables R_1 and R_2 represent the result of the first and second test, respectively, and the chance variable S represents the state of the car. The label associated with R_1 indicates that R_1 is only included in the decision problem if Joe initially chooses to have the car examined by a mechanic, i.e, $T_1 = t_1 \vee T_1 = t_2 \vee T_1 = t_3$. Similarly, the label associated with R_2 specifies that R_2 is only part of the decision problem if either $T_2 = c$ or $T_1 = t_2$. Note that in the first case, we implicitly require that $T_1 = t_3$ or T_2 would not be decided upon. □

Example 4.3.3. An asymmetric influence diagram representation of the reactor problem is depicted in Figure 4.10. The decision to have a test performed on the components of the advanced reactor is represented by the test-decision T; analogously to the examples above, T is basically a decision to look for more evidence. The decision variable B represents the decision to either build an advanced reactor, a conventional reactor or no reactor at all, and the chance variables A and C represent the state of the advanced reactor and the conventional reactor, respectively. The chance variable R represents the result of the test T, and the label associated with R indicates that R is only part of the decision problem if a test is initally performed ($T = t$).

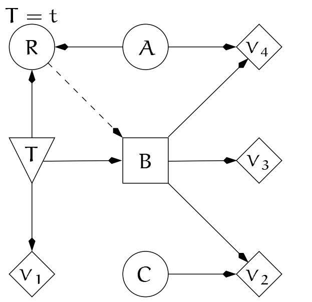

Figure 4.10: An asymmetric influence diagram representing the reactor problem.

The dashed arc from R to B indicates that R may restrict the legitimate decision options for B, i.e., if the test results turn out to be bad, then the nuclear regulatory commision will not allow an advanced reactor to be built. □

4.3.1 The qualitative level

From rule (iii) we can infer the following syntactical simplification: If there exists a directed path from a node X to a node Y in $\bar{I}$, then Y "inherits" the label associated with X, i.e., Y is "effectively" labeled with $f_X \wedge f_Y$, where f_X and f_Y are the labels explicitly associated with X and Y, respectively. This means that we need not explicitly associate Y with the label $f_X \wedge f_Y$ (see Figure 4.11). The set of variables from which a chance node or a value node X "inherits" labels is given by $\mathrm{dep}(X) = \mathcal{Y}$, where $\mathcal{Y}$ is the set of variables from which there exists a directed path to X in $\bar{I}$. For instance, in Figure 4.7 the variable *Meet Friends* is "effectively" conditioned on (*Night club?=y* $\wedge$ (*Accept?=n* $\vee$ *Date?=n*)) since dep(*Meet Friends*) = *Night club?*. To ensure consistency it should be noted that decision nodes and informational arcs "inherit" labels from the empty set.

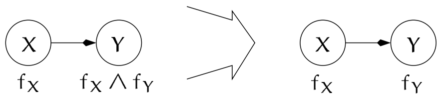

Figure 4.11: The figure illustrates the use of labels when specifying asymmetry.

The asymmetric influence diagram allows the specification of directed cycles with the restriction that before any of the nodes in a cycle are observed the cycle must be "broken", i.e., no matter the variables observed there must exist at least one unsatisfied label associated with a node or an informational arc in the cycle. This syntactical constraint compensates for the traditional constraint that the graph should be acyclic. Having cycles in an asymmetric influence diagram supports the specification of decisions for which the temporal order is dependent on previous observations and decisions. For instance, in Figure 4.12 decision D_2 is taken before D_1 if $X = x$ but D_1 is taken before D_2 if $X \neq x$. Note that the asymmetric influence diagram also allows the occurence of multiple embedded cycles. The use of cycles for representing asymmetry has been further pursued by Demirer and Shenoy (2001).

Formally, a label is a *Boolean function* defined as a combination of *Boolean variables*, the constants *true* (1) and *false* (0) and the operators $\wedge$ (*conjunction*), $\vee$ (*disjunction*), $\neg$ (*negation*), $\Rightarrow$ (*implication*) and $\Leftrightarrow$ (*bi-implication*).

The Boolean variables are used to represent the conditioning on states e.g. if a node Y is conditioned on $X = x$, then $X = x$ should be represented as a Boolean variable in the label associated with Y. However, for ease of notation

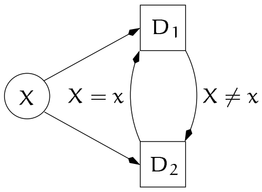

Figure 4.12: The figure illustrates the use of cycles in the asymmetric influence diagram.

we shall use e.g. $X = x$ directly in the label (without creating an actual Boolean variable); a Boolean variable in the context of labels must therefore denote a state configuration of some node in the asymmetric influence diagram.

A *truth assignment* to a Boolean function f is the same as fixing a set of variables in the domain of f, i.e., if $X = x$ represents a Boolean variable in the domain of f, then $X = x$ can be assigned either true or false by associating X with some state $x' \in \mathrm{sp}(X)$ (denoted $f[X \mapsto x']$). E.g. $(X = x)[X \mapsto x'] \equiv 1$ if $x = x'$ and $(X = x)[X \mapsto x'] \equiv 0$ otherwise.

In the remainder of this chapter, we assume that each node X is associated with a label f_X, i.e., if X is not associated with a label in $I = (\mathcal{U}, \mathcal{E}, \mathcal{L})$, then we extend $\mathcal{L}$ with the label $f_X \equiv 1$. Moreover, we will use $\mathrm{dom}(f)$ to denote the domain of the label f; the domain of a label is the set of nodes referenced by the label.

A chance variable X is said to be *present* in I given a configuration $\bar{c}$ over a set of variables $\mathcal{C}$, if $(f_X \wedge \bigwedge_{Y \in \mathrm{dep}(X)} f_Y)[\mathcal{C} \mapsto \bar{c}] \equiv 1$ in I given $\bar{c}$. A chance variable X is said to be *unresolved* in I given a configuration $\bar{c}$ over a set of variables $\mathcal{C}$ if $\mathrm{dom}((f_X \wedge \bigwedge_{Y \in \mathrm{dep}(X)} f_Y)[\mathcal{C} \mapsto \bar{c}]) \neq \emptyset$ and $(f_X \wedge \bigwedge_{Y \in \mathrm{dep}(X)} f_Y)[\mathcal{C} \mapsto \bar{c}] \not\equiv i$, $\forall i \in \{0, 1\}$. The concepts *present* and *unresolved* are similarly defined for value nodes, decision nodes and informational arcs.

The *projection* of a configuration $\bar{c}$ on the variables $\mathcal{C}'$ (denoted $\bar{c}^{\downarrow \mathcal{C}'}$) simply means dropping extra coordinates e.g. $\{T_1 = t_3, T_2 = s\}^{\downarrow \{T_1\}} = \{T_1 = t_3\}$. In what follows we will use ∇ to denote the configuration over the empty set $\emptyset$.

4.3.2 The quantitative level

A decision variable D is associated with a set of *restrictive functions*; a restrictive function is given by $\gamma_D : \mathrm{sp}(\mathrm{pa}(D)^r) \hookrightarrow 2^{\mathrm{sp}(D)}$ and specifies the legitimate decision options for D given a configuration of $\mathrm{pa}(D)^r \subseteq \mathrm{pa}(D)^R$, where $\mathrm{pa}(D)^R$ denotes the set of variables which can restrict the legitimate decision options for D. For instance, in Example 4.3.3 the restrictive function $\gamma_B : \mathrm{sp}(R) \hookrightarrow 2^{\mathrm{sp}(B)}$ associated with B indicates that the nuclear regulatory commission will not allow an advanced reactor to be built if the result R of the initial test was bad, see Table 4.3. Similarly, in Figure 4.7 the restrictive function associated with *Movie* specifies that the state *no-decision* is the only legitimate decision option if *To do?=restaurant* (similar for *Menu* if *To do?=movie*).

$\gamma_B(R)$		B
R	e	$\{n, c, a\}$
	g	$\{n, c, a\}$
	b	$\{n, c\}$

Table 4.3: The restrictive function γ_B associated with B in Example 4.3.3.

The uncertainty associated with a chance variable X is represented by a *partial conditional probability potential* $\phi_X = P(X|pa(X)') : sp(X \cup pa(X)') \hookrightarrow [0;1]$, where $pa(X)' = pa(X) \setminus \mathcal{U}_T$; by definition, a test-decision variable has no probabilistic influence on a chance variable (see e.g. Figure 4.9 and Table 4.1, or Figure 4.10 and Table 4.2). A partial conditional probability potential can specify that given a configuration of the conditioning set for a chance variable X, some states of X are impossible (denoted $\perp$); we make a conceptual distinction between an impossible state and a state having zero probability. Note that if all the states of X are impossible for some configuration of the conditioning set, then this must be reflected in the labeling of X e.g. if $P(X|Y)$ is only defined for $Y = y$, then $Y = y$ must occur in the label of X.

A value node V is associated with a *partial utility function* $\psi_V : sp(pa(V)) \hookrightarrow \mathbb{R}^+$; requiring that a partial utility function does not take on negative values is not an actual restriction since any utility function can be transformed s.t. it adheres to this assumption. Note that this restriction is only required for the evaluation of asymmetric influence diagrams (see Chapter 6), so during the modeling phase we may specify utility functions having negative values. Furthermore, as for the partial influence diagram we assume that the total utility is the sum of the local utilities.

The combination of partial potentials (addition, multiplication and division) is defined similarly to the combination of total functions by treating the undefined value ($\perp$) as an additive identity and a multiplicative zero. In particular, this ensures consistency when defining the total utility as being the sum of the local utilities. The following operations defines multiplication and addition of partial functions. If $\bar{f} : \mathcal{U}_1 \hookrightarrow \mathcal{R}_1$ and $\bar{g} : \mathcal{U}_2 \hookrightarrow \mathcal{R}_2$ are partial functions, then the combination of $\bar{f}$ and $\bar{g}$ is defined as:[2]

$$(\bar{f} \cdot \bar{g})(\bar{x}) = \begin{cases} \bar{f}(\bar{x}^{\downarrow \mathcal{U}_1}) \cdot \bar{g}(\bar{x}^{\downarrow \mathcal{U}_2}) & \text{if } \bar{f}(\bar{x}^{\downarrow \mathcal{U}_1}) \neq \bot \text{ and } \bar{g}(\bar{x}^{\downarrow \mathcal{U}_2}) \neq \bot, \\ \bot & \text{otherwise.} \end{cases}$$

$$(\bar{f} + \bar{g})(\bar{x}) = \begin{cases} \bar{f}(\bar{x}^{\downarrow \mathcal{U}_1}) + \bar{g}(\bar{x}^{\downarrow \mathcal{U}_2}) & \text{if } \bar{f}(\bar{x}^{\downarrow \mathcal{U}_1}) \neq \bot \text{ and } \bar{g}(\bar{x}^{\downarrow \mathcal{U}_2}) \neq \bot, \\ \bar{f}(\bar{x}^{\downarrow \mathcal{U}_1}) & \text{if } \bar{f}(\bar{x}^{\downarrow \mathcal{U}_1}) \neq \bot \text{ and } \bar{g}(\bar{x}^{\downarrow \mathcal{U}_2}) = \bot, \\ \bar{g}(\bar{x}^{\downarrow \mathcal{U}_2}) & \text{if } \bar{g}(\bar{x}^{\downarrow \mathcal{U}_2}) \neq \bot \text{ and } \bar{f}(\bar{x}^{\downarrow \mathcal{U}_1}) = \bot, \\ \bot & \text{otherwise.} \end{cases}$$

As for a partial influence diagram we define a *realization* of an asymmetric influence diagram as an attachment of partial probability potentials and utility functions to the appropriate variables. The partial probability potentials and utility functions associated with an asymmetric influence diagram I is denoted Φ_I and Ψ_I, respectively. Given a realization $\mathcal{R} = \Phi_I \cup \Psi_I$ of an asymmetric influence diagram I the set of partial probability potentials with $X \in \mathcal{X}$ in the domain will be denoted $\Phi_{\mathcal{X}}$, i.e., $\Phi_{\mathcal{X}} = \{\phi \in \Phi_I | \exists X \in \mathcal{X} : X \in \mathrm{dom}(\phi)\}$. Note that we shall sometimes refer to a partial probability potential simply as a probability potential; similarly for partial utility functions.

Finally, recall the distinction between structural and functional asymmetry; structural asymmetry has to do with the occurrence of variables in different scenarios, whereas functional asymmetry refers to the possible outcomes and decision options of the variables. From the specification of the asymmetric influence diagram, we see that structural asymmetry is specified at the qualitative level, whereas functional asymmetry is specified at quantitative level.

4.3.3 Split configurations and decisions in context

When making a decision in an asymmetric decision problem, previous observations and decisions may determine the variables observed before the decision in question; for instance, in Example 4.3.2, the chance variable R_2 is not observed before decision B if e.g. $T_1 = t_1$. For both semantic and computational reasons it is important to identify the variables actually observed before taking a particular decision. For example, in the asymmetric influence diagram depicted in Figure 4.7 it would not be meaningful to have a policy for *Menu* conditioned on both *Date?=n* and *Accept?=y* since this configuration is impossible. Similarly, in Figure 4.13 a policy for D cannot be conditioned on both X and Y since these variables do not coexist. So, in order to reason about the different informational states when taking a decision D we must associate with D a context describing the variables observed. That is, we need to identify the possible temporal

[2] The division of two partial functions is defined similarly to multiplication.

orderings of the variables and, in particular, the variables which influence the occurrence of future variables.

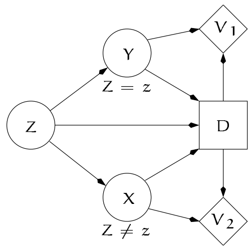

Figure 4.13: The optimal policy for D cannot be conditioned on both X and Y since these variables do not coexist.

Definition 4.3.2. Let I be an asymmetric influence diagram and let $\mathcal{L}$ be the set of labels associated with I. A variable X is said to be a *split variable* in I if there exists a label $f \in \mathcal{L}$ s.t. $X \in \mathrm{dom}(f)$. The set of split variables in I is denoted $\mathcal{S}_I$.[3]

The partial order $\prec$ induced by an asymmetric influence diagram I can now be found by initially treating I as a partial influence diagram (ignoring any labels) and then refining the partial order $\prec'$ induced by the partial influence diagram s.t. for any pair of variables X and Y, where $X \not\prec' Y$ and $Y \not\prec' X$ we have $X \prec Y$ if $X \in \mathcal{S}_I$ (see Figure 4.14); if $\{X, Y\} \subseteq S_I$ we have $X \not\prec Y$ and $Y \not\prec X$.

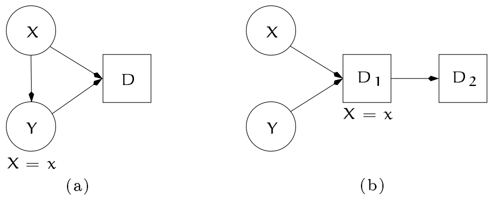

Figure 4.14: In part (a) the asymmetric influence diagram induces the partial order $X \prec Y \prec D$, and in part (b) the asymmetric influence diagram induces the partial order $X \prec Y \prec D_1 \prec D_2$.

In what follows we assume that an asymmetric influence diagram has exactly one split variable S^1_∇ (termed the *initial split variable*) satisfying $\forall Y \in \mathcal{S}_I \backslash \{S^1_\nabla\}$: $S^1_\nabla \prec Y$; *Date?* is the initial split variable in the asymmetric influence diagram depicted in Figure 4.7. Obviously, the set of variables succeeding the initial split

[3]The notion of a split variable is inspired by the framework of Højsgaard (1998).

variable S^1_∇ is dependent of the state of S^1_∇; hence, we define the concept of a missing variable:

Definition 4.3.3. Let I be an asymmetric influence diagram and let S^1_∇ be the initial split variable in I. The chance variable X is said to be *missing* in I given $S^1_\nabla = s_1$ if:

i) $f_X[S^1_\nabla \mapsto s_1] \equiv 0$ or

ii) $\exists Y \in \mathrm{dep}(X)$ s.t. Y is missing given $S^1_\nabla = s_1$ or

iii) $\forall S \in \mathcal{S}_I \backslash \{S^1_\nabla\} : X \prec S$ and X is unresolved given $S^1_\nabla = s_1$.

The above definition is easily adopted to value nodes, decision nodes and informational arcs and will therefore not be described further.[4]

The following definition specifies the asymmetric influence diagram obtained from another asymmetric influence diagram I by instantiating the initial split variable in I.

Definition 4.3.4. Let $I = (\mathcal{U}, \mathcal{E}, \mathcal{L})$ be an asymmetric influence diagram and let S^1_∇ be the initial split variable in I. The asymmetric influence diagram $I' = (\mathcal{U}', \mathcal{E}', \mathcal{L}')$ is said to be *myopicly reduced* from I given $S^1_\nabla = s_1$ if:

- $\mathcal{U}' = \{X \in \mathcal{U} | X$ is not missing given $S^1_\nabla = s_1\}$.
- $\mathcal{E}' = \{(X, Y) \in \mathcal{E} | \{X, Y\} \subseteq \mathcal{U}'$ and (X, Y) is not missing in I given $S^1_\nabla = s_1\}$.
- $\mathcal{L}' = \{f_X[S^1_\nabla \mapsto s_1] | f_X \in \mathcal{L}$ and $X \in \mathcal{U}'\} \cup \{f_{(X,Y)}[S^1_\nabla \mapsto s_1] | f_{(X,Y)} \in \mathcal{L}$ and $(X, Y) \in \mathcal{E}'\}$.

That is, we myopicly reduce an asymmetric influence diagram I by removing the missing nodes and the missing arcs. However, the removal of arcs might render additional nodes missing; thus, for all missing nodes and arcs to be removed we need to remove them iteratively. The asymmetric influence diagram I' obtained from I by iteratively removing missing nodes and arcs is said to be *reduced* from I given $S^1_\nabla = s_1$ and is denoted $I[S^1_\nabla \mapsto s_1]$.

Notice that reducing an asymmetric influence diagram w.r.t. its initial split variable S^1_∇ is the same as instantiating S^1_∇; hence, S^1_∇ is not a split variable in $I[S^1_\nabla \mapsto s_1]$.

In the remainder of this chapter we restrict our attention to asymmetric influence diagrams having exactly one initial split variable. This restriction also applies

[4]Note that a node or an informational arc may also be missing without conditioning on a split variable.

to all asymmetric influence diagrams which have been reduced from other asymmetric influence diagrams. That is, we do not consider asymmetric influence diagrams which can be reduced to an asymmetric influence diagram that does not adhere to this restriction; having a unique initial split variable ensures that the reduction is unambiguous, and it does not seem to exclude any natural decision problems (actually, decision trees have the same property). Furthermore, for ease of notation we shall treat restrictive variables as split variables unless stated otherwise ($\mathcal{S}_I^R$ denotes the union of $\mathcal{S}_I$ and the set of restrictive variables in I). However, we do not require the occurrence of a unique restrictive variable as the order in which the restrictive variables are instantiated is of no importance, i.e., the syntactical constraint on the split variables does not extend to the restrictive variables.

Now, based on the requirement about a unique split variable in an asymmetric influence diagram , I, we can identify the possible state configurations in I. These configurations are found by going in the temporal order specified by I. That is, we iteratively identify the initial split variable in the asymmetric influence diagram reduced from the asymmetric influence diagram in the previous step and assign this split variable a configuration consistent with the previous one.

The initial split variable S^1_∇ in I is identified as described previously. In general, the k*'th split variable* in I w.r.t. the configuration $\bar{s}_{k-1} = (s_1, s_2, \ldots, s_{k-1})$, denoted $S^k_{\bar{s}_{k-1}}$, is the initial split variable in $I[S^1_\nabla \mapsto s_1][S^2_{\bar{s}_1} \mapsto s_2]\cdots[S^{k-1}_{\bar{s}_{k-2}} \mapsto s_{k-1}]$, where $S^i_{\bar{s}_{i-1}}$ is the initial split variable in I w.r.t. the configuration $\bar{s}_{i-1} = (s_1, s_2, \ldots, s_{i-1})$. Obviously, if I' has been reduced from I w.r.t. $S^1_\nabla = s_1$, then s_1 must be a possible state for S^1_∇, i.e., each time an asymmetric influence diagram is reduced the possible outcomes/decision options for the variables are "updated". In what follows we let $I[S^i_{\bar{s}_{i-1}} \mapsto \bar{s}_i]$ denote the asymmetric influence diagram $I[S^1_\nabla \mapsto s_1][S^2_{\bar{s}_1} \mapsto s_2]\cdots[S^i_{\bar{s}_{i-1}} \mapsto s_i]$.

Definition 4.3.5. Let $\mathcal{S}'_I \subseteq \mathcal{S}^R_I$ and $\mathcal{S}''_I = \{S_1, S_2, \ldots, S_l\} \subseteq \mathcal{S}'_I$ be subsets of the split variables contained in the asymmetric influence diagram I. A configuration $\bar{s} = (s_1, s_2, \ldots, s_l)$ over the variables $\mathcal{S}''_I$ is said to be a *split configuration* for $\mathcal{S}'_I$ over $\mathcal{S}''_I$ if:

- S_i is the i'th split variable in I w.r.t. the configuration $\bar{s}_{i-1} = (s_1, s_2, \ldots, s_{i-1})$ and
- S is not a split variable in $I[S^l_{\bar{s}_{l-1}} \mapsto \bar{s}_l]$, $\forall S \in \mathcal{S}'_I \backslash S''_I$.

If $\mathcal{S}'_I = \mathcal{S}^R_I$ then $\bar{s}$ is said to be an *exhaustive split configuration* for I over $\mathcal{S}''_I$.

Notice that for any exhaustive split configuration $\bar{s}$ for I over $\mathcal{S}'_I$ the asymmetric influence diagram $I[\mathcal{S}'_I \mapsto \bar{s}]$ is a partial influence diagram.

For notational convenience, we will sometimes use $\mathrm{I}[\mathcal{S}''_{\mathrm{I}} \mapsto \bar{s}]$ to denote the asymmetric influence diagram reduced from the asymmetric influence diagram I w.r.t. the split configuration $\bar{s}$ over the variables $\mathcal{S}''_{\mathrm{I}}$.

Example 4.3.4. Consider the asymmetric influence diagram depicted in Figure 4.7. The configuration (*Date?=y*) is a split configuration, whereas (*Date?=y*, *Accept?=y*, *To do?=movie*) is an exhaustive split configuration. The configuration (*Date?=n,Club?=n*) is also an exhaustive split configuration since *Date?=n* implies that *Accept?* is never observed, thereby rendering the variables *Movie*, *Menu* and *To Do?* missing. □

The set of exhaustive split configurations for an asymmetric influence diagram I determines the decision scenarios in the corresponding decision tree (disregarding the scenarios which are specified as impossible at the quantitative level). Furthermore, we can use the notion of a split configuration to determine in which *contexts* a decision variable can occur. A context for a decision variable D is a configuration $\bar{s} = (\bar{\omega}, \bar{x})$, where:

- $\bar{s}$ is a split configuration for $\mathcal{S}'_{\mathrm{I}}$ over the variables $\mathcal{S}''_{\mathrm{I}}$ satisfying that there does not exist a split variable $S \in \mathcal{S}_{\mathrm{I}[\mathcal{S}''_{\mathrm{I}} \mapsto \bar{s}]}$ s.t. $S \prec D$ in $\mathrm{I}[\mathcal{S}''_{\mathrm{I}} \mapsto \bar{s}]$.
- $\bar{x}$ is a configuration over the restrictive variables for D in $\mathrm{I}[\mathcal{S}''_{\mathrm{I}} \mapsto \bar{s}]$.

The set of contexts in which a decision variable D can occur is denoted Ω_{D}, e.g., in Example 4.3.2, the contexts in which B can occur are given by $\Omega_{\mathrm{B}} = \{(T_1 = nt), (T_1 = t_1), (T_1 = t_2), (T_1 = t_3, T_2 = s), (T_1 = t_3, T_2 = c)\}$.

The contexts for a decision variable also underlies the specification of a well-defined asymmetric influence diagram, which will be described in Chapter 6.

4.4 Comparison

In this section we compare the proposed framework with the *extended influence diagram* framework (Smith et al. 1993) and the framework of *asymmetric valuation networks* (Shenoy 2000). The aim of this comparison is to clarify the differences between the modeling aspects of the frameworks, and to elaborate over their strengths and weaknesses. Note that we do not intend to compare the extended influence diagram and the asymmetric valuation network, see (Bielza and Shenoy 1998; Bielza and Shenoy 1999) for such a comparison.

4.4.1 Comparison with asymmetric valuation networks

The asymmetric valuation network by Shenoy (2000) is a generalization of the valuation network (Shenoy 1992) for representing symmetric decision problems.

The specification of an asymmetric valuation network includes three different levels; *graphical*, *dependence* and *numeric* (these levels also occur in the specification of a valuation network).

At the graphical level, the asymmetric valuation network consists of a directed acyclic graph, where the nodes are partitioned into *variables* and *valuations*. The variables are furthermore classified as either decision variables or chance variables, having their usual semantic interpretation. The valuations are classified as either *probability valuations* (drawn as triangles), *utility valuations* (drawn as a diamonds) or *indicator valuations* (drawn as a double-bordered triangles). The domain of a valuation is its neighbouring variables in the valuation network.

The probability valuations represent multiplicative factors of the joint probability distribution over the chance variables. If a probability valuation is conditioned upon a set of variables, then this is indicated by directed arcs from the probability valuation to the variables in the head of the valuation; the conditioning variables are specified by undirected arcs between the valuation and the variables in question. The utility valuations represent either additive or multiplicative factors of the total utility, whereas the indicator valuations represent constraints on the possible configurations of the variables in their domains.

The graphical level includes directed arcs between pairs of distinct variables. Semantically, these arcs imply information precedence: If A is a chance variable and D is a decision variable, then the occurrence of a directed arc (A, D) implies that the state of A is observed before deciding on D; conversely, the arc (D, A) implies that the state of A is observed after deciding on D. The information constraints are said to be *well-defined* if for any chance variable A and for any decision variable D, there exists a directed path from A to D or from D to A. Note that this syntactical constraint is actually "weaker" than the equivalent constraint in the influence diagram, i.e., the information constraints in an asymmetric valuation network are well-defined even though information precedence is not specified between two neighbouring decision variables. Based on the notion of well-defined information constraints the decision maker is assumed to have *perfect recall*, i.e., if there is a directed path from a variable X to a decision variable D, then the state of X is known when deciding on D; this also implies that if there is a directed path from D to X then the state of X is not known when deciding on D. Note that the latter implication is a structural property which holds for all realizations of the valuations, however, there may exist realizations s.t. the state configuration observed before deciding on D reveals the state of X, see e.g. (Madsen et al. 2001).

Example 4.4.1. Consider the asymmetric valuation network representation of the used car buyer problem (see Example 4.2.1) depicted in Figure 4.15. The decision variables T_1 and T_2 represent the first and second test decision, respectively, and the decision variable B represents the decision to buy. The chance variables R_1 and R_2 represent the test results; R_1 is the result of the first test, and R_2 is the result of the second test. The variable S represents the state of the car.

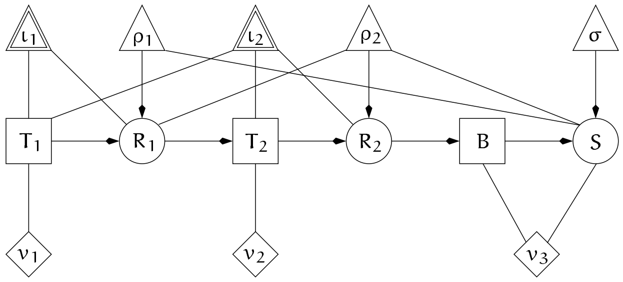

Figure 4.15: An asymmetric valuation network representing the used car buyer problem.

The valuations ι_1 and ι_2 are indicator valuations and represent constraints on the joint state space of $\{T_1, R_1\}$ and $\{T_1, T_2, R_2\}$, respectively. The utility valuations ν_1, ν_2, and ν_3 are additive factors of the total utility function: ν_1 and ν_2 encode the cost of the first and second test decision, and ν_3 encodes the value of the car minus the cost of repairing the defects and buying the car. The probability valuations ρ_1, ρ_2 and σ are multiplicative factors of the joint probability distribution, where $\rho_1 = P(R_1|S)$, $\rho_2 = P(R_2|R_1, S)$ and $\sigma = P(S)$. □

The dependence level consists of a formal specification of the indicator valuations, described at the graphical level, together with a specification of the state space for each chance variable and decision variable.

In the used car buyer problem, the state spaces of the variables R_1, T_2 and R_2 are augmented with artificial states to capture the asymmetric properties of the decision problem. I.e., $sp(R_1) = \{n_1, d_1, \neg d_1\}$, where n_1 denotes no result; $sp(T_2) = \{nc, s, c\}$, where nc denotes no choice; and $sp(R_2) = \{n_2, d_2, \neg d_2\}$, where n_2 denotes no result.

As previously mentioned, the indicator valuations ι_1 and ι_2 specify constraints on the joint state space of the variables in their domains.[5] For instance, ι_1 specifies that the result R_1 of the first test T_1 is only available if Joe initially decides to have the test performed:

$$\Omega_{\iota_1} = \{(nt, n_1), (t_1, d_1), (t_1, \neg d_1), (t_2, d_1), (t_2, \neg d_2), (t_3, d_1), (t_3, \neg d_1)\}.$$

The indicator valuations are also used to specify the *effective state space* for a set of variables. For a given set $\mathcal{U}'$ of variables, the effective state space can be

[5]An indicator valuation ι can unambiguously be specified by the legal configurations of the variables in its domain (denoted Ω_ι).

found by marginalization and combination of the indicator valuations having a variable from $\mathcal{U}'$ in the domain.

If ι is an indicator valuation with domain $\mathsf{dom}(\iota)$, then the marginalization of ι down to $\mathcal{U}' \subseteq \mathsf{dom}(\iota)$ (denoted $\Omega_\iota^{\downarrow \mathcal{U}'}$) is defined as:

$$\Omega_\iota^{\downarrow \mathcal{U}'} = \{\bar{x} \in \mathsf{sp}(\mathcal{U}') | (\bar{x}, \bar{y}) \in \Omega_\iota \text{ for some } \bar{y} \in \mathsf{sp}(\mathsf{dom}(\iota) \backslash \mathcal{U}')\}.$$

The combination of two indicator valuations ι_1 and ι_2 (denoted $\Omega_{\iota_1} \otimes \Omega_{\iota_2}$) is defined as:

$$\Omega_{\iota_1} \otimes \Omega_{\iota_2} = \{\bar{x} \in \mathsf{sp}(\mathsf{dom}(\iota_1) \cup \mathsf{dom}(\iota_2)) | \bar{x}^{\downarrow \mathsf{dom}(\iota_1)} \in \Omega_{\iota_1}, \bar{x}^{\downarrow \mathsf{dom}(\iota_2)} \in \Omega_{\iota_2}\},$$

where $\bar{x}^{\downarrow \mathsf{dom}(\iota_j)}$ (for for $j = 1, 2$) denotes the projection of $\bar{x}$ down to $\mathsf{dom}(\iota_j)$, i.e., dropping coordinates that are not in $\mathsf{dom}(\iota_j)$.

Based on these operations, the effective state space for a set of variables $\mathcal{U}'$ (denoted $\Omega_{\mathcal{U}'}$) is found by combining all indicator valuations with a variable from $\mathcal{U}'$ in the domain, and then marginalizing out the variables not in $\mathcal{U}'$:

$$\Omega_{\mathcal{U}'} = (\otimes\{\Omega_{\iota_j} | \mathsf{dom}(\iota_j) \cap \mathcal{U}' \neq \emptyset\})^{\downarrow \mathcal{U}'}.$$

At the numerical level the utility and probability valuations are specified w.r.t. the effective state space for the variables in their domains. Thus, no values are specified for impossible configurations, i.e., configurations not contained in the effective state space.

Example 4.4.2. An asymmetric valuation network representing the reactor problem (see Example 4.2.2) is depicted in Figure 4.16. The decision variable T represents the decision to have a test performed on the components of the advanced reactor, and the decision variable B represents the decision to either build an advanced reactor, a conventional reactor or no reactor at all. The chance variable R represents the result of the test, and the chance variables A and C represent the state of the advanced reactor and the conventional reactor, respectively.

The state space of the variable R is extended with an artificial state n to encode that a test result is not observed if the initial test is not performed, i.e., $\mathsf{sp}(\mathsf{R}) = \{b, g, e, n\}$.

The utility valuation ν_3 specifies the cost of having a test performed on the components of the advanced reactor. The utility valuations ν_1 and ν_2 specify the value of building a conventional reactor and an advanced reactor, respectively. The probability valuation ρ corresponds to the conditional probability distribution $\mathsf{P}(\mathsf{R}|\mathsf{A})$, and α and χ correspond to the marginal probability distributions $\mathsf{P}(\mathsf{A})$ and $\mathsf{P}(\mathsf{C})$.

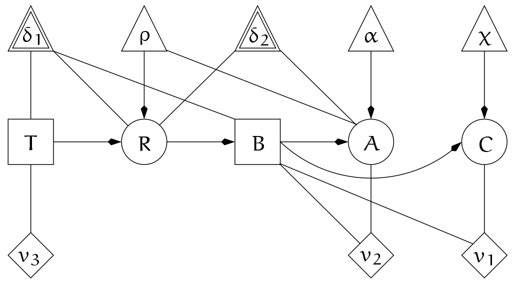

Figure 4.16: An asymmetric valuation network representing the reactor problem.

The indicator valuation δ_1 encodes the possible state configurations of T, R and B; if no test is performed then no test result is available, and if a test is performed and the test result is bad, then an advanced reactor is not allowed to be build. The indicator valuation δ_2 specifies that the configuration $(\mathsf{R} = \mathsf{b}, \mathsf{A} = \mathsf{as})$ has zero probability:

$$\begin{aligned}\Omega_{\delta_1} =& \{(t,b,n),(t,g,n),(t,e,n),(t,b,c),(t,g,c),(t,e,c),(t,g,a),(t,e,a),\\ &(nt,nr,n),(nt,nr,c),(nt,nr,a)\}\\ \Omega_{\delta_2} =& \{(b,al),(b,am),(g,as),(g,al),(g,am),(e,as),(e,al),(e,am),(nr,as),\\ &(nr,al),(nr,am)\}.\end{aligned} \tag{4.5}$$

The effective state space for $\{\mathsf{R}, \mathsf{A}\}$ (used when specifying the probability valuation ρ) is given by:

$$\begin{aligned}\Omega_{\{R,A\}} &= (\Omega_{\delta_1} \otimes \Omega_{\delta_2})^{\downarrow\{R,A\}} = \Omega_{\delta_1}^{\downarrow\{R,A\}} \otimes \Omega_{\delta_2}\\ &= \{(b,al),(b,am),(g,as),(g,al),(g,am),(e,as),(e,al),(e,am),\\ &\quad (nr,as),(nr,al),(nr,am)\}.\end{aligned} \tag{4.6}$$

Thus, when specifying the probability valuation ρ we assign values to the state configurations in the effective state space $\Omega_{\{R,A\}}$ for the variables R and A, i.e., no value is specified for $(\mathsf{R} = \mathsf{b}, \mathsf{A} = \mathsf{as})$. $\square$

Comparison

Compared to the asymmetric influence diagram (and the partial influence diagram) the information constraints are unnecessary strong; a well-defined partial influence diagram (which is also an asymmetric influence diagram) can easily be transformed into an asymmetric valuation network, where the information constraints are not well-defined; as an example we can consider any well-defined partial influence diagram including a chance variable incompatible with a decision variable. Note that the information constraints in the asymmetric valuation network (and in the valuation network) are still weaker than in the influence diagram.

Moreover, the asymmetric valuation network does not support the representation of decision problems in which the relative temporal order of two variables vary w.r.t. previous observations and decisions, i.e., the information constraints are predetermined. It should, however, be noted that this can be modeled (somewhat inefficiently) by creating multiple instances of the variables in question s.t. each set of "copies" are given a distinct temporal order; consistency can be ensured through indicator valuations and by introducing artificial states for these variables.

The asymmetric valuation network uses indicator valuations to encode information about asymmetry in the decision problem. These valuations are included at the graphical level, but they only reveal the existence of structural dependences between variables (asymmetry is specified at the dependence level). In the asymmetric influence diagram, asymmetry is represented at both the qualitative and the quantitative level corresponding to structural and functional asymmetry, respectively. At the qualitative level, structural asymmetry is specified by labels associated with nodes and informational arcs and, at the quantitative level, partial potentials and restrictive functions are used for specifying functional asymmetry; it is important to notice that functional asymmetry is a property which is local to a set of variables. So, as opposed to the asymmetric valuation network, the asymmetric influence diagram allows information about structural asymmetry to be deduced directly from the model. From a modeling perspective, however, it may be a disadvantage to have asymmetry encoded at two different levels of abstraction.

In the asymmetric influence diagram the uncertainties in the decision problem are given in terms of conditional probability potentials, which contribute to an intuitive understanding of the model. In the asymmetric valuation network the probability valuations are not as easy to interpret (semantically, they constitute multiplicative factors of the joint probability), but they are more convenient in situations where a causal model of the probabilities is not available e.g. if the probability model is inferred from a data base. Obviously, if the model is not intended as a tool for communication between humans, then the interpretation of the probability potentials and the probability valuations is of no importance.

An asymmetric valuation network tends to have a larger number of artificial states than an asymmetric influence diagram. Artificial states are used to encode that certain variables may be irrelevant in some scenarios but, in the asymmetric influence diagram, this information is mainly represented by labels associated with the variables in question. For instance, the asymmetric valuation network representation of the used car buyer problem requires an artificial state ("no-result") associated with both R_1 and R_2, and another artificial state ("no-choice") associated with T_2. These artificial states are not include in the asymmetric influence diagram representation, since this information is specified by labels. Note that for some decision problems the asymmetric influence diagram may also require artificial states (or duplicated variables) as described in Example 4.3.1.

Finally, the asymmetric valuation network includes an explicit representation of the valuations at the graphical level. From this representation we can deduce information about the structure of the decision problem. However, the explicit representation of valuations may also have the potential drawback of obscuring the structure of the decision problem. As an example, consider the asymmetric valuation network representation of the dating problem illustrated in Figure 4.17.

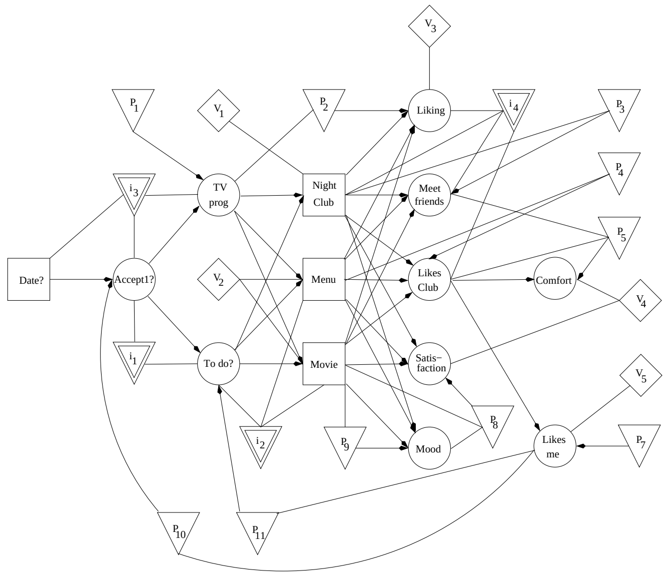

Figure 4.17: An asymmetric valuation network representing the dating problem; the utility valuations for *Satisfaction* and *Comfort* are combined, and the joint probability distribution is assumed to decompose as in Figure 4.7.

4.4.2 Comparison with extended influence diagrams

The *extended influence diagram* framework was proposed by Smith et al. (1993), and augments the influence diagram framework with another representation for describing its conditional distributions. The specification of an extended influence diagram includes two different levels: the *relational level* and the *functional level*.

The relational level consists of a directed acyclic graph similar to the traditional influence diagram, except that it also includes so-called *conditioning arcs* directed towards the decision nodes. For a particular decision node D, these arcs are used for specifying the variables that may restrict the legitimate decision options for D (conditioned on the different scenarios in which D occur). Smith et al. (1993) also assume that the influence diagram contains exactly one value node, however, Bielza and Shenoy (1999) show how this assumption can be relaxed by decomposing the utility function according to the technique by Tatman and Shachter (1990).

Example 4.4.3. An influence diagram representation of the used car buyer problem (see Example 4.2.1) is depicted in Figure 4.18. The decision variables T_1 and T_2 represent the test decisions, and the chance variables R_1 and R_2 represent the outcome of the first and second test, respectively. The chance variable S represents the state of the car, and the decision variable B corresponds to the decision of whether or not to buy the car. The value node V is associated with the total utility function.

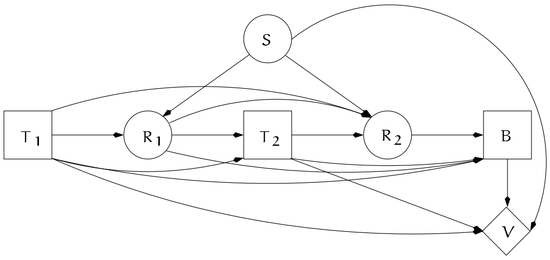

Figure 4.18: The used car buyer problem represented in the framework of influence diagrams.

□

The conditional probability distributions and the utility functions are specified at the functional level. This level also includes a specification of the legitimate decision options for the decision variables, conditioned on the different information scenarios. These specifications are represented by so-called *distribution trees*, which contain both numerical information and information about asymmetry. To emphasize the correspondence between the information states and the conditional distributions assigned to these states, Smith et al. (1993) propose to partition the conditional distributions into a *conditioning function* and an *atomic distribution*. In terms of conditional probability distributions, the conditioning function maps each possible conditioning scenario to an atomic distribution, which describe the probability distribution assigned to that particular scenario. This partitioning of the conditional distributions is reflected in the distribution trees, where the leaf nodes (and their outgoing arcs) correspond to the atomic distributions. For instance, the conditional probability distribution $\mathsf{P}(\mathsf{A}|\mathsf{B})$ can be represented by the distribution tree depicted in Figure 4.19 (the nodes are assumed to be binary).

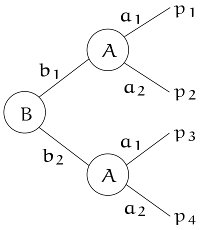

Figure 4.19: The figure illustrates a distribution tree for the conditional probability distribution $\mathsf{P}(\mathsf{A}|\mathsf{B})$. The leaf nodes (and their outgoing arcs) correspond to atomic distributions, whereas the node B (and its outgoing arcs), termed the conditioning tree, corresponds to the conditioning function.

The following example illustrates the extended influence diagram representation of the reactor problem (see Example 4.2.2). In this example we also describe the use of distribution trees for specifying asymmetry.

Example 4.4.4. Consider the influence diagram illustrated in Figure 4.20. The decision variable T represents the decision to have a test performed on the components of the advanced reactor, and the chance variable R represents the outcome of this test. The decision variable B represents the decision to either build an advanced reactor, a conventional reactor or no reactor at all. The chance variables A and C represent the state of the advanced reactor and the conventional reactor, respectively. Observe that the model includes a conditioning arc from T to B.

The distribution tree for B encodes that the nuclear regulatory commission will only allow an advanced reactor to be built if either $\mathfrak{i}$) no test is performed on the components of the advanced reactor ($\mathsf{T} = \mathfrak{nt}$), or $\mathfrak{ii}$) a test is performed ($\mathsf{T} = \mathfrak{t}$) and the outcome of the test is either good or excellent ($\mathsf{R} = \mathfrak{g}$ or $\mathsf{R} = \mathfrak{e}$); see Figure 4.21. The distribution tree for B also illustrates the use of *clipping* and *coalescence*; the subtrees for $\mathsf{R} = \mathfrak{b}$, $\mathsf{R} = \mathfrak{g}$ and $\mathsf{R} = \mathfrak{e}$ are clipped conditioned on $\mathsf{T} = \mathfrak{nt}$ since the corresponding scenarios are impossible (a test result is not observed if a test is not initially performed). Similarly, the distribution tree is coalesced since the atomic distributions for $\mathfrak{g}$ and $\mathfrak{e}$ conditioned on $\mathsf{T} = \mathfrak{t}$ are combined. Note that the state space of R is augmented with an artificial state

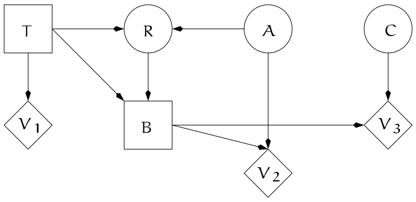

Figure 4.20: An influence diagram representation of the reactor problem. Notice that the model contains a conditioning arc from T to B, which is used when specifying the distribution trees for decision variable B.

nr indicating that no result is observed in case a test is not initially performed.

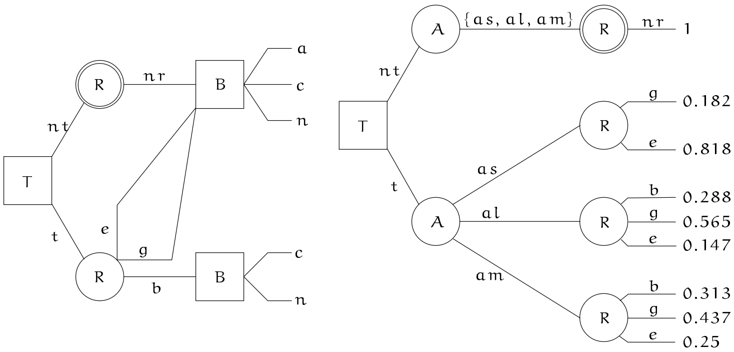

Figure 4.21: The figure illustrates the distribution trees for the decision variable B and the chance variable R, respectively. The double edged circles indicate that the corresponding variables are deterministic.

The distribution tree for R specifies that, no matter the state of the components of the advanced reactor, a test result is not observed if the firm decides not to have a test performed. This is also an example of a *collapsed* distribution tree, i.e., the distribution tree is collapsed over A conditioned on $\mathsf{T} = \mathsf{nt}$.

Distribution trees need not have a conditioning tree. For example, the decision variable T and the chance variables A and C have no parents; hence, their distribution trees are atomic, see Figure 4.22. □

Comparison

As opposed to the asymmetric influence diagram, information about asymmetry cannot be deduced directly from the model as this information is encoded in the

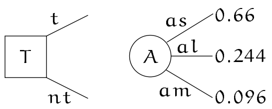

Figure 4.22: The figure illustrates the (atomic) distribution trees for the variables T, A and C, respectively.

distribution trees. For example, in Figure 4.18 we cannot infer that T_2 is only decided upon if we initially decide to have the two-test sequence performed (i.e., $\mathsf{T}_1 = \mathsf{t}_3$). As another example, consider the extended influence diagram representation of the dating problem depicted in Figure 4.23. From this representation we cannot see that e.g. *Night Club?* and *Menu* never occur in the same decision scenario.

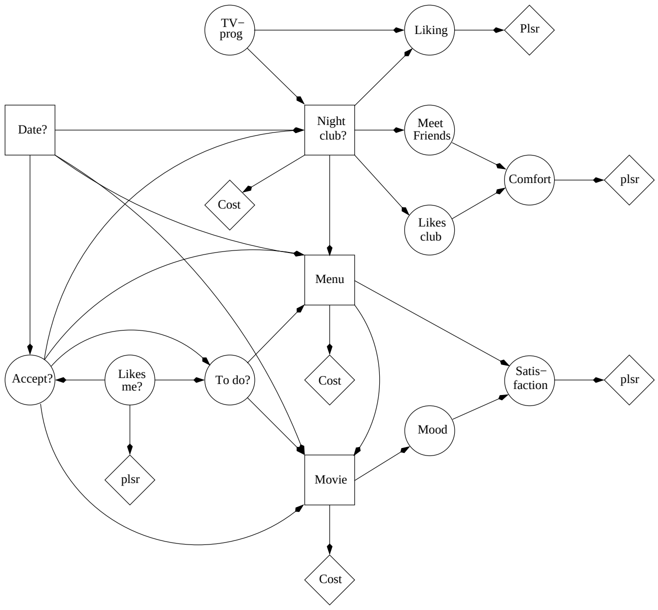

Figure 4.23: An extended influence diagram representation of the dating problem. Note that this representation has been slightly simplified.

The extended influence diagram uses a single representation (distribution trees) to combine both structural and numeric information. This combination may lead to conditionals having larger domains than the corresponding potentials and restrictive functions in the asymmetric influence diagram. For instance, the distribution tree for R (see Figure 4.21) contains T in its domain; however, in the asymmetric influence diagram we only need to specify the partial probability potential $P(R|A)$.

The same information about asymmetry may also occur in different distribution trees e.g. R occurs as a deterministic node in the distribution tree for both B and R. This implies that i) the distribution trees may contain redundant information, and ii) we require consistency when the same information is represented in different distribution trees. Note that this is not the case in the asymmetric influence diagram.

The extended influence diagram is more dependent on the use of artificial states than the asymmetric influence diagram. For instance, in the used car buyer problem we need an artificial state ("no-result") associated with both R_1 and R_2, and an additional artificial state ("no-choice") associated with T_2. Moreover, the extended influence diagram does not support the occurrence of information asymmetry, i.e., situations where the temporal relation between two variables depend on the variables previously observed or decided upon. Note that, similar to the asymmetric valuation network, this type of asymmetry can be modeled by introducing artificial variables corresponding to the possible temporal occurrences of the variable in question.

Finally it is worth to recall the methods for reducing the size of the distribution trees; *coalescence*, *collapsing* and *clipping*. Coalescence and collapsing can be viewed as coalescing in the decision tree framework, whereas clipping has to do with the assessment of the conditional probability distributions, i.e., we do not need a complete assessment if some conditioning states are impossible (having zero probability). These reductions have both representative and computational advantages as they allow a more compact representation of the conditionals; note that these reductions are closely related to the concept of context specific independence (Boutilier et al. 1996). The use of partial potentials and restrictive functions do not allow such reductions, as they only use "clipping" through their specification or indirectly through the use of labels.

4.5 Discussion

In this chapter we have presented a framework, termed asymmetric influence diagrams, for representing asymmetric decision problems. The asymmetric influence diagram is based on the partial influence diagram, and attaches labels with nodes and informational arcs. These labels indicate under which conditions the associated nodes and informational arcs are part of the decision problem, i.e., they are used for representing structural asymmetry. However, the asymmetric influence diagram may still require artificial states (or duplicated variables) for representing certain asymmetric decision problem.

Functional asymmetry, that deals with the possible outcomes of an observation or the legitimate decision options for a decision variable, is encoded at the quantitative level. This is realized through the use of partial probability potentials, partial utility functions and restrictive functions (associated with the decision variables). A partial probability potential allows the modeler to specify that the possible outcomes of a chance variable is dependent on the conditioning state. Similarly, restrictive functions are used to encode that the legitimate decision options for a decision variable is dependent on the different information states.

The asymmetric influence diagram supports the specification of information asymmetry by allowing cycles in the model. The use of cycles has been further pursued by Demirer and Shenoy (2001) in the framework termed *sequential valuation networks*. The sequential valuation network can be seen as a combination of the framework by Covaliu and Oliver (1995) and the asymmetric valuation network (Shenoy 2000); the graphical model combines the sequential decision diagram (which basically corresponds to a clustered decision tree) with valuations similar to those found in the asymmetric valuation network.

Part II

Evaluating decision problems

Chapter 5

Preliminaries and notation

In this chapter we briefly review some existing methods for solving symmetric decision problems. These methods form the basis for an algorithm (presented in Chapter 6) for solving asymmetric decision problems represented in the framework of asymmetric influence diagrams.

5.1 Variable elimination

When solving an influence diagram we seek to determine the maximum expected utility and an optimal strategy for the decision variables involved. As shown in Section 3.2, the maximum expected utility of an influence diagram with chance nodes $\mathcal{U}_C$, decision nodes $\mathcal{U}_D = \{D_1, \ldots, D_n\}$ (indexed according to their ordering) and value nodes $\mathcal{U}_V$ can be calculated as:

$$\rho = \sum_{\mathcal{C}_0} \max_{D_1} \sum_{\mathcal{C}_1} \max_{D_2} \cdots \max_{D_n} \sum_{\mathcal{C}_n} \left[\prod_{C \in \mathcal{U}_C} \phi_C \left(\sum_{V \in \mathcal{U}_V} \psi_V \right) \right]. \tag{5.1}$$

That is, the variables are eliminated in reverse temporal order by alternating sum-marginalizations and max-marginalizations; this type of elimination order is referred to as a *strong* (or a legal) elimination order.

Now, assume that the chance variable A is the first variable to be marginalized out. The set of utility functions $\Psi = \{\psi_V | V \in \mathcal{U}_V\}$ can then be partitioned into two disjoint subsets: Ψ_A which contains the utility functions with A in the domain, and $\Psi \backslash \Psi_A$ which consists of the utility functions not including A in the domain. The set $\Phi = \{\phi_C | C \in \mathcal{U}_C\}$ of probability potentials can be partitioned similarly thus, we have:

$$\begin{aligned}\Psi_A &= \{\psi \mid \psi \in \Psi, A \in \mathrm{dom}(\psi)\}\\ \Phi_A &= \{\phi \mid \phi \in \Phi, A \in \mathrm{dom}(\phi)\}.\end{aligned}$$

Let ϕ_{-A} be the probability potential obtained by eliminating A from the combination of all probability potentials in Φ_A, and let ψ_{-A} be the utility function obtained by eliminating A from the combination of all probability potentials and utility functions in Φ_A and Ψ_A:

$$\begin{aligned}\phi_{-A} &= \sum_A \prod_{\phi \in \Phi_A} \phi\\ \psi_{-A} &= \sum_A \prod_{\phi \in \Phi_A} \phi \left(\sum_{\psi \in \Psi_A} \psi \right).\end{aligned}$$

The marginalization of $A \in \mathcal{C}_n$ in Equation 5.1 can now be written as follows:[1]

$$\begin{aligned}
&\sum_A \left(\prod_{\phi \in \Phi} \phi \right) \left(\sum_{\psi \in \Psi} \psi \right)\\
&\quad = \sum_A \left(\prod_{\phi \in \Phi} \phi \right) \left(\sum_{\psi \in \Psi_A} \psi + \sum_{\psi' \in \Psi \setminus \Psi_A} \psi' \right)\\
&\quad = \left(\prod_{\phi' \in \Phi \setminus \Phi_A} \phi' \right) \left[\sum_A \left(\prod_{\phi \in \Phi_A} \phi \right) \left(\sum_{\psi \in \Psi_A} \psi + \sum_{\psi' \in \Psi \setminus \Psi_A} \psi' \right) \right]\\
&\quad = \left(\prod_{\phi' \in \Phi \setminus \Phi_A} \phi' \right) \left[\sum_A \prod_{\phi \in \Phi_A} \phi \left(\sum_{\psi \in \Psi_A} \psi \right) + \sum_A \prod_{\phi \in \Phi_A} \phi \left(\sum_{\psi' \in \Psi \setminus \Psi_A} \psi' \right) \right]\\
&\quad = \left(\prod_{\phi' \in \Phi \setminus \Phi_A} \phi' \right) \left[\psi_{-A} + \left(\sum_{\psi' \in \Psi \setminus \Psi_A} \psi' \right) \phi_{-A} \right]\\
&\quad = \left(\prod_{\phi' \in \Phi \setminus \Phi_A} \phi' \right) \phi_{-A} \left[\frac{\psi_{-A}}{\phi_{-A}} + \left(\sum_{\psi' \in \Psi \setminus \Psi_A} \psi' \right) \right].
\end{aligned} \tag{5.2}$$

Thus, the elimination of the chance variable A corresponds to the following steps: i) the probability potentials in Φ_A are removed from Φ and replaced by ϕ_{-A}, and ii) the utility functions in Ψ_A are removed from Ψ and replaced by $\frac{\psi_{-A}}{\phi_{-A}}$.

[1] Note that these derivations are similar to those of Madsen and Jensen (1999), and can be seen as a generalization of the results by Dechter (1996).

Consider the case where the decision variable D is the next variable to be eliminated. Analogously to the situation above, we partition the sets Φ and Ψ into the subsets Φ_D, $\Phi \backslash \Phi_D$, Ψ_D and $\Psi \backslash \Psi_D$, and let:

$$\begin{aligned}\phi_{-D} &= \max_D \prod_{\phi \in \Phi_D} \phi \\ \psi_{-D} &= \max_D \prod_{\phi \in \Phi_D} \phi \left(\sum_{\psi \in \Psi_D} \psi \right).\end{aligned}$$

Since D is the next variable to be eliminated, all chance variables succeeding D under the partial order have already been eliminated. Hence, the probability potential $\prod_{\phi \in \Phi_D} \phi$ is a constant function over D; the operation $\max_D \prod_{\phi \in \Phi_D} \phi$ simply corresponds to removing D from the domain of $\prod_{\phi \in \Phi_D} \phi$. Thus, we have:

$$\begin{aligned}&\max_D \left(\prod_{\phi \in \Phi} \phi \right) \left(\sum_{\psi \in \Psi} \psi \right) \\ &= \left(\prod_{\phi' \in \Phi \backslash \Phi_D} \phi' \right) \max_D \left[\prod_{\phi \in \Phi_D} \phi \left(\sum_{\psi \in \Psi_D} \psi \right) + \prod_{\phi \in \Phi_D} \phi \left(\sum_{\psi' \in \Psi \backslash \Psi_D} \psi' \right) \right] \\ &= \left(\prod_{\phi' \in \Phi \backslash \Phi_D} \phi' \right) \left[\psi_{-D} + \phi_{-D} \left(\sum_{\psi' \in \Psi \backslash \Psi_D} \psi' \right) \right] \\ &= \left(\prod_{\phi' \in \Phi \backslash \Phi_D} \phi' \right) \phi_{-D} \left[\frac{\psi_{-D}}{\phi_{-D}} + \left(\sum_{\psi' \in \Psi \backslash \Psi_D} \psi' \right) \right].\end{aligned} \tag{5.3}$$

Note that this result corresponds to the result when marginalizing out a chance variable.

In what follows we shall adopt the generalized marginalization operator $\mathop{\text{M}}$ by Jensen et al. (1994). The definition of this operator depends on the type of variable being marginalized out: Let λ be either a probability potential or a utility function, let A be a chance variable and let D be a decision variable, then:

$$\mathop{\text{M}}_A \lambda = \sum_A \lambda \text{ and } \mathop{\text{M}}_D \lambda = \max_D \lambda.$$

For a set $\mathcal{U}'$ of variables, $\text{M}_{\mathcal{U}'} \lambda$ is defined as a sequence of single variable marginalizations performed in reverse temporal order.

Using this operator, variable elimination can be summarized as follows, see also (Madsen and Jensen 1999):

Algorithm 5.1.1. Let Φ and Ψ be sets of probability potentials and utility functions, respectively. To eliminate a variable X from Φ and Ψ, do:

1. Let $\Phi_X = \{\phi | \phi \in \Phi, X \in \mathrm{dom}(\phi)\}$ and $\Psi_X = \{\psi | \psi \in \Psi, X \in \mathrm{dom}(\psi)\}$.
2. Calculate:

$$\begin{aligned}\phi_{-X} &= \mathop{\text{M}}_{X} \prod_{\phi \in \Phi_X} \phi \\ \psi_{-X} &= \mathop{\text{M}}_{X} \prod_{\phi \in \Phi_X} \phi \left(\sum_{\psi \in \Psi_X} \psi \right).\end{aligned}$$

3. Return $\Phi' = \Phi \backslash \Phi_X \cup \{\phi_{-X}\}$ and $\Psi' = \Psi \backslash \Psi_X \cup \{\frac{\psi_{-X}}{\phi_{-X}}\}$.

□

5.2 The Hugin architecture

The Hugin architecture (Jensen et al. 1994) for evaluating influence diagrams is based on message passing in *strong junction trees*.

A strong junction tree representation of an influence diagram I is identified from its triangulated graph:

1. Remove all informational arcs from I.
2. Moralize I and remove all value nodes; let I^m be the resulting graph.
3. Triangulate I^m by eliminating the variables w.r.t. a strong elimination order, i.e., in reverse temporal order.

The cliques of the triangulated graph is organized in a strong junction tree T, which contains a clique $\mathcal{C}^R$ (referred to as the *strong root*) such that for each pair $(\mathcal{C}_i^*, \mathcal{C}_j^*)$ of adjacent cliques in T, it holds that if $\mathcal{C}_i^*$ is closer to $\mathcal{C}^R$ than $\mathcal{C}_j^*$, then there exists an ordering consistent with $\prec$ s.t. the nodes in $\mathcal{C}_j^* \backslash \mathcal{C}_i^*$ succeed the nodes in $\mathcal{C}_j^* \cap \mathcal{C}_i^*$.

The strong junction tree is initialized by attaching each probability potential and utility function from the influence diagram to a clique that can accommodate it, i.e., its domain is a subset of the nodes in the clique. If no probability potential is assigned to a clique, then this clique is assigned a unity potential; similarly, if no utility function is assigned to a clique, then this clique is assigned a null function. The probability potential of a clique $\mathcal{C}^*$ is then the product of the probability potentials assigned to it (denoted $\phi_{\mathcal{C}^*}$), and the utility function of a clique is the sum of its utility functions (denoted $\psi_{\mathcal{C}^*}$).

Message passing proceeds by absorbing messages from the leaves towards the strong root. A message can be passed from a clique $\mathcal{C}^*$ to its parent clique as soon as $\mathcal{C}^*$ has received messages from all of its children. Hence, messages can be passed independently in the different branches of the strong junction tree.

Let $\mathcal{C}_i^*$ and $\mathcal{C}_j^*$ be neighbouring cliques with separator $\mathcal{S}_{ij}$. Absorption from $\mathcal{C}_j^*$ to $\mathcal{C}_i^*$ then consists of the following steps:

1. Calculate:

$$\phi_{\mathcal{S}_{ij}} = \underset{\mathcal{C}_j^* \setminus \mathcal{S}_{ij}}{\mathsf{M}} \phi_{\mathcal{C}_j^*} \text{ and } \psi_{\mathcal{S}_{ij}} = \underset{\mathcal{C}_j^* \setminus \mathcal{S}_{ij}}{\mathsf{M}} \phi_{\mathcal{C}_j^*} \cdot \psi_{\mathcal{C}_j^*}.$$

2. Assign $\mathcal{C}_i^*$ the potentials:

$$\phi'_{\mathcal{C}_i^*} = \phi_{\mathcal{C}_i^*} \cdot \phi_{\mathcal{S}_{ij}} \text{ and } \psi'_{\mathcal{C}_i^*} = \psi_{\mathcal{C}_i^*} + \frac{\psi_{\mathcal{S}_{ij}}}{\phi_{\mathcal{S}_{ij}}}.$$

Note that a message passed between two cliques contains both a probability potential and a utility function.

After message passing, the optimal policy for a decision variable D can be found from the potentials associated with the clique $\mathcal{C}^*$ which is closest to the strong root and contains D; the variables being required for D is a subset of the variables in $\mathcal{C}^*$. The fact that the required past may be a proper subset of the informational predecessors (see Section 3.3.4) is, to some extent, also reflected in the strong junction tree. For example, in the strong junction tree representation of the influence diagram depicted in Figure 3.3, we can infer that E and F are the only nodes that are (possibly) required for D_2 and D_3, respectively (see Figure 5.1).

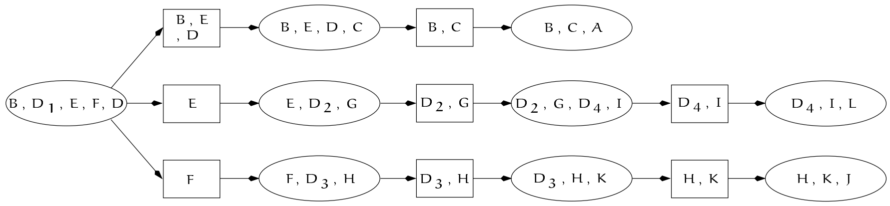

Figure 5.1: A strong junction tree representation of the influence diagram shown in Figure 3.3; cliques are drawn as ellipses and separators are drawn as rectangles.

5.3 The LAZY propagation architecture

In the LAZY propagation architecture (Madsen and Jensen 1999), the principal idea is to maintain a decomposition of the potentials and to postpone computations until they are required.

Similar to the HUGIN architecture, the LAZY propagation architecture is based on message passing in a strong junction tree representation of the influence diagram. Madsen and Jensen (1999) also indicate how the principles of lazy evaluation can be applied to e.g. valuation based systems (Shenoy 1992), and recently the LAZY propagation architecture has been employed in the framework of limited memory influence diagrams (Madsen and Nilsson 2001).

In the LAZY propagation architecture, the strong junction tree T is initialized by assigning the utility functions and the probability potentials to the cliques of T. However, as opposed to the HUGIN architecture, the potentials are not combined; hence, each clique is associated with a set of probability potentials and a set of utility functions. Message passing proceeds by recursively absorbing messages from the leaves towards the strong root: Let $\mathcal{C}_i^*$ and $\mathcal{C}_j^*$ be adjacent cliques and let $\mathcal{S}_{ij}$ be the separator connecting $\mathcal{C}_i^*$ and $\mathcal{C}_j^*$. Based on Algorithm 5.1.1, absorption from $\mathcal{C}_j^*$ to $\mathcal{C}_i^*$ is defined as follows:

1. Let $\mathcal{R}_{\mathcal{S}_{ij}} = \Phi_{\mathcal{C}_j^*} \cup \Psi_{\mathcal{C}_j^*} \cup \left(\bigcup_{\mathcal{S}' \in \mathrm{ch}(\mathcal{C}_j^*)} \Phi'_{\mathcal{S}'} \cup \Psi'_{\mathcal{S}'} \right)$.
2. Let $\Phi'_{\mathcal{S}_{ij}}$ and $\Psi'_{\mathcal{S}_{ij}}$ be the probability potentials and the utility functions obtained by eliminating the variables $\{\mathsf{X} | \mathsf{X} \notin \mathcal{S}_{ij}\}$ from $\mathcal{R}_{\mathcal{S}_{ij}}$ (Algorithm 5.1.1).
3. Associate $\Phi'_{\mathcal{S}_{ij}}$ and $\Psi'_{\mathcal{S}_{ij}}$ with $\mathcal{S}_{ij}$; $\Phi'_{\mathcal{S}_{ij}}$ and $\Psi'_{\mathcal{S}_{ij}}$ are the probability potentials and the utility functions absorbed from $\mathcal{C}_j^*$ to $\mathcal{C}_i^*$.

Madsen and Jensen (1999) also consider possible efficiency improvements by exploiting, e.g., the occurrence of barren variables. They also show how the LAZY propagation architecture supports common tasks such as myopic value of information analysis (Dittmer and Jensen 1997).

Chapter 6

Evaluating asymmetric influence diagrams

In this chapter we present an algorithm for solving asymmetric influence diagrams. The algorithm initially decomposes the asymmetric decision problem into a structure of symmetric subproblems organized as a rooted tree. A solution to the decision problem can then be found by propagating from the leaves towards the root, using existing evaluation methods to solve the symmetric subproblems.

The content of this chapter is based on (Nielsen and Jensen 1999a) and (Nielsen and Jensen 2000).

6.1 Preliminaries

Solving an asymmetric influence diagram I is the same as determining an optimal policy for each decision variable in I. However, as opposed to the partial influence diagram, the informational predecessors for a decision variable D (and, in particular, the required past for D) may vary depending on the context in which D appears (see Section 4.3.3). Thus, we define a strategy based on the contexts for each decision variable in I:

Definition 6.1.1. A *strategy* for an asymmetric influence diagram I is a set of functions $\bar{\Delta} = \{\bar{\delta}_D | D \in \mathcal{U}_D\}$, where $\bar{\delta}_D$ is a *policy* given by:

$$\bar{\delta}_D : \mathrm{sp}(\mathrm{pred}(D)_{\bar{s}}) \to \mathrm{sp}(D), \forall \bar{s} \in \Omega_D,$$

where Ω_D is the set of contexts in which D can occur (see Section 4.3.3) and $\mathrm{pred}(D)_{\bar{s}}$ is the set of variables preceding D under the partial order induced by the asymmetric influence diagram reduced from I w.r.t. $\bar{s}$. A strategy that maximizes the expected utility is termed an *optimal strategy* (denoted Δ), and a policy that maximizes the expected utility for decision D w.r.t. each context $\bar{s} \in \Omega_D$ is termed an *optimal policy* for D (denoted δ).

A well-defined asymmetric influence diagram (specified in Section 6.2) can in principle be solved by first unfolding it into a decision tree, and then use the *average-out and fold-back* algorithm (Raiffa and Schlaifer 1961) on that tree. The average-out and fold-back algorithm is based on the backwards recursion method of dynamic programming (Bellman 1957), and works itself backwards from the nodes having only leaves (value nodes) as children: When a chance node A is visited, it is assigned the expected utility:

$$\rho(A, past(A)) = \sum_{a \in sp(A)} P(A = a | past(A)) \rho(N(A = a, past(A))),$$

where $N(A = a, past(A))$ is the node defined by the arcs labeled with $(A = a, past(A))$; if $N(A = a, past(A))$ is a value node, then $\rho(N(A = a, past(A)))$ is simply the utility associated with $N(A = a, past(A))$. When a decision node D is visited, it is assigned the (maximum) expected utility:

$$\rho(D, past(D)) = \max_{d \in sp(D)} \rho(N(D = d, past(D))),$$

as well as the decision option maximizing the expected utility. An optimal strategy can then be found from the decision options associated with the decision nodes.

The required probabilities (i.e., $P(A = a | past(A))$) can be calculated from the partial probability potentials specified by the realization of the asymmetric influence diagram. That is, the asymmetric influence diagram can be seen as a model of the uncertainty associated with the chance nodes in the decision tree; this is somewhat similar to the approach found in (Call and Miller 1990). Unfortunately, this "brute force" approach creates an unnecessary large decision tree in case the original decision problem contains symmetric substructures, and it furthermore fails to exploit the occurrence of such substructures during the evaluation.

6.2 Evaluating asymmetric influence diagrams

To overcome the shortcomings of the algorithm described above, we propose another algorithm which basically i) decomposes the decision problem into a collection of symmetric subproblems organized in a rooted tree, and ii) propagates from the leaves towards the root using existing evaluation methods to solve the "smaller" symmetric subproblems. The decomposition is performed by reducing the asymmetric influence diagram w.r.t. the possible states of its initial split variable; this reduction is applied recursively until no split variables remain.

Example 6.2.1. Consider the used car buyer problem represented by the asymmetric influence diagram I depicted in Figure 4.9. The test-decision T_1 is the initial split variable in I; thus, I can be reduced w.r.t. $\mathsf{T}_1 = \mathsf{t}_1$, $\mathsf{T}_1 = \mathsf{t}_2$, $\mathsf{T}_1 = \mathsf{t}_3$ and $\mathsf{T}_1 = \mathsf{nt}$ (see Figure 6.1). The asymmetric influence diagrams $\mathrm{I}[\mathsf{T}_1 \mapsto \mathsf{t}_1]$, $\mathrm{I}[\mathsf{T}_1 \mapsto \mathsf{t}_2]$ and $\mathrm{I}[\mathsf{T}_1 \mapsto \mathsf{nt}]$ do not contain any split variables hence, they are not further reduced. On the other hand, the test-decision T_2 is the initial split variable in $\mathrm{I}[\mathsf{T}_1 \mapsto \mathsf{t}_3]$, so $\mathrm{I}[\mathsf{T}_1 \mapsto \mathsf{t}_3]$ can be reduced w.r.t. $\mathsf{T}_2 = \mathsf{s}$ and $\mathsf{T}_2 = \mathsf{c}$; the resulting structures ($\mathrm{I}[\mathsf{T}_1 \mapsto \mathsf{t}_3][\mathsf{T}_2 \mapsto \mathsf{s}]$ and $\mathrm{I}[\mathsf{T}_1 \mapsto \mathsf{t}_3][\mathsf{T}_2 \mapsto \mathsf{c}]$) do not contain any split variables.

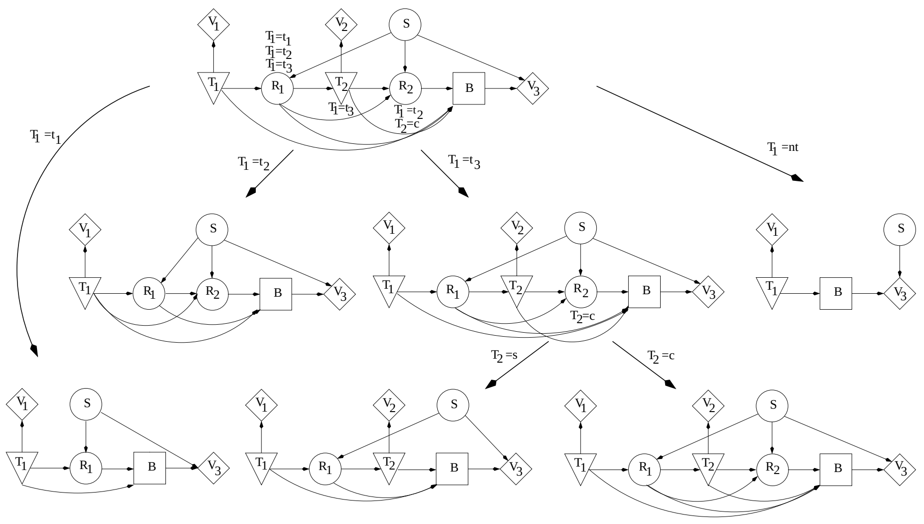

Figure 6.1: A decomposition tree for the asymmetric influence diagram representation of the used car buyer problem.

□

Example 6.2.2. In the asymmetric influence diagram representation of the reactor problem, the test-decision T is the initial split variable (see Figure 4.10). By reducing this structure w.r.t. $\mathsf{T} = \mathsf{nt}$, the label associated with R is unsatisfied; hence, the asymmetric influence diagram $\mathrm{I}[\mathsf{T} \mapsto \mathsf{nt}]$ does not contain any split variables (see Figure 6.2). Conversely, the label associated with R is satisfied for $\mathsf{T} = \mathsf{t}$ thus, R is the initial split variable in $\mathrm{I}[\mathsf{T} \mapsto \mathsf{t}]$; R may restrict the legitimate decision options for B. Reducing $\mathrm{I}[\mathsf{T} \mapsto \mathsf{t}]$ w.r.t. the states of R produces the decision problems $I[T \mapsto t][R \mapsto b]$, $I[T \mapsto t][R \mapsto g]$ and $I[T \mapsto t][R \mapsto e]$ which are identical in structure but differ on the legitimate decision options associated with B.

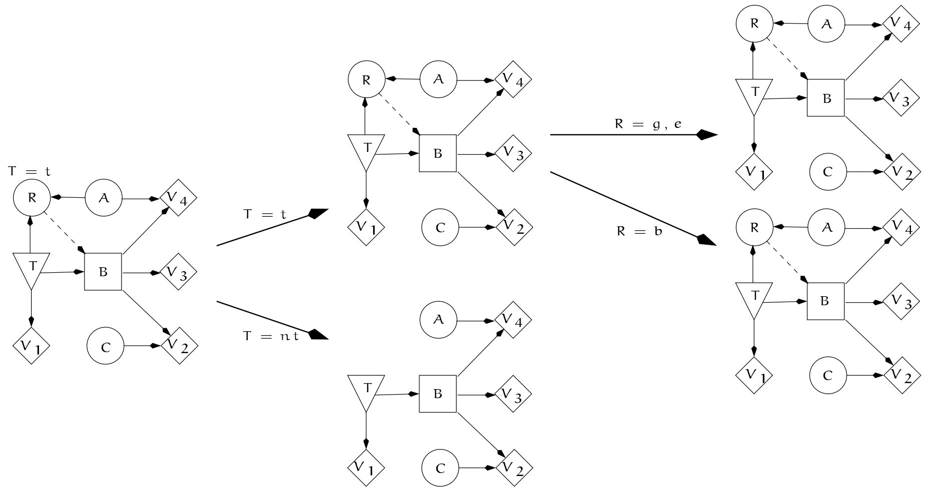

Figure 6.2: The figure illustrates the decomposition tree for the reactor problem depicted in Figure 4.10. Note that reducing $I[T \mapsto t]$ w.r.t. the restrictive variable R produces subproblems that are structurally identical.

□

Based on the decomposition tree for an asymmetric influence diagram, an optimal strategy can now be found by iteratively eliminating the so-called free variables in each of the subproblems:

Definition 6.2.1. Let $I[S^1_\nabla \mapsto s_1]$ be an asymmetric influence diagram reduced from the asymmetric influence diagram I. The variable X is said to be *free* in $I[S^1_\nabla \mapsto s_1]$ if $S^1_\nabla \prec X$ and $\forall S \in \mathcal{S}_{I[S^1_\nabla \mapsto s_1]} : S \not\prec X$, where $\prec$ is the partial order induced by $I[S^1_\nabla \mapsto s_1]$. If X is not free in $I[S^1_\nabla \mapsto s_1]$, then X is said to be *bound* in $I[S^1_\nabla \mapsto s_1]$.

The evaluation of an asymmetric influence diagram I is initiated by invoking Algorithm 6.2.1 on I; note that in the following algorithms we exploit that instantiating the initial split variable in an asymmetric influence diagram produces another asymmetric influence diagram with a unique initial split variable.

Algorithm 6.2.1. Let I be an asymmetric influence diagram and let S^1_∇ be the initial split variable in I. If Algorithm 6.2.1 is invoked on I, then

1. Invoke Algorithm 6.2.1 on $I[S^1_\nabla \mapsto s_1]$, for all $s_1 \in sp(S^1_\nabla)$.
2. Absorb the probability potentials and the utility functions (Algorithm 6.2.2) from $I[S^1_\nabla \mapsto s_1]$ to I, for all $s_1 \in sp(S^1_\nabla)$.
3. If ψ is a utility function absorbed from $I[S^1_\nabla \mapsto s^i_1]$ to I and $S^1_\nabla \notin dom(\psi)$, then condition ψ on $S = s^i_1$ if ψ is not absorbed from $I[S^1_\nabla \mapsto s^j_1]$ to I ($j \neq i$), see Figure 6.3.

□

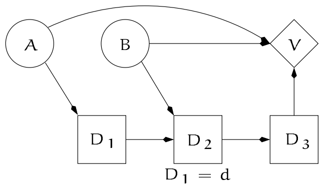

Figure 6.3: The occurrence of D_2 is dependent on the state of the split variable D_1. Thus, the utility function produced by the elimination of D_3, B and D_2 is dependent on D_1; the relative temporal order of B and D_3 vary w.r.t. the state of D_1.

The following algorithm is used by Algorithm 6.2.1 and describes the absorption of probability potentials and utility functions.

Algorithm 6.2.2. Let $I[S^i_{\bar{s}_{i-1}} \mapsto \bar{s}_i]$ be an asymmetric influence diagram and let S be the initial split variable in $I[S^i_{\bar{s}_{i-1}} \mapsto \bar{s}_i]$. If Algorithm 6.2.2 is invoked on $I[S^i_{\bar{s}_{i-1}} \mapsto \bar{s}_i][S \mapsto s]$ from $I[S^i_{\bar{s}_{i-1}} \mapsto \bar{s}_i]$, then:

1. Let $\mathcal{F}$ be the set of free variables in $I[S^i_{\bar{s}_{i-1}} \mapsto \bar{s}_i][S \mapsto s]$.
2. Set:

$$\begin{aligned} \Phi_\mathcal{F} &= \{\phi | \phi \in \Phi_{I[S^i_{\bar{s}_{i-1}} \mapsto \bar{s}_i][S \mapsto s]} \text{ and } \mathcal{F} \cap dom(\phi) \neq \emptyset\} \\ \Psi_\mathcal{F} &= \{\psi | \psi \in \Psi_{I[S^i_{\bar{s}_{i-1}} \mapsto \bar{s}_i][S \mapsto s]} \text{ and } \mathcal{F} \cap dom(\psi) \neq \emptyset\}. \end{aligned}$$

3. Eliminate the variables in $\mathcal{F}$ from $\Phi_\mathcal{F} \cup \Psi_\mathcal{F}$ w.r.t. the partial order induced by $I[S^i_{\bar{s}_{i-1}} \mapsto \bar{s}_i][S \mapsto s]$ (Algorithm 6.2.3). Let Φ' and Ψ' be the sets of probability potentials and utility functions obtained.
4. For each $\lambda \in \Phi_\mathcal{F} \cup \Psi_\mathcal{F}$, remove λ from $I[S^j_{\bar{s}_{j-1}} \mapsto \bar{s}_j]$ and associate $\Phi' \cup \Psi'$ with I and $I[S^j_{\bar{s}_{j-1}} \mapsto \bar{s}_j]$, for all $1 \leq j \leq i$.

□

Observe that in step 4, we update the probability potentials and the utility functions associated with each structure that lie on the path from the root to $I[S^i_{\bar{s}_{i-1}} \mapsto \bar{s}_i]$ in the decomposition tree for I. Note also that the conditioning of ψ (in Algorithm 6.2.1) therefore applies to all of these structures as well.

The algorithm below describes the elimination of free variables, and is inspired by the LAZY evaluation architecture (Madsen and Jensen 1999). However, we conjecture that, in principle, any elimination algorithm can be used.

Algorithm 6.2.3. Let I be an asymmetric influence diagram, and let Φ and Ψ be the sets of probability potentials and utility functions associated with I. If Algorithm 6.2.3 is invoked on $\Phi \cup \Psi$ w.r.t. a variable X, then:

1. Set $\Phi_X = \{\phi | \phi \in \Phi, X \in \mathrm{dom}(\phi)\}$ and $\Psi_X = \{\psi | \psi \in \Psi, X \in \mathrm{dom}(\psi)\}$.
2. Calculate:

$$\begin{aligned} \phi_{-X} &= \bigvee_X \prod_{\phi \in \Phi_X} \phi \\ \psi_{-X} &= \bigvee_X \prod_{\phi \in \Phi_X} \phi \left(\sum_{\psi \in \Psi_X} \psi \right), \end{aligned}$$

   where $\bigvee$ is the generalized marginalization operator defined in Section 5.1.
3. Return $\Phi' = \Phi \backslash \Phi_X \cup \{\phi_{-X}\}$ and $\Psi' = \Psi \backslash \Psi_X \cup \{\frac{\psi_{-X}}{\phi_{-X}}\}$.

$\square$

During the evaluation, the decision option maximizing the utility function, from which a decision variable D is eliminated, should be recorded as the optimal policy for D w.r.t. the context in question.

Having described the evaluation of an asymmetric influence diagram, we define the concept of a *well-defined* asymmetric influence diagram. The definition is based on the notion of a significant chance variable, which can be adopted from the partial influence diagram framework by considering the admissible total orderings of the free variables in each of the asymmetric influence diagrams produced during the decomposition.

Definition 6.2.2. An asymmetric influence diagram I is said to *define a decision scenario* if:

- for all split configurations $\bar{s}$, there does not exist a free chance variable A and a free decision variable D in $I[\mathcal{S}''_I \mapsto \bar{s}]$ s.t. A is significant for D; $\bar{s}$ is a configuration over the variables $\mathcal{S}''_I \subseteq \mathcal{S}^R_I$, and

- for any decision variable D and for each context $\bar{s} = (\bar{\omega}, \bar{x})$ for D, there does not exist two restrictive functions γ_D^1 and γ_D^2 s.t. $\mathsf{dom}(\gamma_D^1) \subseteq \mathsf{pa}(D)_{\bar{s}}^R$ and $\mathrm{dom}(\gamma_D^2) \subseteq \mathsf{pa}(D)_{\bar{s}}^R$, where $\mathsf{pa}(D)_{\bar{s}}^R$ is the set of restrictive variables which are present in I given $\bar{s}$.

The structural characterization of the chance variables being significant for a given decision variable can also be adopted from the partial influence diagram framework. However, a utility function ψ should also be considered relevant for a split variable S if ψ is relevant for a future decision D and the state of S either determines if D is present in the resulting decision problem, or determines the set of variables being required for D (see Figure 6.3).

In order to prove the soundness of the proposed evaluation algorithm, we need the following lemma.

Lemma 6.2.1. Let I be an asymmetric influence diagram, and let S be the initial split variable in I. If the probability potential ϕ is absorbed from $I[S \mapsto s_i]$ to I and $S \notin \mathsf{dom}(\phi)$, then a probability potential ϕ' is absorbed from $I[S \mapsto s_j]$ to I and $\phi = \phi'$, $\forall s_i, s_j \in \mathsf{sp}(S)$.

Proof. The chance variables succeeding S in $I[S \mapsto s_i]$ and $I[S \mapsto s_j]$ identify the probability potentials involved in the elimination of the variables in $I[S \mapsto s_i]$ and $I[S \mapsto s_j]$, respectively. Thus, we need to show that the probability potentials produced by the elimination of the chance variables which are present in $I[S \mapsto s_i]$ but not in $I[S \mapsto s_j]$ contribute as unity potentials.

Consider a chance variable X that is present in $I[S \mapsto s_i]$ but not in $I[S \mapsto s_j]$, and assume that the elimination of X does not contribute as a unity potential. This implies that there exists a directed path from X to a chance variable Y in $\bar{I}[S \mapsto s_i]$ s.t. $Y \prec S$. However, no such variable Y can exist in $\bar{I}[S \mapsto s_i]$ since the absence of X in $\bar{I}[S \mapsto s_j]$ renders Y missing in I. □

Notice that in the theorem above, the probability potentials ϕ and ϕ' need not be strictly identical. For instance, consider the asymmetric influence diagram depicted in Figure 6.4 and assume that T is binary. The probability potential absorbed from $I[T \mapsto t_2]$ is given by $P(B) = \sum_C P(B|C)P(C)$ and the probability potential absorbed from $I[T \mapsto t_1]$ is given by $P(B|A) = \sum_E \sum_C P(B|C)P(C)P(E|C,A)$. However, $P(B|A) = P(B)$.

Based on the lemma above we now have the following theorem.

Theorem 6.2.1. If Algorithm 6.2.1 is invoked on an asymmetric influence diagram, I, which define a decision scenario, then it computes an optimal policy for each decision variable in I.

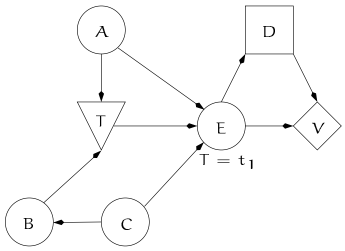

Figure 6.4: The probability potential absorbed from $I[T \mapsto t_1]$ is equivalent (but not identical) to the probability potential absorbed from $I[T \mapsto t_2]$.

Proof. To prove the theorem we need to show that the optimal strategy computed by Algorithm 6.2.1 is equivalent to the optimal strategy found by solving the corresponding decision tree.

Since I defines a decision scenario, a decision tree representation can be found by initially treating non-split variables as split variables; each subproblem in the resulting decomposition tree contains exactly one free variable which, together with the previous splits, uniquely identifies a node in the corresponding decision tree.

From Algorithm 6.2.1 it follows that the optimal strategy is invariant of whether or not non-split variables are treated as split variables. Hence, we only need to show that if non-split variables are treated as split variables, then Algorithm 6.2.1 is equivalent to the average-out and fold-back algorithm. For the leaf structures, this is straightforward: The key observation is that the set of probability potentials associated with these structures can be seen as a probability model for the corresponding nodes in the decision tree representation. For the non-leaf structures we note that the marginalization operation is a value preserving reduction of the probability potentials and the utility functions, and from Lemma 6.2.1 the proof follows immediately.

□

Before presenting an example of the evaluation algorithm, recall that the asymmetric influence diagram requires that all utilities are positive. This restriction ensures that undefined utilities do not influence the optimal policy for a decison variable. To illustrate the problem consider the model depicted in Figure 6.5, where both A and D are assumed to be binary. Let the chance variable A be associated with a deterministic conditional probability potential s.t. the state of A corresponds to the state of D, and specify ψ_V s.t. $\psi_V(A = a_1) = -10$ and

$\psi_V(A = a_2) = \perp$. When marginalizing out A we obtain the utility function:

$$\begin{aligned}\psi(D = d_1) &= -10 \cdot 1 + 0 \cdot \perp = -10 \\ \psi(D = d_2) &= -10 \cdot 0 + 1 \cdot \perp = 0.\end{aligned}$$

Thus, $D = d_2$ is identified as an optimal policy for D. However, this is a result of obtaining an outcome with an undefined utility; the problem is that information about asymmetry is lost when A is marginalized out. Note that it does not suffice simply to marginalize over the legitimate decision options for D.

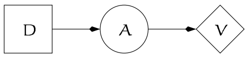

Figure 6.5: The problem with negative utilities.

The following example illustrates the proposed evaluation algorithm by solving the used car buyer problem.

Example 6.2.3. Consider the asymmetric influence diagram representation of the used car buyer problem (depicted in Figure 4.9), and assume the probability potentials specified in Example 4.2.1. The utility function ψ_{V_3} is specified in Table 6.1, and the utility functions associated with V_1 and V_2 are given by $\psi_{V_1}(T_1) = (0(nt); -9(t_1); -13(t_2); -10(t_3))$ and $\psi_{V_2}(T_2) = (0(s); -4(c))$, respectively.

$\psi_{V_3}(S, B)$	$B = b$	$B = g$	$B = \neg b$
$S = p$	60	20	0
$S = l$	-100	40	0

Table 6.1: The utility function $\psi_{V_3}(S, B)$.

Since some of the utilities are negative, we need to add a positive constant c to all the utilities before evaluating the decision problem (thereby making them positive); the maximum expected utility function for a decision variable can still be found by simply subtracting c from the values computed during the evaluation.[1] Note that in the following calculations we will sometimes omit the intermediate results produced by the elimination of a single variable, and instead eliminate several variables in one step. Note also that the utilities given below have been transformed to their original scale.

[1] Actually, for this particular example the problem of having negative utilities does not apply.

The evaluation is initiated by eliminating the free variables in each of the leaf structures in the decomposition tree (see Figure 6.1).

Eliminating the free variables in $I[T_1 \mapsto t_1]$: $\mathcal{F} = \{R_1, B, S\}$ constitutes the set of free variables in $I[T_1 \mapsto t_1]$, and we therefore have $\Psi_{\mathcal{F}} = \{\psi_{V_3}(S, B)\}$ and $\Phi_{\mathcal{F}} = \{P(S), P(R_1|S)\}$.

The free variables are eliminated in reverse temporal order; hence, we start off by marginalizing out the chance variable S. Marginalizing out S produces the probability potential $\phi_{-S}(R_1) = \sum_S P(S)P(R_1|S)$ and the utility function $\psi_{-S}(R_1, B) = \sum_S P(S)P(R_1|S)\psi_{V_3}(S, B)$. Thus, $\Phi_{\mathcal{F}}$ is replaced by $\phi_{-S}(R_1)$ and $\Psi_{\mathcal{F}}$ is replaced by $\psi'(R_1, B) = \frac{\psi_{-S}(R_1,B)}{\phi_{-S}(R_1)}$:

$$\begin{aligned} \Psi'_{\mathcal{F}} &= \Psi_{\mathcal{F}} \backslash \{\psi_{V_3}(S, B)\} \cup \{\psi'(R_1, B)\} \\ \Phi'_{\mathcal{F}} &= \Phi_{\mathcal{F}} \backslash \{P(S), P(R_1|S)\} \cup \{\phi_{-S}(R_1)\}. \end{aligned} \tag{6.4}$$

The elimination of the decision variable B involves only the utility function $\psi'(R_1, B)$; hence, $\Psi'_{\mathcal{F}}$ is simply replaced by $\Psi''_{\mathcal{F}} = \{\psi_{-B}(R_1)\}$, where $\psi_{-B}(R_1) = \max_B \psi'(R_1, B)$. As an intermediate result we obtain a partial optimal policy for B specified by $\delta_B(R_1 = d_1, T_1 = t_1) = g(32)$ and $\delta_B(R_1 = \neg d_1, T_1 = t_1) = b(44)$; the numbers in parenthesis correspond the maximum expected utilities. Finally, the elimination of R_1 yields $\psi(T_1 = t_1) = \sum_{R_1} \phi_{-S}(R_1)\psi_{-B}(R_1) = 41.6$.

The derivation above can be summarized as:

$$\psi(T_1 = t_1) = \sum_{R_1} \left(\left(\sum_S (P(S)P(R_1|S)) \right) \max_B \left(\frac{\sum_S P(S)P(R_1|S)\psi_{V_3}(B, S)}{\sum_S P(S)P(R_1|S)} \right) \right).$$

Eliminating the free variables in $I[T_1 \mapsto t_2]$: The set of free variables in $I[T_1 \mapsto t_2]$ is given by $\mathcal{F} = \{R_1, R_2, B, S\}$; thus, $\Psi_{\mathcal{F}} = \{\psi_{V_3}(S, B)\}$ and $\Phi_{\mathcal{F}} = \{P(S), P(R_2|S, R_1), P(R_1|S)\}$. Eliminating the chance variable S produces the probability potential $\phi_{-S}(R_1, R_2) = \sum_S P(S)P(R_1|S)P(R_2|R_1, S)$ together with the utility function $\psi_{-S}(R_1, R_2, B) = \sum_S P(S)P(R_1|S)P(R_2|R_1, S)\psi_{V_3}(S, B)$; hence, we set $\Phi'_{\mathcal{F}} = \{\phi_{-S}(R_1, R_2)\}$ and $\Psi'_{\mathcal{F}} = \left\{ \frac{\psi_{-S}(R_1,R_2,B)}{\phi_{-S}(R_1,R_2)} \right\}$.

The elimination of B produces the utility function $\psi_{-B}(R_1, R_2) = \max_B \psi'(R_1, R_2, B)$, where $\psi'(R_1, R_2, B) = \frac{\psi_{-S}(R_1,R_2,B)}{\phi_{-S}(R_1,R_2)}$; the partial optimal policy for B is specified in Table 6.2. The sets of probability potentials and utility functions are updated as $\Phi''_{\mathcal{F}} = \{\phi_{-S}(R_1, R_2)\}$ and $\Psi''_{\mathcal{F}} = \{\psi_{-B}(R_1, R_2)\}$.

$\delta_B(T_1 = t_2, R_1, R_2)$	$R_1 = d_1$	$R_1 = \neg d_1$
$R_2 = d_2$	$g(40)$	$g(28)$
$R_2 = \neg d_2$	$g(28)$	$b(53.6)$

Table 6.2: The partial optimal policy for B given $T_1 = t_2$.

Eliminating the chance variable R_2 produces the probability potential $\phi_{-R_2}(R_1) = \sum_{R_2} \phi_{-S}(R_1, R_2)$ and the utility function $\psi_{-R_2}(R_1) = \sum_{R_2} \phi_{-S}(R_1, R_2)\psi_{-B}(R_1, R_2)$; hence, $\Phi''_{\mathcal{F}}$ and $\Psi''_{\mathcal{F}}$ are replaced by $\phi_{-R_2}(R_1)$ and $\psi'(R_1) = \frac{\psi_{-R_2}(R_1)}{\phi_{-R_2}(R_1)}$, respectively. Finally, eliminating R_1 produces $\psi(T_1 = t_2) = \sum_{R_1} \phi_{-R_2}(R_1)\psi'(R_1) = 45.87$; the derivations can be summarized as:

$$\begin{aligned}\psi(T_1 = t_2) = \sum_{R_1}\sum_{R_2} \Bigg(\Bigg(\sum_{S} (P(S)P(R_1|S)P(R_2|S, R_1)) \Bigg) \\ \max_{B} \left(\frac{\sum_S P(S)P(R_1|S)P(R_2|S, R_1)\psi_{V_3}(B, S)}{\sum_S P(S)P(R_1|S)P(R_2|S, R_1)} \right) \Bigg).\end{aligned} \tag{6.6}$$

Eliminating the free variables in $I[T_1 \mapsto t_3][T_2 \mapsto s]$: The set of free variables in $I[T_1 \mapsto t_3][T_2 \mapsto s]$ (i.e., $\mathcal{F} = \{B, S\}$) is a subset of the free variables in $I[T_1 \mapsto t_1]$, and since the variables are eliminated in the same order, the calculations are identical. That is, we return the maximum expected utility function for B conditioned on $T_2 = s$, i.e., the utility function specified by $\psi^*(R_1 = d_1, T_2 = s) = 32(g)$ and $\psi^*(R_1 = \neg d_1, T_2 = s) = 44(b)$. However, as opposed to the previous steps we also return the probability potential $P(R_1)$ produced by the eliminations.

Eliminating the free variables in $I[T_1 \mapsto t_3][T_2 \mapsto c]$: Analogously to the evaluation of $I[T_1 \mapsto t_3][T_2 \mapsto s]$, we can reuse some of the calculations performed previously; the free variables in $I[T_1 \mapsto t_3][T_2 \mapsto c]$ is a subset of the free variables in $I[T_1 \mapsto t_2]$. Thus, the utility function returned to $I[T_1 = t_3]$ is given by:

$$\begin{aligned}\psi^*(R_1, T_2 = c) &= \sum_{R_2} P(R_2|R_1) \left(\max_{B} \left(\frac{\sum_S P(S)P(R_1|S)P(R_2|S, R_1)\psi_{V_3}(B, S)}{\sum_S P(S)P(R_1|S)P(R_2|S, R_1)} \right) \right) \\ &= (32; 49.34),\end{aligned} \tag{6.7}$$

and analogously to the evaluation of $I[T_1 \mapsto t_3][T_2 \mapsto s]$ we also return the probability potential $P(R_1)$. Note that the optimal policy for B is similar to the one specified for B in $I[T_1 \mapsto t_2]$.

Eliminating the free variables in $I[T_1 \mapsto nt]$: The variables S and B constitute the free variables in $I[T_1 \mapsto nt]$ hence, the utility function returned to I is simply $\psi(T_1 = nt) = \max_B \sum_S P(S)\psi_{V_3}(B, S) = 28$; the partial optimal policy for B is given by $\delta_B(T_1 = nt) = b(28)$.

Eliminating the free variables in $I[T_1 \mapsto t_3]$: The free variables in $I[T_1 \mapsto t_3]$ are T_2 and R_1, which occur in the domain of $\psi_{V_2}(T_2)$, together with $P(R_1)$, $\psi^*(R_1, T_2 = s)$ and $\psi^*(R_1, T_2 = c)$ produced by the elimination of the free variables in $I[T_1 \mapsto t_3][T_2 \mapsto s]$ and $I[T_1 \mapsto t_3][T_2 \mapsto c]$. By eliminating the free variables from these functions we obtain the utility function $\psi(T_1 = t_3) = 42.67$; the optimal policy for T_2 is given by $\delta_{T_2}(R_1 = d_1, T_1 = t_3) = s(32)$ and $\delta_{T_2}(R_1 = \neg d_1, T_1 = t_3) = c(45.34)$.

The optimal policy for T_1 can now be determined by combining $\psi_{V_1}(T_1)$ with the utility functions returned, i.e., $\delta_{T_1} = t_2(32.87)$. □

6.3 Comparison

In this section we compare the proposed evaluation algorithm with the evaluation of both asymmetric valuation networks (Shenoy 2000) and extended influence diagrams (Smith et al. 1993).

6.3.1 Comparison with asymmetric valuation networks

The algorithm for solving asymmetric valuation networks is know as the *fusion algorithm*, and is a generalization of the fusion algorithm for solving symmetric valuation networks (Shenoy 1992). The main difference is due to the indicator valuations, since the numerical computations are only performed on the effective state spaces defined by these valuations. For instance, the combination of two probability valuations ρ_1 and ρ_2 is defined as:

$$(\rho_1 \otimes \rho_2)(\bar{x}) = (\rho_2 \otimes \rho_1)(\bar{x}) = \rho_1\left(\bar{x}^{\downarrow \mathrm{dom}(\rho_1)}\right)\rho_2\left(\bar{x}^{\downarrow \mathrm{dom}(\rho_2)}\right),$$

where $\bar{x} \in \Omega_{\mathrm{dom}(\rho_1)\cup\mathrm{dom}(\rho_2)}$; $\Omega_{\mathrm{dom}(\rho_1)\cup\mathrm{dom}(\rho_2)}$ is the effective state space for the variables in $\mathrm{dom}(\rho_1) \cup \mathrm{dom}(\rho_2)$. Similarly, marginalization of a probability valuation ρ over a chance variable $X \in \mathrm{dom}(\rho)$ (denoted $\rho^{\downarrow(\mathrm{dom}(\rho)\setminus\{X\})}$) is defined as:

$$\rho^{\downarrow(\mathrm{dom}(\rho)\setminus\{X\})}(\bar{c}) = \sum \left\{\rho(\bar{c}, x) | x \in sp(X), (\bar{c}, x) \in \Omega_{\mathrm{dom}(\rho)}\right\},$$

for all $\bar{c} \in \Omega_{\mathrm{dom}(\rho)\setminus\{X\}}$.

The operations for adding, dividing and max-marginalizing valuations are defined in a similar way, see (Shenoy 2000) for details. In what follows we adopt the notation by Shenoy (2000) when referring to these operations, i.e., for marginalization we use the projection notation ($\downarrow$) and combination is denoted by $\otimes$ (no matter the type of valuations involved).

To describe the fusion algorithm, consider a set of valuations consisting of utility valuations $\mathcal{V}$, probability valuations Γ and indicator valuations Υ. For a variable X we denote by $\mathcal{V}_X$, Γ_X and Υ_X the utility, probability and indicator valuations including X in their domain. The fusion operation $\mathrm{Fus}_X(\mathcal{V} \cup \Gamma \cup \Upsilon)$ w.r.t. the valuations $\mathcal{V} \cup \Gamma \cup \Upsilon$ is then defined differently depending on the type of variable X. If X is a chance variable, then:

$$\begin{aligned}\mathrm{Fus}_X(\mathcal{V} \cup \Gamma \cup \Upsilon) = & \left\{ \left[\nu \otimes \left(\frac{\rho}{\rho^{\downarrow(\mathrm{dom}(\Gamma_X \cup \Upsilon_X) \setminus \{X\})}} \right) \right]^{\downarrow(\mathrm{dom}(\mathcal{V}_X \cup \Gamma_X \cup \Upsilon_X) \setminus \{X\})} \right\} \\ & \cup \left\{ \rho^{\downarrow(\mathrm{dom}(\Gamma_X \cup \Upsilon_X) \setminus \{X\})} \right\} \cup \{\mathcal{V} \setminus \mathcal{V}_X\} \cup \{\Upsilon \setminus \Upsilon_X\} \cup \{\Gamma \setminus \Gamma_X\},\end{aligned} \tag{6.10}$$

where

$$\begin{aligned}\rho &= (\otimes\{\iota | \iota \in \Upsilon_X\}) \otimes (\otimes\{\rho | \rho \in \Gamma_X\}); \\ \nu &= \otimes\{\nu | \nu \in \mathcal{V}_X\}.\end{aligned}$$

If X is a decision variable, then:

$$\begin{aligned}\mathrm{Fus}_X(\mathcal{V} \cup \Gamma \cup \Upsilon) = & \left\{ \nu^{\downarrow(\mathrm{dom}(\mathcal{V}_X) \setminus \{X\})} \right\} \cup \left\{ (\rho \otimes \zeta_X)^{\downarrow(\mathrm{dom}(\mathcal{V}_X \cup \Gamma_X \cup \Upsilon_X) \setminus \{X\})} \right\} \\ & \cup \{\mathcal{V} \setminus \mathcal{V}_X\} \cup \{\Upsilon \setminus \Upsilon_X\} \cup \{\Gamma \setminus \Gamma_X\},\end{aligned} \tag{6.11}$$

where ζ_X is the indicator valuation representing the optimal policy for X.

Note that Shenoy (2000) also derives cases where the fusion algorithm can be optimized based on the specific valuations involved.

Example 6.3.1. Consider the asymmetric valuation network representation of the used car buyer problem (depicted in Figure 4.15).

When solving this decision problem the fusion algorithm visits the nodes according to the information constraints: S, B, R_2, T_2, R_1, T_1.

Fusion with S*:* Let $\rho_3 = (\rho_1 \otimes \rho_2 \otimes \sigma)^{\downarrow(\{R_1, R_2\})}$. Since S is a chance variable, we apply Equation 6.10:

$$\mathrm{Fus}_S(\mathcal{V}, \Gamma, \Upsilon) = \{\nu_1, \nu_2\} \cup \{\iota_1, \iota_2\} \cup \{\rho_3\} \cup \left\{ \left[\nu_3 \otimes \left(\frac{\rho_1 \otimes \rho_2 \otimes \sigma}{\rho_3} \right) \right]^{\downarrow(\{R_1, R_2, B\})} \right\}.$$

If $\nu_4 = [\nu_3 \otimes ((\rho_1 \otimes \rho_2 \otimes \sigma)/\rho_3)]^{\downarrow(\{R_1,R_2,B\})}$, then the resulting sets of valuations are $\mathcal{V}_1 = \{\nu_1, \nu_2, \nu_4\}$, $\Gamma_1 = \{\rho_3\}$ and $\Upsilon = \{\iota_1, \iota_2\}$.

Fusion with B*:* Note that $\mathrm{dom}(\rho_3) = \{R_1, R_2\}$ and $\mathrm{dom}(\nu_4) = \{R_1, R_2, B\}$, and since B is a decision variable we apply Equation 6.11:[2]

$$\mathrm{Fus}_B(\mathcal{V}_1, \Gamma_1, \Upsilon) = \{\nu_1, \nu_2\} \cup \{\iota_1, \iota_2\} \cup \{\rho_3\} \cup \left\{\nu_4^{\downarrow(\{R_1,R_2\})}\right\}.$$

Denote by ν_5 the utility valuation $\nu_4^{\downarrow(\{R_1,R_2\})}$. The resulting sets of valuations are then $\mathcal{V}_2 = \{\nu_1, \nu_2, \nu_5\}$, $\Gamma_1 = \{\rho_3\}$ and $\Upsilon = \{\iota_1, \iota_2\}$.

Fusion with R_2*:* Since R_2 is a chance variable we again apply Equation 6.10.

$$\begin{aligned}\mathrm{Fus}_{R_2}(\mathcal{V}_2, \Gamma_1, \Upsilon) =& \{\nu_1, \nu_2\} \cup \{\iota_1\} \cup \left\{(\rho_3 \otimes \iota_2)^{\downarrow(\{T_1,R_1,T_2\})}\right\} \\ & \cup \left\{\left[\nu_5 \otimes \left(\frac{(\rho_3 \otimes \iota_2)}{(\rho_3 \otimes \iota_2)^{\downarrow(\{T_1,R_1,T_2\})}}\right)\right]^{\downarrow(\{T_1,R_1,T_2\})}\right\}.\end{aligned} \tag{6.14}$$

Let $\nu_6 = [\nu_5 \otimes ((\rho_3 \otimes \iota_2)/(\rho_3 \otimes \iota_2)^{\downarrow(\{T_1,R_1,T_2\})})]^{\downarrow(\{T_1,R_1,T_2\})}$ and $\rho_4 = (\rho_3 \otimes \iota_2)^{\downarrow(\{T_1,R_1,T_2\})}$. The resulting sets of valuations are then $\mathcal{V}_3 = \{\nu_1, \nu_2, \nu_6\}$, $\Gamma_2 = \{\rho_4\}$ and $\Upsilon_1 = \{\iota_1\}$.

Fusion with T_2*:* Fusion with T_2 proceeds according to the fusion with B, however, in this case the probability valuation ρ_4 contains T_2 in the domain:

$$\mathrm{Fus}_{T_2}(\mathcal{V}_3, \Gamma_2, \Upsilon_1) = \{\nu_1\} \cup \{\iota_1\} \cup \left\{(\nu_2 \otimes \nu_6)^{\downarrow(\{T_1,R_1\})}\right\} \cup \left\{(\rho_4 \otimes \zeta_{T_2})^{\downarrow(\{T_1,R_1\})}\right\},$$

where ζ_{T_2} is an indicator valuation representing the optimal policy for T_2. Let $\nu_7 = (\nu_2 \otimes \nu_6)^{\downarrow(\{T_1,R_1\})}$ and let $\rho_5 = (\rho_4 \otimes \zeta_{T_2})^{\downarrow(\{T_1,R_1\})}$; the resulting sets of valuations are then $\mathcal{V}_4 = \{\nu_1, \nu_7\}$, $\Gamma_3 = \{\rho_5\}$ and $\Upsilon_1 = \{\iota_1\}$.

Fusion with R_1*:* Since R_1 is a chance variable we have:

$$\begin{aligned}\mathrm{Fus}_{R_1}(\mathcal{V}_4, \Gamma_3, \Upsilon_1) =& \{\nu_1\} \cup \left\{(\iota_1 \otimes \rho_5)^{\downarrow(\{T_1\})}\right\} \\ & \cup \left\{\left[\nu_7 \otimes \left(\frac{(\iota_1 \times \rho_5)}{(\iota_1 \otimes \rho_5)^{\downarrow(\{T_1\})}}\right)\right]^{\downarrow(\{T_1\})}\right\}.\end{aligned} \tag{6.16}$$

Note that R_1 is the only chance variable left hence, this expression reduces to $\mathrm{Fus}_{R_1}(\mathcal{V}_4, \Gamma_3, \Upsilon_1) = \{\nu_1\} \cup \left\{(\nu_7 \otimes \iota_1 \otimes \rho_5)^{\downarrow(\{T_1\})}\right\}$. Now, if $\nu_8 = (\nu_7 \otimes \iota_1 \otimes \rho_5)^{\downarrow(\{T_1\})}$, then the resulting valuations are $\mathcal{V}_5 = \{\nu_1, \nu_8\}$, $\Gamma_4 = \emptyset$ and $\Upsilon_2 = \emptyset$.

Fusion with T_1*:* This operation only involves ν_1 and ν_8 hence, we get:

$$\mathrm{Fus}_{T_1}(\mathcal{V}_5, \emptyset, \emptyset) = \left\{(\nu_1 \otimes \nu_8)^{\downarrow(\emptyset)}\right\}. \tag{6.17}$$

[2]Actually, we apply a slightly modified version of Equation 6.11 since ρ_3 does not contain B in the domain.

From the fusion with T_1 the optimal policy for T_1 is found.[3] □

The asymmetric valuation network relies on artificial states to represent asymmetry. From a computational perspective this has the undesirable effect that we in general work with larger valuations during the solution phase, as compared to the asymmetric influence diagram. For instance, fusion with S produces a probability valuation ρ_3 over the effective state space $\Omega_{\{R_1,R_2,S\}}$ which contains 14 configurations. On the other hand, the corresponding operation in the asymmetric influence diagram produces a probability potential with 8 configurations (including the configuration (d_1, d_2, p) having zero probability).

When evaluating an asymmetric influence diagram, the calculations are performed according to its decomposition tree. This implies that the same calculations may be required in different subproblems, i.e., we may perform redundant calculations; obviously, this problem does not occur when evaluating an asymmetric valuation network using the fusion algorithm. One way around this problem is to maintain a cache of previous calculations e.g. a hash table indexed with the calculated potentials. However, the cache may also contain redundant information as not all probability potentials and utility functions need to be cached. Alternatively, we can adopt the method proposed by Cano et al. (2000a). Using this approach we perform two propagations in the decomposition tree: During the first propagation, potentials that are used more than once are marked as *useful* without actually carrying out the calculations. The next propagation then proceeds as usual, but when a probability potential or a utility function marked as useful (by the first propagation) has not been calculated, it is saved into the cache. Note that in Example 6.2.3 we exploit that calculations can be reused.

As illustrated by the evaluation of $I[T_1 \mapsto t_2]$ in Example 6.2.3, the proposed evaluation algorithm may perform redundant divisions (i.e., during the marginalization of R_2). One way to avoid this problem is simply to postpone divisions until they are required for marginalizing out a decision variable (or until all free variables have been marginalized out). Note that this problem may also occur in the fusion algorithm.

Finally, we conjecture that any algorithm for solving influence diagrams can in principle be used to solve an asymmetric influence diagram, given corresponding modifications to Algorithm 6.2.2 and Algorithm 6.2.3.

6.3.2 Comparison with extended influence diagrams

The algorithm for evaluating extended influence diagrams differs slightly from the evaluation of the traditional influence diagram using arc-reversal and node-removal (Shachter 1986). The main difference is that during the evaluation

[3] For a complete description of all the numerical calculations see (Shenoy 2000).

the algorithm exploits the occurrence of e.g. clipping and coalescence in the distribution trees. Hence, computational savings can be achieved, as compared to the proposed evaluation algorithm since the asymmetric influence diagram framework does not support these reductions (except for "clipping"), see also Section 4.4.

It should be noted, though, that during the solution phase we cannot take full advantage of these properties, since information about asymmetry is encoded in different distribution trees. For instance, when eliminating the chance variable A in the reactor problem we first employ arc-reversal w.r.t. the arc from A to R. Eliminating A then produces a new influence diagram together with a new distribution tree for V_2 (see Figure 6.6). In this distribution tree the scenario $(T = t, R = b, B = a)$ is not clipped, since the information about this scenario being impossible is encoded in the distribution tree for B. A detailed description of this problem can also be found in (Bielza and Shenoy 1999).

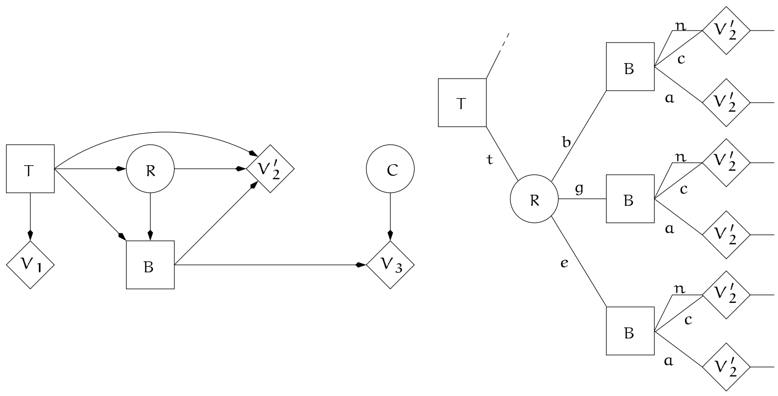

Figure 6.6: The influence diagram and the (partial) distribution tree obtained after eliminating the chance variable A.

In the extended influence diagram framework, the combination of numerical information and information about asymmetry may result in the conditionals being larger than in the asymmetric influence diagram framework. For instance, in the reactor problem the decision variable T occurs in the domain of the conditional for R (see Figure 4.21), which is not the case in the asymmetric influence diagram representation. Thus, during the evaluation of an extended influence diagram we will in general have larger conditionals than in the evaluation of an asymmetric influence diagram. The increased size of the conditionals is also a result of the use of artificial states for representing asymmetry. Note that the asymmetric influence diagram framework may also require artificial states, however, such asymmetry is in general represented using labels (see Example 4.3.1).

Finally, as described in Section 6.3.1, the evaluation of an asymmetric influence diagram may result in the same potential being computed multiple times; similar to the fusion algorithm, this problem does not occur when evaluating an extended influence diagram. However, as also described in Section 6.3.1, this computational inefficiency can be overcome by caching probability potentials and utility functions that are used more than once.

6.4 Discussion

We have presented an algorithm for solving asymmetric decision problems formulated in the framework of asymmetric influence diagrams. The method is based on a decomposition of the original decision problem into a collection of smaller symmetric subproblems organized in a rooted tree structure.

The evaluation proceeds by recursively solving the smaller symmetric subproblems starting from leaves and working towards the root. Thus, the different branches in the decomposition tree can be solved independently of each other; however, this also implies that we may need to perform the same calculations in different branches. One way to overcome this computational problem is to apply the approach suggested by Cano et al. (2000a). That is, we perform two propagations in the decomposition tree: During the first propagation, probability potentials and utility functions that are used more than once are marked as *useful* without actually performing any calculations. In the second propagation, probability potentials and utility functions that have been marked as useful (and have not yet been calculated) are saved into the cache.

The notion of a well-defined asymmetric influence diagram has been defined; however, from an operational point of view it is not quite as easy to determine whether or not an asymmetric influence diagram is well-defined as it is for a partial influence diagram. The main problem is that all the utility functions relevant for a split variable cannot be determined solely by using the d-separation criterion; we also need to investigate the future decision variables that may be influenced by a split variable, and this influence may be the result of the labeling of the variables.

Part III

Analyzing decision problems

Chapter 7

Decomposition of influence diagrams

When solving a decision problem we want to determine an optimal policy for the decision variables of interest. A policy for a decision variable is in principle a function over its past, however, as shown in Chapter 3, some of the past may be irrelevant. In this chapter we present a method to decompose a decision problem into a collection of smaller subproblems s.t. a solution (with no redundant variables) to the original decision problem can be found by solving the subproblems independently. The method is based on an operational characterization of the future variables being relevant for a decision variable, thereby also providing a characterization of those parts of a decision problem that are relevant for a particular decision.

The content of this chapter is based on (Nielsen 2001).

7.1 Introduction

Given an influence diagram representation of a decision problem, we can use the explicit structural information for analyzing relevance. For example, when analyzing the influence diagram depicted in Figure 7.1 we find that knowledge of the states of D_1, A, D_2 and D_3 does not improve decision D_4 as long as the states of B and C are known. Hence, the variables D_1, A, D_2 and D_3 are not required for D_4, and having identified these irrelevant variables the domain of the policy for D_4 is reduced considerably; see Chapter 3 for different operational characterizations of the required past for a decision variable.

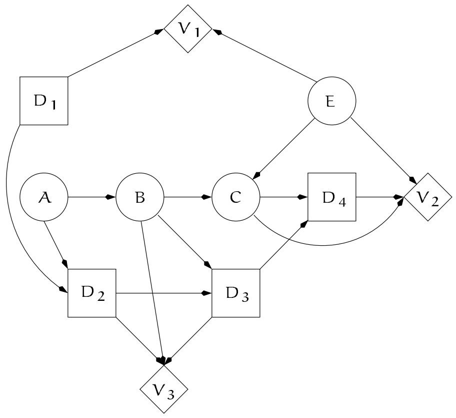

Figure 7.1: An influence diagram with four chance nodes A, B, C and E, four decision nodes D_1, D_2, D_3, D_4 and three value nodes.

Analogously to the possible occurrence of redundant variables in the past of a decision, there may also exist future variables which are irrelevant. By also identifying these variables, we obtain a complete specification of those parts of a decision problem that are relevant for a particular decision.

The advantages of being able to characterize the relevant parts of a decision problem are twofold. First of all, if a decision variable is of particular importance, then we can focus our attention on the parts relevant for that decision variable when specifying the quantitative part (probabilities and utilities) of the model. Similarly, when using the influence diagram as a tool for communication we can answer questions as to whether a variable X provides information which is relevant for a particular decision D. Being able to answer such questions is also useful in the context of *information filtering*, i.e., the problem of adjusting the configuration and quantity of information displayed to a decision maker, see e.g. (Horwitz and Barry 1995).

The second advantage has to do with the evaluation. When referring to the evaluation of an influence diagram we usually mean computing an optimal policy for all decisions involved. This interpretation can, however, be slightly misleading as we might only be interested in a subset of the decisions; actually, in most cases we are only interested in the initial decision as this decision can be seen as generating a new decision problem. When only a subset of the decisions is of interest, then the influence diagram may contain superfluous information, i.e., variables which have no impact on the solution for the decisions in question. By removing such irrelevant variables we can reduce the size of the problem and, thereby, the computational complexity of solving the decision problem.

In this chapter we propose a method to decompose an influence diagram into a collection of smaller influence diagrams. The decomposition produces an influence diagram for each decision variable of interest, and each of these influence diagrams contains exactly the variables necessary and sufficient for determining an optimal policy for the associated decision variable; hence, the influence diagrams can be solved independently of each other. The decomposition does not place any restrictions on the subsequent evaluation, so we can use any evaluation scheme that we find suitable, see e.g. (Shachter 1986; Shenoy 1992; Jensen, Jensen, and Dittmer 1994; Madsen and Jensen 1999). Additionally, since the influence diagrams only include variables that are relevant for the associated decision variables, we are always given policies with no redundant variables. Note, that the methods cited above do not ensure this property themselves, i.e., solving an ordinary influence diagram is not guaranteed to produce optimal policies with no redundant variables. Note also that even though the proposed method is given in the context of influence diagrams it can be applied to partial influence diagrams as well.

7.2 Decomposition of influence diagrams

In this section we describe a method for decomposing an influence diagram, I, into a collection of smaller influence diagrams s.t. an optimal strategy for I can be found by solving the smaller influence diagrams independently. More precisely, the decomposition produces an influence diagram for each decision variable, D, in I s.t. an optimal policy for D can be found by solving the associated influence diagram independently of the other influence diagrams produced by the decomposition.

Before describing the decomposition method, it is important to notice that when calculating an optimal strategy for an influence diagram I, the variables in $\mathcal{C}_0 = \text{pred}(D_1)$ are never marginalized out. This allows us to define a *partial realization* $\mathcal{R}$ of I as a realization which does not necessarily include the potential λ_X if there does not exist a variable $Y \in \text{dom}(\lambda_X)$ s.t. $\mathcal{C}_0 \prec Y$; we use λ_X when referring to the probability potential or the utility function associated with X, if it is of no importance whether X is a chance node or a value node. We allow a (partial) realization $\mathcal{R}$ to be extended with a set $\mathcal{R}'$ of potentials if there does not exist a potential in $\mathcal{R}'$ with the same domain as a potential in $\mathcal{R}$. That is, the extension is admissible if no variable is associated with more than one potential w.r.t. $\mathcal{R} \cup \mathcal{R}'$.

Decomposing an influence diagram I is essentially a question of identifying the probability potentials and the utility functions involved in the computation of the optimal policy δ_{D_i}, for all $1 \leq i \leq n$ and for any realization of I (see Equation 3.11). These probability potentials and utility functions are uniquely identified by their associated variables. Thus, apart from the required variables we also need to identify the future variables that are *relevant* for the decisions in the influence diagram, i.e., the variables that may influence the optimal policy for a decision variable:

Definition 7.2.1. Let I be an influence diagram and let D be a decision variable in I. The variable X ($D \prec X$) is said to be *relevant* for D if either:

- $X \in \mathcal{U}_C \cup \mathcal{U}_V$ and there exists two realizations $\mathcal{R}_1$ and $\mathcal{R}_2$ of I that only differ on the probability potential or the utility function associated with X s.t. the optimal policies for D are different in $\mathcal{R}_1$ and $\mathcal{R}_2$, or

- $X \in \mathcal{U}_D$ and there exists a realization of I and two different policies δ_X^1 and δ_X^2 for X s.t. the optimal policies for D are different w.r.t. δ_X^1 and δ_X^2.

The set of variables being relevant for D is denoted rel(D).

The variables $\mathrm{req}(D) \cup \{D\} \cup \mathrm{rel}(D)$ determine the probability potentials and the utility functions involved in the computation of δ_D. Thus, we seek a decomposition scheme which produces a collection of influence diagrams s.t. each influence diagram contains exactly the relevant and required variables for the associated decision and, furthermore, obeys the information constraints specified by the original influence diagram. Note that the latter property may require the inclusion of informational arcs which are not part of the original influence diagram.

Definition 7.2.2. Let I be an influence diagram with nodes $\mathcal{U}^I$ and arcs $\mathcal{E}^I$, and let D be a decision variable in I. The influence diagram I_D is said to *reflect the decision problem* D in I if:

- $\mathcal{U}^{I_D} = \mathrm{rel}(D) \cup \mathrm{req}(D) \cup \{D\}$, and

- $\mathcal{E}^{I_D} = \{(X,Y) | X, Y \in \mathcal{U}^{I_D} \text{ and } (X,Y) \in \mathcal{E}^I\} \cup \{(X,Y) | X \in \mathcal{U}^{I_D}, Y \in \mathcal{U}_D^{I_D}, \text{ where } X \prec^I Y \text{ and } X \not\prec^{I_D} Y\}$.

Definition 3.2.2 and Definition 7.2.1 give semantic characterizations of the required variables and the relevant variables for a particular decision variable, respectively. Different syntactical characterizations of the required variables are described in Chapter 3, and in this chapter we shall adopt Algorithm 3.3.1 (Shachter 1999). Recall that this algorithm is based on the chance variable representation of a policy and, as shown in Chapter 3, the value nodes being relevant for a decision variable D_i are exactly the value nodes $\mathcal{V}_i$ found by that algorithm.

Example 7.2.1. Consider the influence diagram depicted in Figure 7.1. By applying Algorithm 3.3.1 to this influence diagram we start off by visiting decision D_4. As $\mathrm{pred}(D_4) = \{D_1, A, D_2, B, D_3, C\}$ we find that only B and C are required for D_4; all other nodes $X \in \mathrm{pred}(D_4)$ are d-separated from $\mathcal{V}_4 = \{V_2\}$ given $\mathrm{pred}(D_4)\setminus\{X\}$. Note that from Algorithm 3.3.1 we can also infer that V_2 is the only value node relevant for D_4. Decision D_4 is then replaced by its chance variable representation D_4' having B and C as parents (see Figure 7.2a), and the algorithm proceeds to decision D_3.

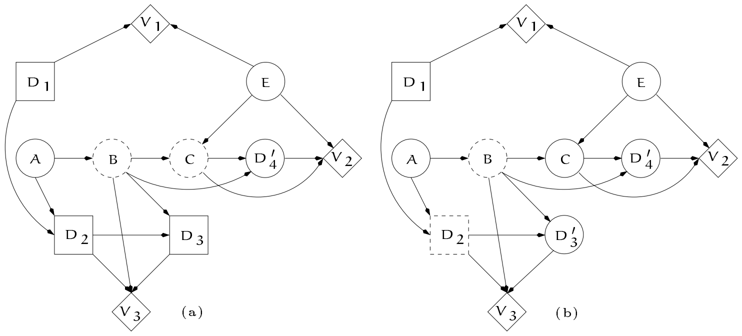

Figure 7.2: The figures depict the influence diagrams obtained from the influence diagram in Figure 7.1, by applying the first two iterations of Algorithm 3.3.1. The dashed nodes in Figure (a) and (b) are the nodes required for D_4 and D_3, respectively.

Since $\mathcal{V}_3 = \{V_3\}$ in the influence diagram produced by the previous step (see Figure 7.2a), we find that D_2 and B are the only variables required for D_3. Decision D_3 is then replaced by its chance variable representation (see Figure 7.2b), and the algorithm proceeds as above until all decision variables have been investigated. □

So, from Algorithm 3.3.1 we have a syntactical characterization of the variables required for a given decision variable D as well as the set of value nodes relevant for D. Moreover, from Definition 7.2.1 it is apparent that a decision variable D' ($D \prec D'$) is relevant for D if and only if there exist a utility function ψ relevant for both D and D'. Thus, we have narrowed our search for relevant variables down to the chance variables succeeding D under $\prec$.

Theorem 7.2.1. Let I be an influence diagram and let D_i be a decision variable in I. Let I_i denote the influence diagram obtained from I by replacing D_j with its chance variable representation, for $j = i+1, \ldots, n$. Then the chance variable X ($D_i \prec X$) is relevant for D_i if and only if

- X is not barren in the influence diagram formed from I_i by removing all value nodes that are not relevant for D_i, and

- there exists a utility potential ψ_V relevant for D_i s.t. X is d-connected to V in I_i given $\{D_i\} \cup \text{pred}(D_i)$.

A proof of Theorem 7.2.1 is given in Section 7.3.

Having found a syntactical characterization of the variables being required and relevant for a decision variable D, we have the following theorem which shows that performing a decomposition according to Definition 7.2.2 is sound w.r.t. the optimal policies.

Theorem 7.2.2. Let I be an influence diagram and let $\mathcal{R}$ be a realization of I. If I_D reflects the decision problem D in I, then $\mathcal{R}$ is a (partial) realization of I_D and $\delta_D^{I_D} = \delta_D^I$.

Proof. From the construction of I_D we only need to show that for any realization of I, if λ_X is needed for the computation of δ_D^I, then $\{X\} \cup pa(X) \subseteq \mathcal{U}^{I_D}$. That is, λ_X is contained in the (partial) realization of I_D. Now, consider $X \succ D$ and assume that λ_X is a probability potential. Since λ_X is relevant for the computation of δ_D we have that $X \in \mathcal{U}^{I_D}$ (see Theorem 7.2.1). Moreover, from the properties of d-separation it follows that for any $Y \in pa(X)$, $Y \in \text{req}(D)$ if $Y \in \text{pred}(D)$ and $Y \in \text{rel}(D)$ if $Y \notin \text{pred}(D)$. On the other hand, if X is required for D, then there exists a variable $Y \in pa(X)$ which is relevant for D; if this was not the case, λ_X would not be relevant for the calculation of δ_D. Thus, the variables in $pa(X)$ are either required or relevant for D.

Finally, if X is a value node the proof follows immediately. $\square$

The following two examples illustrate the proposed decomposition method.

Example 7.2.2. Consider the influence diagram depicted in Figure 7.1. From Algorithm 3.3.1 we find that V_2 is the only value node relevant for D_4, and by Theorem 7.2.1 this implies that E is relevant for D_4. By Algorithm 3.3.1 we also have that B and C are required for D_4. These nodes define the probability potentials and the utility function which are needed to compute an optimal policy for D_4, and by Definition 7.2.2 they constitute the nodes in the influence diagram that reflects the decision problem D_4 (see Figure 7.3(D_4)). Note that i) the informational arc from B to D_4 ensures that the information constraints are consistent with the original influence diagram, and ii) B is not associated with a probability potential as it is not marginalized out when calculating an optimal policy for D_4 (we work with a partial realization).

Continuing to decision D_3, we start off by replacing D_4 with a chance variable D_4' having B and C as parents. From Algorithm 3.3.1 we have that B and D_2 are required for D_3 since V_3 is relevant for D_3. These nodes then constitute

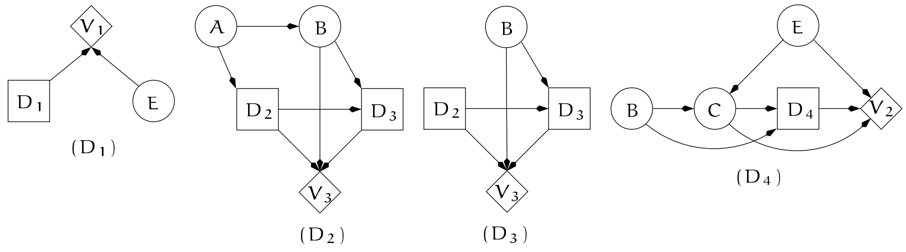

Figure 7.3: The figure shows a decomposition of the influence diagram depicted in Figure 7.1 into four smaller influence diagrams.

the nodes in the influence diagram that reflects the decision problem D_3 (see Figure 7.3(D_3)); by Theorem 7.2.1 no future variables are relevant for D_3. The algorithm proceeds as above until all decision nodes have been considered, see Figure 7.3(D_2) and Figure 7.3(D_1). It should be noted though, that when investigating D_1 the chance node C is d-connected to V_1 which is relevant for D_1. However, C is not relevant for D_1 since it is barren in the influence diagram formed by removing the value nodes V_2 and V_3 which are not relevant for D_1. □

Example 7.2.3. Figure 7.4 shows the influence diagram obtained from the influence diagram in Figure 3.3 by removing the arc (E, G); we assume that the temporal ordering of the decision nodes corresponds to the indicies of these nodes.

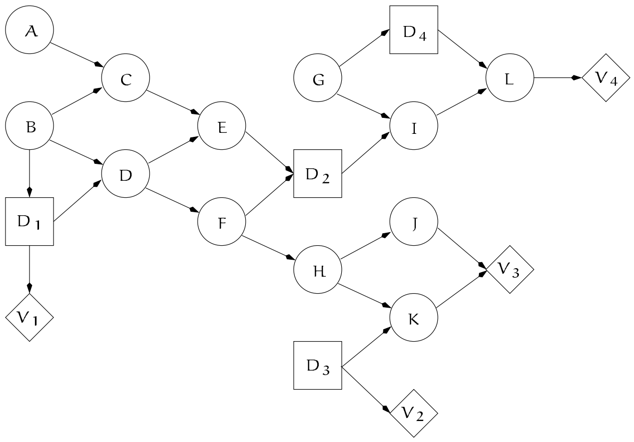

Figure 7.4: The influence diagram obtained from the influence diagram in Figure 3.3 by removing the arc (E, G).

When decomposing this influence diagram we visit the nodes in reverse temporal order. Hence, starting with D_4 we find that D_2 and G are required for D_4, and that I, L and V_4 are relevant for D_4, i.e., $\text{req}(D_4) = \{D_2, G\}$ and $\text{rel}(D_4) = \{I, L, V_4\}$. Continuing to decision D_3 we have that $\text{req}(D_3) = \{F\}$ and $\text{rel}(D_3) = \{H, J, K, V_2, V_3\}$. Finally for decision D_2 and D_1 we get:

- $\text{req}(D_2) = \emptyset$ and $\text{rel}(D_2) = \{G, D_4, I, L, V_4\}$.
- $\text{req}(D_1) = \{B\}$ and $\text{rel}(D_1) = \{D, F, D_3, H, J, K, V_1, V_2, V_3\}$.

When decomposing the influence diagram according to this analysis we obtain the influence diagrams depicted in Figure 7.5. Note that the influence diagrams representing the decision problems for D_2 and D_4 are identical. Note also that the influence diagrams reflecting the decision problems for D_1 and D_3 include an additional informational arc to ensures that the information constraints are consistent with the original influence diagram. Moreover, in the influence diagram reflecting the decision problem D_3, a probability potential is not assigned to the chance variable F, i.e., we have a partial realization.

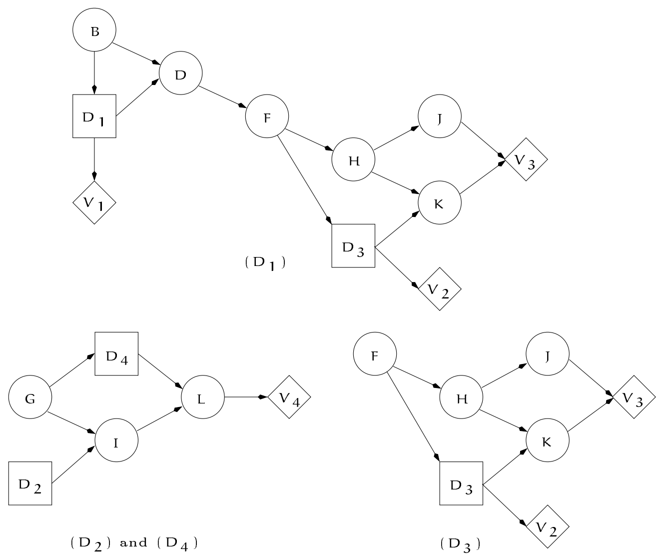

Figure 7.5: The figure illustrates a decomposition of the influence diagram in Figure 7.4. Note that the influence diagrams representing the decision problems for D_2 and D_4 are identical.

□

In addition to producing policies with no redundant variables, the decomposition of an influence diagram may also yield a reduction in the computational complexity when calculating an optimal strategy. For instance, the strong junction tree representation of the original influence diagram depicted in Figure 7.1 contains a clique consisting of five variables. On the other hand, the largest clique in the junction trees, for the influence diagrams produced by the decomposition, contains only four variables (see Figure 7.6).

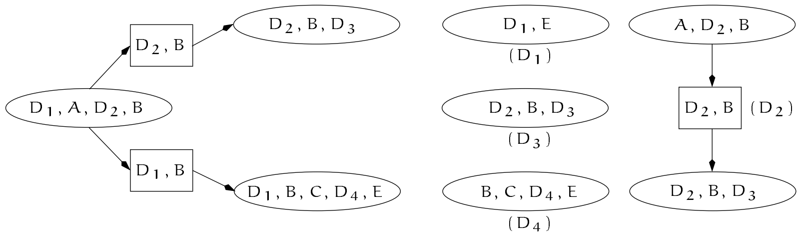

Figure 7.6: The left part of the figure shows that the strong junction tree for the influence diagram depicted in Figure 7.1. The right part of the figure shows the strong junction trees for the influence diagrams produced by decomposing the influence diagram in Figure 7.1.

It is easy to verify that the computational complexity of solving one subproblem is not larger than the computational complexity of solving the original decision problem. However, when working with overlapping subproblems we may need to repeat calculations previously performed. An immediate approach to overcome this problem is to cache/reuse some of the previous computations.

7.3 Proofs

Before giving the proof of Theorem 7.2.1 we need the following lemma.

Lemma 7.3.1. Let X and Y be chance variables in a Bayesian network BN, and let $\bar{c}$ be a configuration over a set of variables $\mathcal{Z}$ not containing X or Y. If X is not barren, then X is d-separated from Y given $\bar{c}$ if and only if for any two realizations of BN differing only on the potential $P(X|pa(X))$ it holds that $P_1(Y|\bar{c}) = P_2(Y|\bar{c})$.

Proof. Consider the Bayesian network BN' obtained from BN by adding an artificial binary chance variable W as a parent for X. Let P_1 and P_2 be any two realizations differing only on the potential $P(X|pa(X))$ in BN, and let the conditional probability distribution for X in BN' be specified s.t. $P'(X|pa(X), W = 0) = P_1(X|pa(X))$ and $P'(X|pa(X), W = 1) = P_2(X|pa(X))$.

Recall that W and Y are d-separated given $\bar{c}$ if and only if for any realization of BN' the state of W has no impact on $P'(Y|\bar{c})$, i.e., $P'(Y|\bar{c}, W = 0) = P'(Y|\bar{c}, W = 1)$. Thus, W is d-separated from Y given $\bar{c}$ if and only if for any two realizations P_1 and P_2 of BN, differing only on $P(X|pa(X))$, it holds that $P_1(Y|\bar{c}) = P_2(Y|\bar{c})$ in BN. However, from the construction of BN' we have that as X is not barren, W is d-separated from Y given $\bar{c}$ if and only if X is d-separated from Y given $\bar{c}$.

□

Based on the lemma above, we present the proof of Theorem 7.2.1.

Proof (Theorem 7.2.1) It is easy to verify that replacing D_j with its chance variable policy, for $j = i+1, \ldots, n$, does not change the optimal policy for D_i nor does it introduce any "false" structural dependencies (see also the proof of Theorem 3.3.3).

So we can restrict our attention to the last decision variable in I. From Equation 3.11 we have:

$$\delta_{D_n}(\text{pred}(D_n)) = \arg\max_{D_n} \sum_{\mathcal{C}_n} P(\mathcal{C}_n|\mathcal{C}_0, \ldots, \mathcal{C}_{n-1}, D_1, \ldots, D_n) \sum_{V \in \mathcal{U}_V} \psi_V \; .$$

Without loss of generality we can assume that ψ_V is the only utility function relevant for D_n.

$$\begin{aligned}
\delta_{D_n}(\text{pred}(D_n)) &= \arg\max_{D_n} \sum_{\mathcal{C}_n} P(\mathcal{C}_n|\text{pred}(D_n), D_n)\psi_V(pa(V)) \\
&= \arg\max_{D_n} \sum_{\mathcal{C}_n \cap pa(V)} \psi_V(pa(V)) \sum_{\mathcal{C}_n \setminus pa(V)} P(\mathcal{C}_n|\text{pred}(D_n), D_n) \\
&= \arg\max_{D_n} \sum_{\mathcal{C}_n \cap pa(V)} \psi_V(pa(V))P(\mathcal{C}_n \cap pa(V)|\text{pred}(D_n), D_n) \; .
\end{aligned} \tag{7.2}$$

So, X is relevant for D_n if and only if $P(X|pa(X))$ has an effect on $P(\mathcal{C}_n \cap pa(V)|\text{pred}(D_n), D_n)$. By Lemma 7.3.1 we have that this holds if and only if X is d-connected to $\mathcal{C}_n \cap pa(V)$ given $\text{pred}(D_n) \cup \{D_n\}$. X is d-connected to $\mathcal{C}_n \cap pa(V)$ given $\text{pred}(D_n) \cup \{D_n\}$ if and only if X is d-connected to V given $\text{pred}(D_n) \cup \{D_n\}$, thereby completing the proof. □

7.4 Discussion

We have presented an operational characterization of the future variables being relevant for a decision variable. These variables, together with the variables of the required past, describe the parts of a decision problem which are necessary and sufficient to consider when calculating an optimal policy for a particular decision variable.

Moreover, based on this characterization we have presented a method for decomposing an influence diagram into a collection of smaller influence diagrams. A solution to the original influence diagram can then be found by solving these smaller influence diagrams independently. The decomposition ensures that no redundant variables are contained in the optimal strategy and, furthermore, it may also reduce the computational complexity of solving the original decision problem.

The proposed method may also provide a way to simplify the elicitation of the structure of a complex decision problem. For instance, it may be possible to characterize conditions that are necessary and sufficient to ensure the consistency of different subproblems based on the existence of a decision problem that decomposes into these subproblems.

Chapter 8

Sensitivity analysis in influence diagrams

One of the main difficulties when representing decision problems using (partial) influence diagrams is to elicit the utilities and the probabilities. This makes it desirable to be able to investigate: i) how sensitive the solution is to variations in some utility or probability parameter, and ii) how robust the solution is to joint variations over a set of parameters. In this chapter we propose two methods for performing these types of analysis.

The content of this chapter is based on (Nielsen and Jensen 2001a).

8.1 Introduction

When solving an influence diagram the aim is to determine an optimal policy for each decision involved. These policies (and the maximum expected utility of the influence diagram) are sensitive to variations in both the utilities and the probabilities, but, unfortunately, it is usually the case that these values are difficult to elicit when subject to second order uncertainty. This makes it desirable to be able to determine how robust the solution is to variations in both the utilities and the probabilities, see also (Poh and Horvitz 1993). Based on such an analysis, the modeler may focus his/her attention on parameter values for which the solution is particularly sensitive (a situation known as *one-way* sensitivity analysis). Another important type of analysis concerns the joint variation of a set of parameters (n-*way* sensitivity analysis). This type of analysis yields little information about the individual parameters, but provides general insight into the overall robustness of the model.

Methods for performing both one-way and n-way sensitivity analysis have previously been described in the medical decision making literature, see e.g. (Felli and Hazen 1999) for an overview. Most of these methods are approximate and require that the decision maker assigns a probability distribution to each of the uncertain parameters. However, when a decision problem contains many uncertain parameters the task of assigning probability distributions to these parameters becomes cumbersome and time-consuming. This problem especially appears in relation to uncertain utility parameters. First of all, the utility parameters are not independent when the decision maker can specify a preference ordering over the outcomes; thus, we need conditional probability distributions over the parameters. Secondly, the probability distributions may be difficult to elicit when the utility values have no straightforward interpretation; this is the case in e.g. (Skaanning 2000).

One-way and n-way sensitivity analysis have also been addressed within the framework of Bayesian networks. For instance, Laskey (1995) provides a first-order approximation of the effect when varying a parameter value w.r.t. a posterior marginal. Alternatively, (Castillo et al. 1997; Kjærulff and van der Gaag 2000; Coupé 2000) describe exact methods for performing one-way and n-way sensitivity analysis in a Bayesian network. These methods, however, cannot immediately be applied in the context of influence diagrams, as the optimal policy for each decision variable can be seen as the posterior marginal in question for Bayesian networks. That is, for each decision variable we would in principle need to consider all scenarios leading to that decision variable, a frequently impossible task!

In this chapter we propose exact methods for performing one-way and n-way sensitivity analysis in an influence diagram. The methods mainly address the situation with uncertain utility parameters, and does not require a probability distribution over the parameters. The methods are also adopted for one-way sensitivity analysis for probability parameters, and we discuss their application to n-way sensitivity analysis w.r.t. probability parameters. An explicit representation of the parameters forms the basis for the algorithms, and the calculations are performed by propagation in the strong junction tree representation of the influence diagram. It is worth noticing that the results presented in this chapter can also be adopted to the framework of valuation based systems (Shenoy 1992).

8.2 Sensitivity analysis

One of the initial steps towards a coherent approach for sensitivity analysis in influence diagrams was proposed by Bielza et al. (1996), who considers the situation where some of the utilities and probabilities have only been partially assessed. The method assumes that these uncertain values are represented by parameters, and a set of non-dominated strategies is computed based on this parametric model. Unfortunately, the set of strategies grows intractably large when dealing with sequential decision problems.

In what follows we make a distinction between *value sensitivity* and *decision sensitivity* when referring to sensitivity analysis. Value sensitivity concerns variations in the maximum expected utility when changing a set of parameters, and decision sensitivity refers to changes in the optimal strategy.

Analysis of decision sensitivity can furthermore be characterized as either *threshold proximity* or *probabilistic sensitivity analysis*. Threshold proximity uses a distance measure to determine the admissible values a parameter (or set of parameters) can be assigned without changing the optimal strategy found w.r.t. the initial values of the parameters. Probabilistic sensitivity analysis assigns a probability distribution to each parameter under investigation. Hence, the analysis is based on the probability of obtaining a different optimal strategy rather than determining the admissible domain in which the parameters can be varied, see e.g. (Doubilet et al. 1985).

A method based on value sensitivity has been proposed in (Felli and Hazen 1998). This method uses the expected value of perfect information as a sensitivity indicator, and similar to probabilistic sensitivity analysis it requires that a probability distribution is assigned to each parameter under investigation. More precisely, let $\bar{\mathsf{t}}$ be the uncertain parameters, and let Δ^0 be the optimal strategy found w.r.t. the initial values of the parameters. If ρ_Δ and ρ_{Δ^0} denote the expected utilities of the influence diagram under the strategies Δ and Δ^0, respectively, then the expected value of perfect information (denoted by $\mathsf{EVPI}(\bar{\mathsf{t}})$) is given by:

$$\mathsf{EVPI}(\bar{\mathsf{t}}) = \mathsf{E}_{\bar{\mathsf{t}}}[\max_{\Delta}[\rho_\Delta(\bar{\mathsf{t}})] - \rho_{\Delta^0}(\bar{\mathsf{t}})].$$

This expectation can be very difficult to evaluate hence, simulations are usually applied. A detailed description and comparison of the methods outlined above, can be found in (Felli and Hazen 1998; Felli and Hazen 1999).

One drawback of employing sensitivity analysis based upon probability distributions over the uncertain parameters, concerns the elicitation of these distributions. Given that the decision maker has a (partial) preference ordering over the outcomes, the uncertain utility parameters are not independent, i.e., we work with conditional probability distributions over the utility parameters.[1] This makes it difficult to elicit these distributions, in particular, when the utilities have no straightforward interpretation; in (Skaanning 2000), for example, the utilities are defined based on a combination of cost, insult and risk. Parameter dependence also has an impact on the computational complexity. Even when the parameters are assumed to be independent, the calcu-

[1] Analogously to the standard assumptions within the learning community (see e.g. (Cooper and Herskovits 1991)), it is usually assumed that the probability parameters are independent.

lations can be rather cumbersome, and Monte Carlo methods are usually applied to sample values for the parameters. For instance, when applying the method based on the expected value of perfect information we need to compute $\text{diff}_i = [\max_\Delta[\rho_\Delta(\bar{t}^i)] - \rho_{\Delta^0}(\bar{t}^i)]$, for each generated sample $\bar{t}^i$; the expectation is then approximated by $\mathsf{EVPI}(\bar{t}) \approx \frac{1}{N}\sum_{i=1}^{N} \text{diff}_i$, where N is the number of generated samples, see e.g. (Bielza et al. 2000). However, when the parameters are not independent we will in general require a larger set of samples in order to get a representative sample from the joint probability distribution over all the parameters. Thus, dependent parameters may induce a serious computational overhead since the decision problem is evaluated for each generated sample.

Finally, it should be noted that decision sensitivity has recently been subject to criticism on the ground that it tends to overestimate the sensitivity of a model, see e.g. (Felli and Hazen 1999). On the other hand, one may argue that if a parameter induces significant changes in the expected utility without influencing the optimal strategy, then the model is also insensitive to the parameter. To overcome the problem of overestimating model sensitivity, one may perform the decision analysis iteratively. That is, after decision sensitivity has been analyzed we can calculate the expected utility of the optimal strategy induced by changing the parameters; if the expected utility is not significantly different from the expected utility of the initial strategy, then the analysis is repeated and the strategy is allowed to change accordingly. Thus, by analyzing decision sensitivity iteratively, we can avoid the problem of overestimating the sensitivity of the model.

Based on the discussions above we shall focus on threshold proximity hence, when making subsequent reference to sensitivity analysis in an influence diagram, we consider the effect on the optimal strategy when varying a set of parameters associated with a collection of parameterized utility functions and probability potentials. That is, how much can the parameters be varied without changing the optimal policies found w.r.t. the initial values of the parameters (decision sensitivity).

8.3 Basic properties of sensitivity analysis in influence diagrams

In this section we consider an explicit representation of both utility and probability parameters. Based on this representation we identify the structure of the parameterized expected utility functions for the decision variables; these functions form the basis for the subsequent sensitivity analysis. It should be noted that the explicit representation of the parameters is mainly used for ease of exposition, and computational complexity is therefore not considered in this section.

8.3.1 Parameterized probability potentials and utility functions

Let $\psi_V(pa(V))$ be a parameterized utility function given by:

$$\psi_V(pa(V) = \pi')(t) = \begin{cases} t & \text{if } \pi' = \pi \\ x_{\pi'} & \text{otherwise,} \end{cases}$$

where π and π' are configurations over $pa(V)$ and $x_{\pi'} = \psi_V(pa(V) = \pi')$; without loss of generality we assume that ψ_V is only parameterized for $pa(V) = \pi$.

The utility parameter t can be modeled explicitly by introducing a new parent T for V having states 0 and 1, and with prior distribution $(t^0, 1 - t^0)$ (see Figure 8.1a). The new utility function $\psi^*_V(pa(V), T)$ is then constructed s.t.:

$$\psi^*_V(pa(V) = \pi', T) = \psi_V(pa(V) = \pi'), \forall \pi' \neq \pi$$

and

$$\begin{aligned} \psi^*_V(pa(V) = \pi, T = 0) &= t^{\min} \\ \psi^*_V(pa(V) = \pi, T = 1) &= t^{\max}, \end{aligned}$$

where $[t^{\min}; t^{\max}]$ is the domain of t. The prior distribution $(t^0, 1 - t^0)$ of T is specified such that $t^0 \cdot t^{\min} + (1 - t^0) \cdot t^{\max}$ is the initial value for $\psi_V(pa(V) = \pi)$. In what follows, any variable T that models a parameter t will be referred to as a *parameter variable*.

Now, let $P(A|pa(A))$ be a conditional probability potential and let $P(A = a_j|pa(A) = \pi) = x_j$, for all $1 \leq j \leq |sp(A)|$. If $P(A|pa(A))$ is parameterized for $A = a_i$ given $pa(A) = \pi$, then:

$$P(A = a_j|pa(A) = \pi)(t) = \begin{cases} t & \text{if } j = i \\ x_j \cdot \left(\frac{1-t}{1-x_i}\right) & \text{otherwise.} \end{cases} \tag{8.4}$$

Thus, by changing the parameter t the residual probability mass is distributed over the states $sp(A)\backslash\{a_i\}$, i.e., we employ proportional scaling. To avoid the potential problem of zero divisions, uncertain probability parameters are assumed to only take on values in the domain $(0; 1)$, i.e, they are not allowed to attain extreme values. This also conforms with the intuition that extreme probabilities (i.e., 0 or 1) are not subject to second order uncertainty, and is therefore not in the focus of sensitivity analysis; if the second order uncertainty is described by a probability distribution over the interval $[0; 1]$, then the variance is 0 if the mean is either 0 or 1.

Similar to utility parameters, a probability parameter can be modeled explicitly by introducing a new parent T for A (see Figure 8.1b) s.t.:

$$P^*(A|pa(A) = \pi', T) = P(A|pa(A) = \pi'), \forall \pi' \neq \pi \tag{8.5}$$

and

$$\begin{aligned} P^*(A|pa(A) = \pi, T = 0) &= P(A|pa(A) = \pi)(0); \\ P^*(A|pa(A) = \pi, T = 1) &= P(A|pa(A) = \pi)(1). \end{aligned}$$

The prior distribution $(1 - t^0, t^0)$ over T is specified s.t. $t^0 = P(A = a_i|pa(A) = \pi)$.

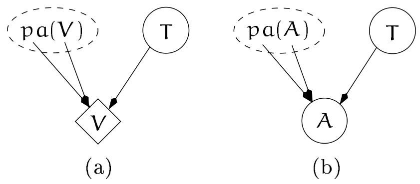

Figure 8.1: Explicit modeling of utility and probability parameters, respectively.

Note that, as argued by Wang and Druzdzel (2000), it may be desirable only to change some of the probabilities and keeping the other probabilities fixed. This situation can easily be accommodated by modifying Equation 8.4 s.t. the residual probability mass is distributed over the parameters which are allowed to vary: If $P(A|pa(A))$ is parameterized for $A = a_i$ given $pa(A) = \pi$ and $\mathcal{K} \subset \{1, \ldots, |sp(A)|\} \setminus \{i\}$ is the set of probabilities we want to keep fixed, then:

$$P(A = a_j|pa(A) = \pi)(t) = \begin{cases} t & \text{if } j = i \\ x_j & \text{if } j \in \mathcal{K} \\ x_j \cdot \left(\frac{1 - t - \sum_{k \in \mathcal{K}} x_k}{1 - x_i - \sum_{k \in \mathcal{K}} x_k} \right) & \text{if } j \notin \mathcal{K}. \end{cases}$$

To model this type of parameterized probability potential we need to define $P^*(A|pa(A), T)$ slightly different for the configuration $pa(A) = \pi$:

$$\begin{aligned} P^*(A|pa(A) = \pi, T = 0) &= P(A|pa(A) = \pi)(0); \\ P^*(A|pa(A) = \pi, T = 1) &= P(A|pa(A) = \pi)(1 - \sum_{k \in \mathcal{K}} x_k). \end{aligned}$$

Accordingly, T is assigned the prior distribution:

$$P(T) = (1 - \frac{t^0}{1 - \sum_{k \in \mathcal{K}} x_k}, \frac{t^0}{1 - \sum_{k \in \mathcal{K}} x_k}),$$

where t_0 can vary within the interval $(0; 1 - \sum_{k \in \mathcal{K}} x_k)$.

So, by varying the prior distribution over the parameter variables we can investigate the effect of changing the parameters. Moreover, by marginalizing out the parameter variables it is easy to see that an explicit modeling of the parameters does not change the optimal strategy nor the maximum expected utility.

8.3.2 Parameterized expected utility functions

In what follows we describe the general structure of the parameterized expected utility functions $\rho(D_i, \mathrm{pred}(D_i))(t)$, for all $1 \leq i \leq n$ and for some parameter t.

Consider a utility function ψ_V being parameterized w.r.t. the configuration $pa(V) = \pi$; for now we only consider the last decision variable D_n in the influence diagram I, and without loss of generality we assume that ψ_V is the only utility function in I. The parameterized expected utility function for decision D_n is then given by:

$$\begin{aligned}
\rho(D_n, \mathrm{pred}(D_n))(t) &= \sum_{\mathcal{C}_n, T} P(\mathcal{C}_n, T | D_n, \mathrm{pred}(D_n)) \psi_V^*(T, pa(V)) \\
&= \sum_{T} P(T) \sum_{\mathcal{C}_n} P(\mathcal{C}_n | D_n, \mathrm{pred}(D_n))) \psi_V^*(T, pa(V)) \\
&= t_0 \sum_{\mathcal{C}_n} P(\mathcal{C}_n | D_n, \mathrm{pred}(D_n)) \psi_V^*(T = 0, pa(V)) \qquad (8.8) \\
&\quad + (1 - t_0) \sum_{\mathcal{C}_n} P(\mathcal{C}_n | D_n, \mathrm{pred}(D_n)) \psi_V^*(T = 1, pa(V)).
\end{aligned}$$

That is, the parameterized expected utility function is linear w.r.t. t. Thus, we can locally vary the prior distribution of T (corresponding to the parameter t) and observe the effect on $\rho(D_n, \mathrm{pred}(D_n))$. It is easy to see that having multiple utility parameters $\bar{t}$ also yields $\rho(D_n, \mathrm{pred}(D_n))(\bar{t})$ being linear in $\bar{t}$.

When having a parameterized conditional probability distribution $P(A|pa(A))(t)$, the expected utility function is given by:

$$\begin{aligned}
\rho(D_n, \mathrm{pred}(D_n))(t) &= \sum_{\mathcal{C}_n, T} P(\mathcal{C}_n, T | D_n, \mathrm{pred}(D_n)) \psi_V(pa(V)) \\
&= \frac{\sum_T P(T) \sum_{\mathcal{C}_n} \prod_{\phi \in \Phi_{C_n}} \phi \psi_V(pa(V))}{\sum_T P(T) \sum_{\mathcal{C}_n} \prod_{\phi \in \Phi_{C_n}} \phi}, \qquad (8.9)
\end{aligned}$$

where Φ_{C_n} is the set of conditional probability potentials having a variable $X \in \mathcal{C}_n$ in the domain.

By Equation 8.9 it follows that if neither A nor any of its descendants are contained in $\mathrm{pred}(D_n)$, then by iteratively marginalizing out the last chance variable descendant of A (in the denominator) we obtain a collection of unity potentials. Thus, the expression in the denominator reduces to a constant function over t, and $\rho(D_n, \mathrm{pred}(D_n))(t)$ is therefore linear in t; otherwise, it is a quotient of two linear functions (note that this is consistent with the results found in e.g. (Kjærulff and van der Gaag 2000)). It is worth noticing that Equation 8.8 and Equation 8.9 actually give the parameterized expression for all configurations over the past for D_n.

Now, if the influence diagram contains multiple decision variables, then we are also interested in the parameterized expected utility functions for the remaining decision variables, i.e., $\rho(D_i, \mathrm{req}(D_i))(t), \forall 1 \leq i \leq n$. More precisely, we are looking for the parameterized expected utility function for decision variable D_i (for all $1 \leq i \leq n$) given the optimal policies for the future decision variables; recall that we wish to determine how much a parameter can be varied without changing the optimal strategy. The definition of a chance variable representation of a policy (Definition 3.3.8) allows us to use Equation 8.8 and Equation 8.9 iteratively, i.e., we work ourselves backwards in reverse temporal order: When visiting a decision node D_i we calculate $\rho(D_i, \mathrm{req}(D_i))(t)$ and replace it with its chance variable representation. Thus, when visiting a decision variable D all future decision variables have been replaced by their chance variable representations, and we can therefore consider D as being the last decision variable in I. This also implies that the structure of the parameterized expected utility function, for any decision variable D, corresponds to either Equation 8.8 or Equation 8.9 (depending on whether t is a utility parameter or a probability parameter).

So the main goal of sensitivity analysis therefore reduces to the task of identifying the coefficients of the parameterized expected utility functions.

8.4 One-way sensitivity analysis w.r.t. utility parameters

In this section we describe a method for analyzing how much a single parameter can be varied without changing the optimal strategy. If not stated otherwise, we shall restrict our attention to the case where all parameters belong to utility functions; note that utility parameters, in particular, are usually subject to second-order uncertainty (see e.g. (McCord and de Neufville 1984)) although advances have been made in learning the utilities from a database, see (Chajewska and Koller 2000).

Recall that an optimal policy for a decision variable is only a function over the required past for that decision variable.[2] However, an optimal policy may

[2]For the sake of simplifying notation we shall assume that for all decision variables D_i there

still contain superfluous information since some of the configurations over the required past may be impossible (i.e., configurations having zero probability). This gives rise to the following definition of a reduced optimal strategy.

Definition 8.4.1. Let I be an influence diagram and let $\Delta = \{\delta_D | D \in \mathcal{U}_D\}$ be an optimal strategy for I. Let $\widehat{sp}(req(D)) \subseteq sp(req(D))$ be the set of configurations over $req(D)$ having non-zero probability w.r.t. Δ. The function

$$\hat{\delta}_D : \widehat{sp}(req(D)) \to sp(D)$$

is said to be the *reduced optimal policy* for D in I if for all $\bar{c} \in \widehat{sp}(req(D))$: $\hat{\delta}_D(\bar{c}) = \delta_D(\bar{c})$. The set $\hat{\Delta} = \{\hat{\delta}_D | D \in \mathcal{U}_D\}$ of reduced optimal policies is said to be the *reduced optimal strategy* for I.

In order to determine the configurations $\widehat{sp}(req(D))$ we exploit the chance variable representation of a policy (see Definition 3.3.8):

1. Replace all decision variables with chance variable representations of their optimal policies.
2. Perform a propagation in the junction tree representation of the corresponding Bayesian network (value nodes are removed).
3. When the junction tree is consistent there exists at least one clique containing $req(D_j)$, for all $1 \leq j \leq n$, s.t. $P(req(D_j))$ can be read directly from that clique.

Consider now an influence diagram I and let $\bar{t} = (t_1, \ldots, t_m)$ be a set of parameters having initial values $\bar{t}^0 = (t_1^0, \ldots, t_m^0)$. As some configurations over the required past may be impossible, these configurations should have no impact on the sensitivity analysis. Hence, we define the admissible domain of a utility or probability parameter t_i based on the notion of a reduced optimal strategy:

Definition 8.4.2. Let I be an influence diagram and let $\bar{t} = (t_1, \ldots, t_m)$ be a set of utility or probability parameters in I having initial values $\bar{t}^0 = (t_1^0, \ldots, t_m^0)$. Let $\hat{\Delta}$ and $\hat{\Delta}_{t_i=t'}$ denote the reduced optimal strategies for I given the parameter values $\bar{t}^0$ and $(t_1^0, \ldots, t_{i-1}^0, t_i = t', t_{i+1}^0, \ldots, t_m^0)$, respectively. The *admissible domain* for t_i is the set, $\mathcal{R}_i$, of values satisfying:

$$t' \in \mathcal{R}_i \Leftrightarrow \hat{\Delta} = \hat{\Delta}_{t_i=t'}.$$

Let $\mathcal{R}(D_j)_i^\pi$ denote the values that can be assigned to t_i without changing $\hat{\delta}_{D_j}(\pi)$, given the optimal policies for the succeeding decision variables. The admissible domain $\mathcal{R}_i$ can then be characterized as follows.

is exactly one optimal policy for each configuration over the required past.

Theorem 8.4.1. Let I be an influence diagram with decision variables $\mathcal{U}_D = \{D_1, \ldots, D_n\}$, and let t_i be either a utility parameter or a probability parameter. If $\widehat{sp}(\text{req}(D_j))$ is defined as in Definition 8.4.1, then:

$$\mathcal{R}_i = \cap_{j=1}^{n} \cap_{\pi \in \widehat{sp}(\text{req}(D_j))} \mathcal{R}(D_j)_i^{\pi}.$$

Proof. If t_i is a utility parameter, then the proof is straightforward. If t_i is a probability parameter we exploit the assumption that an uncertain probability parameter cannot attain extreme values (i.e., 0 or 1), hence the set of impossible configurations over the required past is invariant of the value assigned to t_i. From this observation the proof follows immediately. □

Note that Theorem 8.4.1 does not hold if probability parameters are allowed to attain extreme values.

Now, consider the last decision variable D_n, and assume that the optimal policy for D_n specifies that $\delta_{D_n}(\pi) = d$, for some configuration $\pi \in \widehat{sp}(\text{req}(D_n))$. We are then interested in computing the set $\mathcal{R}(D_n)_i^{\pi}$ s.t.:

$$\begin{aligned}\mathcal{R}(D_n)_i^{\pi} = \{t' | \rho(D_n = d, \pi)(t_i = t') \geq \rho(D_n = d', \pi)(t_i = t'), \\ \forall d' \in sp(D_n) \backslash \{d\}\}.\end{aligned} \tag{8.13}$$

Since $\rho(D_n = d, \pi)(t_i)$ and $\rho(D_n = d', \pi)(t_i)$ are continuous in t_i this reduces to solving the following equality for all $d' \in sp(D_n) \backslash \{d\}$:

$$\rho(D_n = d, \pi)(t_i) - \rho(D_n = d', \pi)(t_i) = 0, \tag{8.14}$$

which is simply a linear equation. This also implies that $\mathcal{R}(D_n)_i^{\pi}$ is a convex set and moreover, $\mathcal{R}_i$ is a convex subset of the real line since the intersection of a collection of convex sets is itself a convex set; by Section 8.3.2 these considerations apply for all decision variables in the influence diagram.

In the remainder of this section we assume that the influence diagram has been evaluated w.r.t. the initial values $\bar{t}^0$ of the parameters. That is, we have identified an optimal strategy for I and, in particular, the expected utility function for each decision variable.

8.4.1 Algorithm 1

An immediate approach for calculating the required coefficients (needed to determine the admissible domains) is simply to perform multiple propagations in the strong junction tree representation of the influence diagram: One propagation w.r.t. the initial values of the parameters, followed by an additional propagation for each parameter in question. For instance, consider an influence diagram with three utility parameters t_1, t_2 and t_3. For this situation we perform a propagation w.r.t. $\bar{t}^0 = (t_1^0, t_2^0, t_3^0)$, followed by a propagation with the values $\bar{t}^1 = (t_1^0 + \epsilon_1, t_2^0, t_3^0)$, $\bar{t}^2 = (t_1^0, t_2^0 + \epsilon_2, t_3^0)$ and $\bar{t}^3 = (t_1^0, t_2^0, t_3^0 + \epsilon_3)$, where ϵ_1, ϵ_2 and ϵ_3 are chosen s.t. the optimal policies w.r.t. $\bar{t}^1$, $\bar{t}^2$ and $\bar{t}^3$ are equivalent to the ones obtained w.r.t. $\bar{t}^0$. From these propagations we can immediately identify all the coefficients needed for calculating the admissible domains, by exploiting that the expected utility functions are linear in the parameters. In general, this approach requires a propagation for each parameter since we need to consider the effect from the individual parameters, i.e., it does not suffice to perform a propagation w.r.t. $\bar{t}^0$ followed by a propagation for $\bar{t}^0 + \bar{\epsilon}$.

Note that ϵ_1, ϵ_2 and ϵ_3 can easily be found after propagating with the initial values $\bar{t}_0$ for the parameters: If we let

$$\epsilon_{D_i} = \min_{\pi \in \widehat{sp}(req(D_i))} \min_{d' \in sp(D_i) \setminus \{\delta_{D_i}(\pi)\}} (\rho(\delta_{D_i}(\pi), \pi) - \rho(d', \pi)),$$

then ϵ_1, ϵ_2 and ϵ_3 can be assigned any value within the interval $(-\epsilon; \epsilon)$, where $\epsilon = \min_{i=1}^n \epsilon_{D_i}$; observe that $\rho(\delta_{D_i}(\pi), \pi)$ and $\rho(d', \pi)$ can be read directly from the expected utility functions computed during the initial propagation.

8.4.2 Algorithm 2

As an alternative to the method above, we can use the explicit representation of the parameters $\bar{t}$ and work ourselves backwards in reverse temporal order, as outlined in Section 8.3.2.[3] More precisely, having introduced the parameter variables $\mathcal{T}$ we can employ lazy evaluation (Madsen and Jensen 1999), but instead of marginalizing out the parameter variables in $\mathcal{T}$ the elimination of these variables is postponed: When the decision variable D_n is the next variable to be marginalized out (not including the parameter variables), we have the expected utility function for D_n as a function over $\mathcal{T}$ (see Equation 8.8). Decision D_n is then replaced by its chance variable representation (allowing D_n to be eliminated as a chance variable), and the propagation proceeds as above until all decision variables have been investigated (iteratively postponing the elimination of the parameter variables); the chance variable representation of the optimal policy for D_n can be found by simply "simulating" the marginalization of the parameter variables.

This approach can be improved by not considering the parameter variables as actually being part of the influence diagram. That is, even though the parameterized utility functions are extended we do not represent the parameter variables explicitly in the influence diagram; note that their prior distributions are assumed to be known, so in the following we shall still make reference to these variables. Consider an influence diagram I with utility parameters $\bar{t}$ and assume

[3] This method is inspired by the myopic value of information analysis given in (Dittmer and Jensen 1997).

that these parameters are modeled by the parameter variables $\mathcal{T}$. If $\mathcal{T}$ consists of a single parameter variable T, then (as shown in Section 8.3) $\rho(D_n, \mathrm{req}(D_n))(t)$ can be found by marginalizing out the variables in $\mathcal{C}_n$ (not including T). If we are working with a set $\bar{t} = (t_1, \ldots, t_m)$ of parameters (for $m > 1$), then in order to determine $\rho(D_n, \mathrm{req}(D_n))(t_i)$ we would in principle need to marginalize out the parameter variables in $\mathcal{T} \setminus \{T_i\}$ s.t. the parameterized expected utility function is only a function over t_i; unfortunately, this should be done for all parameter variables in $\mathcal{T}$. However, by observing that $\rho(D_n = d, \mathrm{req}(D_n) = \pi)(\bar{t})$ is linear in the parameters ($\rho(d, \pi)(\bar{t})$ defines a hyperplane in $\mathfrak{R}^m$), the partial derivatives

$$\frac{\partial(\rho(D_n = d, \mathrm{req}(D_n) = \pi)(\bar{t}))}{\partial t_i}$$

are constant functions over the remaining parameters. This implies that the slope in Equation 8.8 can be found by conditioning on an arbitrary configuration over the remaining parameter variables, i.e., they need not be marginalized out. Moreover, since $\rho(D_n, \mathrm{req}(D_n))(\bar{t}_0)$ is known (from the initial evaluation) we can solve Equation 8.8 w.r.t. the constant. Hence we obtain all coefficients of $\rho(D_n, \mathrm{req}(D_n))(t_i)$ for each parameter t_i in $\bar{t}$. These observations also apply when calculating $\rho(D_j, \mathrm{req}(D_j))(t)$ for $1 \leq j < n$, since each time a decision variable is visited we can substitute it with its chance variable representation as described in Section 8.3.2. In this respect we simulate the marginalization of the parameter variables when a decision variable is visited.

From the description above, one-way sensitivity analysis w.r.t. the utility parameters $\bar{t}$ can be performed by one propagation in the strong junction tree representation of the influence diagram; recall that we assume that an initial propagation has been performed w.r.t. $\bar{t}^0$, and that parameter variables are not marginalized out during the propagation.

Algorithm 8.4.1. To perform one-way sensitivity analysis w.r.t. the utility parameters $\bar{t} = (t_1, \ldots, t_m)$, do:

1. Set $j := n$ and $\mathcal{R}_i := [t_i^{\min}; t_i^{\max}]$, for all $i = 1, \ldots, m$.
2. **Repeat**
    a) Propagate to the clique containing D_j.
    b) Compute $\mathcal{R}(D_j)_i^\pi$ (see Equation 8.14) w.r.t. parameter t_i, for all $1 \leq i \leq m$.
    c) Let $\mathcal{R}(D_j)_i = \cap_{\pi \in \widehat{sp}(\mathrm{req}(D_j))} \mathcal{R}(D_j)_i^\pi$ and set $\mathcal{R}_i := \mathcal{R}_i \cap \mathcal{R}(D_j)_i$, for all $1 \leq i \leq m$.
    d) Set $j := j - 1$ and continue the propagation from the current clique.
3. **Until** $j = 0$.

□

Compared to a single propagation in the strong junction tree representation of the influence diagram (with no parameter variables), the main computational increase follows from the extension of the parameterized utility functions; in the worst case, all extended utility functions are combined during propagation. However, by employing lazy evaluation we can use the distributive law when marginalizing out a chance variable from a set of utility functions (this procedure is also described in (Madsen and Jensen 1999)). For instance, marginalizing out a chance variable A from a probability potential ϕ and a set $\{\psi_1, \psi_2, \ldots, \psi_l\}$ of utility functions yields:

$$\sum_A \phi \left(\sum_{i=1}^{l} \psi_i \right) = \sum_{i=1}^{l} \left(\sum_A \phi \cdot \psi_i \right).$$

That is, the extended utility functions are not combined when marginalizing out a chance variable.

Finally, it is important to notice that given a preference ordering over the outcomes, a utility parameter is only allowed to vary within the interval bounded by the utilities assigned to the other outcomes.[4] For instance, assume that we have a single utility function and assume that the preference ordering over the outcomes c_i and c_j specifies that $c_i \preceq c_j$. In this situation the admissible domain for t_i is restricted by the inequality $t_i \leq t_j^0$. Thus, restrictions may be placed on the admissible domain of a parameter and, in particular, if the restricted domain is a proper subset of the admissible domain, then the model is insensitive to the parameter in question. Note that such restrictions can immediately be adopted in the proposed framework by simply reducing $\mathcal{R}(D_j)_i^\pi$ accordingly, see Theorem 8.4.1 and Equation 8.13.

8.5 n-way sensitivity analysis w.r.t. utility parameters

When performing n-way sensitivity analysis we wish to determine the joint effect of varying a set of utility parameters $\bar{t} = (t_1, \ldots, t_m)$. More precisely, we may look for a ***safe ball*** $\mathcal{S}$ of values which the parameters can be assigned without changing the optimal strategy found w.r.t. their initial values $\bar{t}_0$.

Consider decision variable D_n and assume that the optimal policy for D_n specifies that $\delta_{D_n}(\pi) = d$ for some configuration $\pi \in \widehat{\mathrm{sp}}(\mathrm{req}(D_n))$. We then look for a safe ball $\mathcal{S}(D_n)^\pi \subseteq \mathfrak{R}^m$ s.t. for all $d' \in \mathrm{sp}(D_n) \setminus \{d\}$ and for all $\bar{t}' \in \mathcal{S}(D_n)^\pi$:

$$\rho(d, \pi)(\bar{t}') \geq \rho(d', \pi)(\bar{t}').$$

[4]We may assume that this preference ordering is consistent with the initial values assigned to the utility parameters. Hence, the preference ordering can be read directly from these values.

Notice that we do not require $\mathcal{S}(D_n)^\pi$ being the largest subset of $\mathfrak{R}^m$ satisfying the inequality.

Since we consider the parameters simultaneously, the admissible domain of a parameter t_i will, in general, depend on the values assigned to the other parameters. Thus, we characterize the safe ball $\mathcal{S}$ as the set of parameter values $\bar{t}'$ being within a given "distance" from $\bar{t}^0$: As the parameterized expected utility functions are linear in the utility parameters we shall use the geometric measure to describe the distance:

$$\mathrm{dist}(\bar{t}^0, \bar{t}') = \sqrt{\sum_{i=1}^{m} (t_i' - t_i^0)^2}.$$

Identifying a safe ball for the parameters then corresponds to minimizing the distance from $\bar{t}^0$ to the hyperplane:

$$\rho(d, \pi)(\bar{t}) - \rho(d', \pi)(\bar{t}) = 0, \tag{8.20}$$

for all $d' \in sp(D_n) \setminus \{d\}$. The distance from $\bar{t}^0$ to this hyperplane is simply the distance from $\bar{t}^0$ to $\bar{t}'$, where $\bar{t}'$ satisfies Equation 8.20 and the line segment between $\bar{t}^0$ and $\bar{t}'$ is orthogonal to the hyperplane. Thus, we define the safe ball $\mathcal{S}(D_n)^\pi$ for decision D_n, given $\pi \in \widehat{sp}(\mathrm{req}(D_n))$, as:

$$\mathcal{S}(D_n)^\pi = \{\bar{t} | \mathrm{dist}(\bar{t}_0, \bar{t}) \leq \mathrm{dist}(\bar{t}_0, \bar{t}')\}$$

and the safe ball $\mathcal{S}(D_n)$ for D_n as

$$\mathcal{S}(D_n) = \cap_{\pi \in \widehat{sp}(\mathrm{req}(D_n))} \mathcal{S}(D_n)^\pi.$$

Finally the safe ball for the parameters $\bar{t}$ is given by:

$$\mathcal{S} = \cap_{i=1}^{n} \mathcal{S}(D_i). \tag{8.23}$$

The safe ball $\mathcal{S}$ can equivalently be characterized in terms of its radius $r_\mathcal{S}$. That is, if r_{D_i} denotes the radius of the safe ball $\mathcal{S}(D_i)$, then $r_\mathcal{S} = \min_{i=1}^{n} r_{D_i}$ and $\mathcal{S} = \{\bar{t}' | \mathrm{dist}(\bar{t}^0, \bar{t}') \leq r_\mathcal{S}\}$.

Example 8.5.1. Suppose that we have a ternary decision variable D, and two utility parameters t_1 and t_2 with initial values t_1^0 and t_2^0, respectively. Given a configuration π over the required past for D, we seek the distance from (t_1^0, t_2^0) to the lines

$$\begin{aligned} a_1^1 t_1 + a_2^1 t_2 + c_1 &= 0 \\ a_1^2 t_1 + a_2^2 t_2 + c_2 &= 0 \end{aligned}$$

defined by Equation 8.20 (see Figure 8.2).

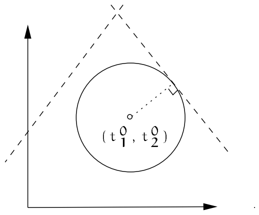

Figure 8.2: The safe ball for D is described by identifying the minimum distance from (t_1^0, t_2^0) to the lines defined by the parameterized expected utility function.

The minimum distance $d_{\bar{t}^0}^{\min}$ is then simply:

$$d_{\bar{t}^0}^{\min} = \min\left\{ \frac{|a_1^1 t_1^0 + a_2^1 t_2^0 + c_1|}{\sqrt{(a_1^1)^2 + (a_2^1)^2}}, \frac{|a_1^2 t_1^0 + a_2^2 t_2^0 + c_2|}{\sqrt{(a_1^2)^2 + (a_2^2)^2}} \right\}$$

Hence, $\mathcal{S}(D)^\pi = \{\bar{t}' | \mathrm{dist}(\bar{t}^0, \bar{t}') \leq d_{\bar{t}^0}^{\min}\}$. □

It should be noticed that the safe ball specifies a set of values which can be assigned to the parameters without changing the reduced optimal strategy. However, there may exist configurations $\bar{t}' \notin S$ which generate a reduced optimal strategy equivalent to the reduced optimal strategy generated by $\bar{t}^0$.

Finally, based on the analysis above the algorithms described in Section 8.4 can immediately be adopted for n-way sensitivity analysis. Moreover, the following properties of one-way sensitivity analysis also apply to n-way sensitivity analysis: i) we can exploit the distributive law when marginalizing out the variables, and ii) we need not consider all parameters simultaneously since the expected utility functions are linear in the parameters, i.e., their coefficients can be calculated independently. The latter property is especially useful in case we have a single utility function with multiple parameters, i.e, the proposed method allows us to establish a tradeoff between time and space: We may partition the parameter set into disjoint subsets, and then identify the required coefficients for each of these subsets. This property also implies that having initially performed one-way sensitivity analysis w.r.t. $\bar{t}$ we have all the coefficients necessary for calculating a safe ball for $\bar{t}$, i.e., no additional propagations are needed for n-way sensitivity analysis.

Note again that given a preference ordering over the outcomes, a utility parameter is only allowed to vary within the interval bounded by the utilities assigned to the other outcomes. Analogously to the case of one-way sensitivity analysis, we can restrict the safe ball according to such a preference ordering by simply introducing additional linear constraints when calculating the safe ball. For instance, assume that we have a single utility function and assume that the preference ordering over the outcomes c_i and c_j specifies that $c_i \preceq c_j$. If n-way sensitivity analysis is performed w.r.t. t_i and t_j, then the safe ball is restricted by the inequality $t_i \leq t_j$.

8.6 Sensitivity analysis w.r.t. probability parameters

When performing sensitivity analysis with probability parameters or with a mixture of utility and probability parameters, the expected utility functions are no longer linear in the parameters. In general, for one-way sensitivity analysis the expected utility functions are quotients of two linear functions, and for n-way sensitivity analysis the expected utility functions are quotients of two multi-linear functions. In what follows we consider the case where all parameters belong to conditional probability potentials, and, unless stated otherwise, we assume that the denominator is a non-constant function over the parameters considered; structurally, it is easy to determine when the denominator evaluates to a constant (see Section 8.3.2).

Analogously to the method described in Section 8.4, one-way sensitivity analysis w.r.t. a probability parameter t_i corresponds to determining the admissible domain $\mathcal{R}_i$ for t_i. Again, consider the last decision variable D_n, and assume that the optimal policy for D_n specifies that $\delta_{D_n}(\pi) = d$, for some configuration $\pi \in \widehat{\mathrm{sp}}(\mathrm{req}(D_n))$. So we seek to identify the admissible domain $\mathcal{R}(D_n)_i^\pi$ s.t.:

$$\begin{aligned}\mathcal{R}(D_n)_i^\pi = \{t' | \rho(D_n = d, \pi)(t_i = t') \geq \rho(D_n = d', \pi)(t_i = t'),\\ \forall d' \in \mathrm{sp}(D_n) \backslash \{d\}\}.\end{aligned}$$

Similar to the case where t_i is a utility parameter, this reduces to solving the following equality for all $d' \in \mathrm{sp}(D_n)\backslash\{d\}$:

$$\rho(D_n = d, \pi)(t_i) - \rho(D_n = d', \pi)(t_i) = 0.$$

As both $\rho(d_1, \pi_{D_i})(t_i)$ and $\rho(d_2, \pi_{D_i})(t_i)$ are quotients of two linear functions, finding a solution to this equality reduces to finding the root(s) of a second order polynomial. If the polynomial is concave, then $\mathcal{R}(D_n)_i^\pi$ is a convex set, and by arguments similar to the case of one-way sensitivity analysis w.r.t. utility parameters we have that the admissible domain $\mathcal{R}_i$ is a convex set. On the other hand, if the polynomial is convex, it does not necessarily specify a convex set for t_i (we may have two disjoint intervals). However, as we look for an interval in which t_i can be varied continuously without changing the optimal strategy we let $\mathcal{R}(D_n)_i^\pi$ be the interval containing t_i, i.e., the admissible domain $\mathcal{R}(D_n)_i^\pi$ is still a convex set. Moreover, as with utility parameters we can apply the analysis above to all decision variables in the influence diagram by iteratively substituting the decision variables with their chance variable representations.

It is now easy to see that the algorithm for one-way sensitivity analysis w.r.t. utility parameters can be adopted for probability parameters. Unfortunately, though, when visiting a decision variable D all parameter variables except T_i must be marginalized out in order to determine the coefficients for $\rho(D = d, \pi)(t_i)$ and $\rho(D = d', \pi)(t_i)$, given $\pi \in \widehat{sp}(\text{req}(D))$. Moreover, we can no longer exploit the distributive law when marginalizing out the variables; hence, with a large number of parameters we need to partition the parameters into subsets and apply the algorithm for each of these subsets. Note that the coefficients can also be found by multiple propagations in the strong junction tree representation of the influence diagram. However, as opposed to the case with utility parameters we need to perform two propagations for each probability parameter.

n-way sensitivity analysis w.r.t. probability parameters is in principle performed analogously to n-way sensitivity analysis w.r.t. utility parameters. That is, we seek a safe set $\mathcal{S}$ based on the minimum distance from $\bar{t}^0$ to the hypersurface

$$\rho(d, \pi)(\bar{t}) - \rho(d', \pi)(\bar{t}) = 0, \forall d' \in sp(D_n) \setminus \{d\}. \tag{8.26}$$

However, as both $\rho(d, \pi)(\bar{t})$ and $\rho(d', \pi)(\bar{t})$ are quotients of two multi-linear functions the number of coefficients grows exponentially in the number of parameters. This also implies, that in order to calculate the coefficients (using multiple propagations), the number of necessary propagations grows exponentially in the number of parameters. Analogously, by introducing parameter variables we would need to model all parameters explicitly and, in the worst case, all extended probability potentials are combined during propagation. Thus, in both of these cases the computational complexity would be impractical (even for relatively small sized domains), and one appears bound to use approximate algorithms e.g. probabilistic sensitivity analysis (see Section 8.2).

8.7 Discussion

One of the main difficulties when representing decision problems using influence diagrams or valuation based systems is eliciting the utilities and the probabilities. In this chapter we have presented exact non-probabilistic methods for performing one-way and n-way sensitivity analysis in influence diagrams w.r.t. these types of parameters. The methods are based on an explicit representation of the parameters in question, and the calculations are performed in the underlying junction tree representation. Moreover, we have shown that when one-way sensitivity analysis has been performed w.r.t. the utility parameters in question, then n-sensitivity analysis comes free of charge, i.e., no additional propagations are needed.

When modeling the parameters explicitly we obtain probability potentials and utility functions which are completely symmetric w.r.t. the parameter variables, for each non-parameterized configuration of the parent set (see Equation 8.5). Such types of regularities can efficiently be exploited using a representation similar to *probability trees* (Cano et al. 2000b), i.e., the size of the probability potentials and the utility functions does not necessarily increase exponentially in the number of parameter variables.

Furthermore, many of the calculations needed for sensitivity analysis can be avoided by initially performing a structural analysis of the influence diagram. For instance, if the parameter t_i belongs to the utility function ψ, then we only need to consider the decision variables for which ψ is relevant.

Compared to probabilistic sensitivity analysis using Monte Carlo methods, the computational complexity of the proposed method is in general smaller when considering one-way and n-sensitivity analysis w.r.t. utility parameters (and one-way sensitivity analysis w.r.t. probability parameters). For instance, (Bielza et al. 2000) describes an application of EVPI based on Monte Carlo simulations. In this analysis, each of the parameter sets under investigation (including singletons) was analyzed based on 100 samples, i.e., the influence diagram was evaluated 100 times for each parameter set; all parameters were assumed independent. Note that when the utility parameters are dependent (which is the case when a preference ordering is specified over the outcomes) the number of required samples will in general be larger compared to the case where the parameters are independent. In comparison, to perform both one-way and n-way sensitivity analysis w.r.t. all the utility parameters the proposed method would require at most $m + 1$ propagations, where m is the number of parameters. On the other hand, when doing n-way sensitivity analysis w.r.t. probability parameters, probabilistic sensitivity analysis appears significantly more efficient.

Part IV

Decision making based on non-expected utility

Chapter 9

Rational decision making based on rank-dependent utility

Non-expected utility theories, such as rank dependent utility theory, have been proposed as alternative models to expected utility theory in decision making under risk. These models do not share the separability property of expected utility theory hence, in dynamic decision problems, the sophisticated strategy is likely to be dominated w.r.t. stochastic dominance. Although a rational non-expected utility behavior is necessarily a non-consequentialist behavior (i.e., the choice in a subtree depends on what happens in the rest of the tree), we show that it is nonetheless possible to define a procedure which: i) involves a "rolling back" of the decision tree; and ii) selects a non-dominated strategy that realizes a compromise between the decision maker's discordant goals at the different decision nodes. Relative to the computations involved in the standard expected utility evaluation of a decision problem, the main computational increase is due to the identification of non-dominated strategies by linear programming. A simulation, using the rank dependent utility criterion, confirms the computational tractability of the model.

The content of this chapter is based on (Nielsen and Jaffray 2001a) and (Nielsen and Jaffray 2001b).

9.1 Introduction

The standard normative model for decision making under risk, i.e., under uncertainty represented by probabilities, is *expected utility theory*: the value of a decision is the sum of the utilities of its potential consequences weighted by the probabilities of their occurrence. Thus its prescriptive use requires the elicitation of a personal parameter of the decision maker, the so-called von Neumann-Morgenstern (vNM) utility function. Unfortunately, it has turned out that expected utility theory has a poor performance when considered as a descriptive model, in particular it is unable to account for experimentally well established certainty/possibility effects, see e.g. (Kahneman and Tversky 1979). This makes the vNM utility dependent on the elicitation method used (see (McCord and de Neufville 1984)) and, furthermore, makes the model unable to explain certain choices if it must account for some others.

The prescriptive approach of decision aiding cannot simply disregard these shortcomings of expected utility theory. The Analyst's task is to propose a solution which the decision maker is presumed to undertake. However, the decision maker is usually not prepared to blindly follow the Analyst's advice before he has gained sufficient confidence in the model used. Since the decision maker is generally only certain about his preferences in extremely simple choice situations (this is the very reason why these are used in parameter elicitation), he will test the adequacy of the model on such situations. Thus, the decision maker is likely to reject the model as a result of the inability of expected utility theory to simultaneously account for all of his choices. To summarize, a prescriptive model has to respect a balance between descriptive and normative qualities, which is not quite achieved by expected utility theory. This problem has motivated the consideration of various alternative models, generically known as non-expected utility theories. Most of these models depart from expected utility theory (which they nonetheless include as a special case) by using criteria which not only have lost the additive separability of the expected utility criterion but, furthermore, fail to satisfy even weaker separability requirements.

Non-separability seems likely to have disastrous implications from an operational point of view. Many applications deal with sequential decision problems, which involve a tremendous number of feasible strategies (sequences of conditional decisions), and they only become computationally tractable with expected utility criteria, which support *backward induction* (dynamic programming). When this principle is invalid, which is generally the case with non-separable criteria, one appears bound to directly evaluate and compare all strategies.

However, a closer look at the implications of non-separability shows that non-expected utility theories raise a preliminary question of arbitrage between different preferences at different times concerning future decisions. As we shall see, solutions can be proposed which construct a compromise strategy by a procedure involving backward induction. Thus, non-separable preferences do not automatically mean computational intractability.

Non-expected utility theories have also been criticized from the normative point of view, as they appear to make the decision maker open to manipulations that can induce sure losses or more generally, make the decision maker likely to choose a dominated strategy in sequential decision problems (Wakker 1998).

However as initially noticed by Machina (1989), all the arguments advanced against non-expected utility criteria rest on the assumption that the decision maker is a *consequentialist*, i.e., his choice at a decision node depends only on data relative to the subtree rooted at that node, and it is therefore not influenced by the past nor by counterfactuals. Machina (1989) claims that it is inappropriate to impose the property of consequentialism on non-expected utility behavior since the very notion of a non-expected utility criterion implies a type of non-separability which contradicts the consequentialist assumption.

By renouncing consequentialism, the decision maker can easily get around the pitfalls of dynamic decision making. This is for instance the case with *resolute choice*, a type of behavior discussed in (McClennen 1990). The resolute decision maker initially commits himself to a strategy, i.e., initially chooses a strategy and never deviates from it later. If, as generally understood, the strategy to which he commits himself is the optimal strategy for his initial preferences, the properties guaranteed by this criterion are automatically satisfied. However the ability of the decision maker to commit himself is a disputable matter, and so is the restriction of commitment to this sole strategy.

To accommodate this problem Jaffray (1999) elaborates on resolute choice and suggests a model in which the decision maker's multiple *Selves* (associated with the decision maker's knowledge and state-of-mind at different times (McClennen 1990)) collaborate and attempt to select a compromise strategy. This strategy should achieve the goals they have in common, such as being non-dominated, without requiring from them excessive renunciation.

In this chapter we present an operational approach to the model by Jaffray (1999), where the non-expected utility criterion attributed to the decision maker is rank dependent utility (Quiggin 1982). We propose an algorithm, for solving decision trees, which (if a solution exists) produces a non-dominated strategy which is acceptable by all the Selves. The algorithm relies on backward induction and is somewhat similar to the average-out and fold-back algorithm for solving decision trees in the expected utility model (Raiffa and Schlaifer 1961).

9.2 Preliminaries

Recall that a decision tree T is a rooted tree, where the nodes $\mathcal{U}$ can be partitioned into three disjoint subsets: decision nodes, chance nodes and value nodes (or *consequences*); in this chapter we shall use $\mathcal{C}$ to denote the set of value nodes/consequences to emphasize the distinction between a consequence and the utility assigned to a consequence.

A node X is said to *precede* another node Y in the decision tree T if there exists a directed path from X to Y (analogously, Y is said to *succeed* X in T). A decision node D' is said to be an *immediate decision successor* of another decision node D if D' succeeds D and no other decision node lies on the path between D and D'. The set of decision nodes which constitute the immediate decision successors of a decision node D is denoted $\mathcal{D}_{\mathsf{D}}$. The *chance node successors* of a decision node D are the chance nodes which succeed D but do not succeed any immediate decision successor of D. The set of chance node successors for a decision node D is denoted $\mathcal{C}_{\mathsf{D}}$.

Definition 9.2.1. Let T be a decision tree, and let $\mathcal{U}^{\Delta} \subseteq \mathcal{U}$ be a set of nodes containing:

i) The root node.

ii) Exactly one child of each $\mathsf{X} \in \mathcal{U}^{\Delta}_{\mathsf{D}} = \mathcal{U}_{\mathsf{D}} \cap \mathcal{U}^{\Delta}$.

iii) All children of each $\mathsf{X} \in \mathcal{U}^{\Delta}_{\mathsf{C}} = \mathcal{U}_{\mathsf{C}} \cap \mathcal{U}^{\Delta}$.

Then the set of arcs $\Delta = \{(\mathsf{Y},\mathsf{Z}) | \mathsf{Y} \in \mathcal{U}^{\Delta}_{\mathsf{D}}, \mathsf{Z} \in \mathcal{U}^{\Delta}\} \subseteq \mathcal{E}$ is said to be a *strategy* for T. The restriction of a strategy to a subtree $\mathsf{T}(\mathsf{X})$, which in fact is a strategy in $\mathsf{T}(\mathsf{X})$, is called a *substrategy*.

Notice that this definition of a strategy differs from the definitions given in the previous chapters, where a strategy specifies a decision option for each $\mathsf{X} \in \mathcal{U}_{\mathsf{D}}$. In the remainder of this chapter we assume that a decision tree always has a decision node as root node.

For every chance node X, we assume that a probability distribution $\mathsf{P}(\mathsf{X}|\mathfrak{past}(\mathsf{X}))$ specifies the probabilities of the achievable outcomes, conditioned on the outcomes and decisions belonging to the path from the root to X, $\mathfrak{past}(\mathsf{X})$. For any substrategy with root X, these probabilities are sufficient (by repeated use of the theorem of compound probabilities) for determining the conditional probability of reaching a given value node (consequence) from X given $\mathfrak{past}(\mathsf{X})$. A situation where any (sub)strategy generates a probability distribution over the leafs is known as a situation of *decision making under risk*.

9.2.1 Single stage decision making

Preferences among lotteries

Consider a decision tree with a unique decision node being root. Assume that events have probabilities (situation of *risk*), so that each feasible decision option can be associated with a (single stage) *lottery*, i.e., the discrete probability distribution it generates on the consequence set:

$$L = (c_1, p_1; ...; c_i, p_i; ...; c_n, p_n),$$

where $p_i \geq 0$ $(i = 1, ..., n)$ denotes the probability of obtaining the consequence c_i and $\sum_{i=1}^{n} p_i = 1$ (graphically, a lottery L can be represented as in Figure 9.1).

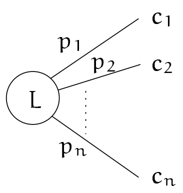

Figure 9.1: A graphical representation of the single stage lottery $L = (c_1, p_1; c_2, p_2; \ldots; c_n, p_n)$.

With $L' = (c_1, p'_1; ...; c_i, p'_i; ...; c_n, p'_n)$, $qL + (1-q)L'$ denotes a *compound lottery* formed by the *mixture* of the two lotteries L and L', giving consequence c_i with probability $qp_i + (1-q)p'_i$.

We assume that the decision maker's preferences at the root only depends on the lotteries generated by the decisions. Note that this assumption implies both that: i) the only relevant aspect of an event is its probability (*events anonymity)* and ii) the process of resolution of uncertainty is irrelevant (*reduction of compound lotteries*). We can then directly define the decision maker's preferences by a weak order $\succsim$ on the lottery set: as usual, $L \succsim L'$ is read as "L is preferred or indifferent to L'". The derived relations, $\succ$, $\precsim$, $\prec$, $\sim$ have their standard meaning, and we shall assume that this preference ordering can be expressed by a real function V (a *preference function*) satisfying $V(L) \geq V(L') \Leftrightarrow L \succsim L'$, hence also:

$$V(L) > V(L') \Leftrightarrow L \succ L' \text{ and } V(L) = V(L') \Leftrightarrow L \sim L'.$$

Having established this preference ordering, the reduction of compound lotteries can be illustrated as in Figure 9.2.

From the reduction of compound lotteries we will not distinguish between a compound lottery and its probabilistically equivalent single stage lottery that is, we implicitly refer to single stage lotteries when making subsequent references to lotteries.

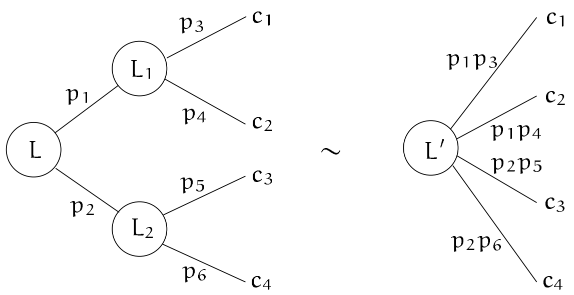

Figure 9.2: The figure illustrates a compound lottery and its probabilistically equivalent single stage lottery; the decision maker is assumed to be indifferent between L and L'.

Expected utility

Under expected utility theory, the preference function $V(\cdot) = EU(\cdot)$ on a lottery L takes on the form:

$$EU(L)_u = EU(c_1, p_1; ...; c_i, p_i; ...; c_n, p_n)_u = \sum_{i=1}^{n} u(c_i)p_i, \tag{9.3}$$

where the function $u(\cdot)$ is a personal characteristic of the decision maker, his von Neumann-Morgenstern (vNM) utility function.[1] The identification of a constant lottery with its unique consequence makes $u(\cdot)$ a utility representing the decision maker's preferences under certainty. Moreover, since $u(\cdot)$, like $EU(\cdot)$, is *cardinal* (unique up to affine transformations), its shape can receive an interpretation; in particular, when the consequence set $\mathcal{C}$ is a convex subset of a linear space, a concave $u(\cdot)$ characterizes *risk aversion*.[2]

The preference function of expected utility theory, together with the reduction of compound lotteries, implies the notion of an *optimal strategy*: From the reduction of compound lotteries we have that any strategy Δ can be identified with a single stage lottery L_Δ; thus, we say that a strategy Δ is an optimal strategy w.r.t. the utility function u if and only if:

$$EU(L_\Delta)_u \geq EU(L_{\Delta'})_u, \forall \Delta' \neq \Delta.$$

The cornerstone of expected utility theory is the following property which is clearly implied by the linearity of the preference function w.r.t. the probabilities.

[1] In this chapter, a utility function is denoted by $u(\cdot)$ to make the notation consistent with existing work within this area.

[2] Informally, a function $u(\cdot)$ is concave when the hypersurface representing it is above all its chords: $u(qc + (1-q)c') \geq qu(c) + (1-q)u(c')$, for $q \in (0, 1)$. The function $u(\cdot)$ is convex if $-u(\cdot)$ is concave.

Axiom 1 (Independence axiom). For all lotteries L, L', and L^* and for all q in $(0, 1]$, the lottery L is (strictly) preferred to the lottery L' if and only if the mixture $qL + (1 - q)L^*$ is (strictly) preferred to the mixture $qL' + (1 - q)L^*$.

Consider decision makers who have the same preference, $\succsim$, under certainty, hence with $u(\cdot)$ in $\mathcal{V} = \{u(\cdot) : \mathcal{C} \to \mathfrak{R} | u(c) \geq u(c') \Leftrightarrow c \succsim c'\}$. Although the model allows lotteries to be ranked differently by different decision makers, it is generally agreed that consistency with the following partial ordering is a natural rationality requirement.

Definition 9.2.2. The lottery L *stochastically dominates lottery* L' *at the first order* when:

- $\sum_{\{i:u(c_i)\leq k\}} p_i \leq \sum_{\{i:u(c_i)\leq k\}} p'_i$, for all $k \in \mathfrak{R}$ and
- $\sum_{\{i:u(c_i)\leq k\}} p_i < \sum_{\{i:u(c_i)\leq k\}} p'_i$, for some $k \in \mathfrak{R}$.

Note that this relation does not depend on the choice of $u(\cdot)$ in $\mathcal{V}$. The justification of this requirement relies on the possibility of exhibiting a situation in which two decisions exist, which generate L and L' respectively, and where the consequences of the first decision are, whatever happens, preferred (and for some event strictly preferred) to those of the second decision (point-wise dominance).

Expected utility criteria are necessarily consistent with first order stochastic dominance. Actually, this property provides a characterization of first order stochastic dominance, as stated in the following theorem due to Hadar and Russell (1969), which is exploited later in this chapter.

Theorem 9.2.1. Let $\mathcal{V}$ be the set of utility functions representing the decision maker's ordering on the consequence set $\mathcal{C}$, and let L and L' be two lotteries. Then the following statements are equivalent:

i) L stochastically dominates L' at the first order.

ii) $\forall u \in \mathcal{V} : EU(L)_u \geq EU(L')_u$ and $\exists u \in \mathcal{V} : EU(L)_u > EU(L')_u$.

The theorem also implies that if there exists a utility function $u \in \mathcal{V}$ s.t. $EU(L)_u > EU(L')_u$ then L is not first order stochastically dominated by L'; this property also applies if, for all utility functions $u \in \mathcal{V}$ it holds that $EU(L)_u \geq EU(L')_u$.

Another important type of stochastic dominance is *second order stochastic dominance* which is involved in a possible definition of risk aversion.

Definition 9.2.3. The lottery L *stochastically dominates lottery* L' *at the second order* when:

- $\sum_{i:u(o_i)\leq k}\left(\sum_{j:u(o_j)\leq u(o_i)} p_j\right) \leq \sum_{i:u(o_i)\leq k}\left(\sum_{j:u(o_j)\leq u(o_i)} p'_j\right)$, for all $k \in \mathfrak{R}$ and
- $\sum_{i:u(o_i)\leq k}\left(\sum_{j:u(o_j)\leq u(o_i)} p_j\right) < \sum_{i:u(o_i)\leq k}\left(\sum_{j:u(o_j)\leq u(o_i)} p'_j\right)$, for some $k \in \mathfrak{R}$.

As for first order stochastic dominance, this relation does not depend on the choice of $u(\cdot)$ in $\mathcal{V}$ and, like first order stochastic dominance, the definition of second order stochastic dominance can be formulated in terms of a specific class of utility functions; for second order stochastic domaince we consider the same set of utility functions as for first order stochastic dominance except that the utility functions are also required to be concave.

Theorem 9.2.2. Suppose the consequence set $\mathcal{C}$ is a convex subset of a linear space, and let L and L' be two lotteries. Then the following statements are equivalent:

i) L stochastically dominates L' at the second order.

ii) For every concave $u \in \mathcal{V} : EU(L)_u \geq EU(L')_u$, and there exists a concave $u \in \mathcal{V} : EU(L)_u > EU(L')_u$.

In the remainder of this chapter we shall mainly consider stochastic dominance of the first order thus, if not stated otherwise, any future reference to dominance will implicitly refer to first order stochastic dominance.

A non-expected utility model: Rank dependent utility

Motivated by experimental observations of phenomena inconsistent with expected utility theory, in particular the *Allais Paradox* (Allais 1979) and (Kahneman and Tversky 1979), several authors have suggested an alternative to expected utility theory, now most often called *rank dependent utility* theory, see e.g. (Allais 1988) and (Yaari 1987). This model was, however, first presented and axiomatized by Quiggin (1982) under the name of *anticipated utility* theory.

In the rank dependent utility model, there are two parameters: i) a utility function $u(\cdot)$ which, as the vNM utility function of expected utility theory, is cardinal and represents the decision maker's preferences under certainty, and ii) a *probability transformation* (or *weighting*) function $q(\cdot)$, which is strictly increasing from $[0,1]$ onto itself. The utility $V(L)_{u,q}$ of a lottery L is determined as follows. First, the components of L are re-indexed in increasing order of their utility:

$$L = (c_1, p_1; ...; c_i, p_i; ...; c_n, p_n) \text{ with } u(c_1) \leq ... \leq u(c_i) \leq ... \leq u(c_n).$$

Then, its utility is evaluated as:

$$V(L)_{u,q} = \sum_{i=1}^{n} u(c_i) \left[q\left(\sum_{j=i}^{n} p_j\right) - q\left(\sum_{j=i+1}^{n} p_j\right) \right]$$

or, equivalently as

$$V(L)_{u,q} = u(c_1) + \sum_{i=2}^{n} [u(c_i) - u(c_{i-1})]\, q\left(\sum_{j=i}^{n} p_j\right).$$

The crucial idea underlying the rank dependent utility model is that the utility of each consequence is given a variable weight, which depends on both its probability and on its ranking w.r.t. the other consequences of the lottery. The second expression suggests that the decision maker bases his evaluation on the probabilities of achieving (at least) some particular utility levels.

From the second expression of $V(L)_{u,q}$ it is clear that, in rank dependent utility theory, preferences are consistent with first order stochastic dominance; the function $q(\cdot)$ is strictly increasing. It can be shown that consistency with ***strong risk aversion*** (L stochastically dominates L' at the second order and $EU(L) = EU(L')$ imply $L \succsim L'$) also holds if and only if $q(\cdot)$ is convex and $u(\cdot)$ is concave (Chew et al. 1987). Sufficient conditions for the preferences w.r.t. *weak risk aversion* ($EU(L) \succsim L$) are also known (Chateauneuf and Cohen 1984); these conditions also involve both $q(\cdot)$ and $u(\cdot)$, see (Currim and Sarin 1989) and (Wu and Gonzalez 1996). Examples include:

$$\begin{aligned} q(p) &= \frac{p^{\gamma}}{(p^{\gamma} + (1-p)^{\gamma})^{\delta}} \text{ (See Figure 9.3a)} \\ q(p) &= \exp(-(-\ln(p))^{\omega}) \text{ (See Figure 9.3b)} \end{aligned}$$

Note that rank dependent utility theory no longer satisfies Axiom 1.

9.2.2 Dynamic decision making

Dynamic consistency

Consider a decision problem represented by a decision tree, and suppose that at each decision node the decision maker has a preference ordering on the substrategies that are feasible at that node. We say that the decision maker is *dynamically consistent* if substrategies of optimal strategies are optimal substrategies. More precisely, assume that Δ_1 is an optimal strategy of the subtree rooted at decision node D_1 given the preference ordering at D_1. If decision node D_2 is reachable

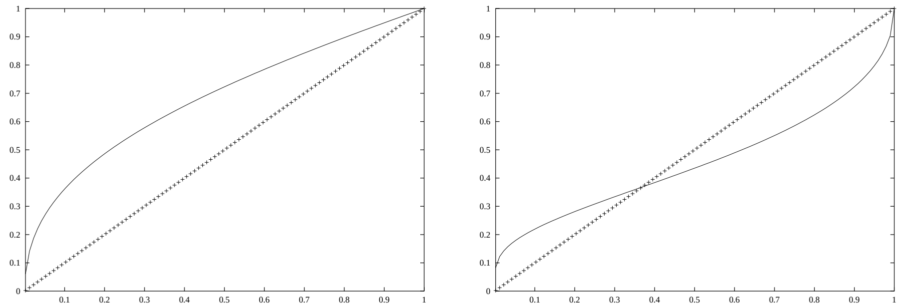

Figure 9.3: The probability transformation functions $q(p) = \frac{p^{0.5}}{(p^{0.5}+(1-p)^{0.5})^{0.5}}$ and $q(p) = \exp(-(-\ln(p))^{0.5}))$ are depicted in Figure (a) and (b), respectively.

from D_1 with Δ_1, then substrategy Δ_2, the restriction of Δ_1 to the subtree rooted at D_2, should itself be optimal for the preference ordering at that node.

A decision maker who is an expected utility maximizer at each decision node (always with the same vNM utility function) is automatically dynamically consistent. This is a straightforward implication of the independence axiom. Consider now a decision maker who is a rank dependent utility maximizer at each decision node (always with the same vNM utility function and probability weighting function). The following example shows that this decision maker can be dynamically inconsistent; this problem was initially addressed by Schick (1986).

Example 9.2.1. Consider the decision tree depicted in Figure 9.4 and assume that the decision maker's preferences are expressed by a rank dependent utility preference function, where the probability weighting function is given by:

$$q(p) = \exp(-(-\ln(p))^{0.5})$$

At D_1 the possible strategies are $\{(D_1, A)\}$, $\{(D_1, B), (D_2, C)\}$ and $\{(D_1, B), (D_2, E)\}$, where:

$$\begin{aligned}
V(\{(D_1, A)\}) &= 2 + (5-2) \cdot q(0.48 + 0.25) + (30-5) \cdot q(0.25) &= 11.41 \\
V(\{(D_1, B), (D_2, E)\}) &= 2 + (5-2) \cdot q(0.5 + 0.25) + (30-5) \cdot q(0.25) &= 11.46 \\
V(\{(D_1, B), (D_2, C)\}) &= 5 + (10-5) \cdot q(0.25 + 0.25) + (20-10) \cdot q(0.25) \\
&= 10.26
\end{aligned}$$

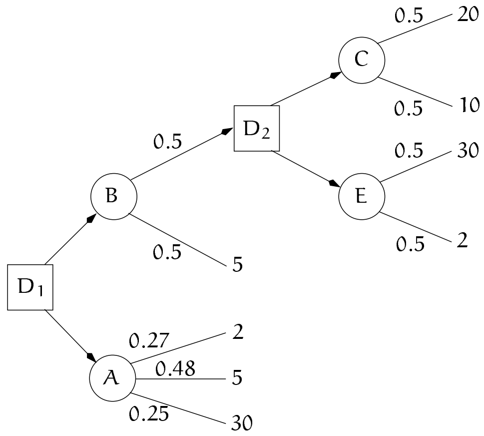

Figure 9.4: A decision tree representation of a dynamic decision problem with two decisions. The values associated with the leaf nodes correspond to the utilities of these consequences.

Thus, at D_1 the decision maker would prefer $\{(D_1, B), (D_2, E)\}$ over $\{(D_1, B), (D_2, C)\}$ and $\{(D_1, A)\}$ however, at D_2 we have:

$$\begin{aligned} V(\{(D_2, C)\}) &= 10 + (20 - 10) \cdot q(0.5) = 14.35 \\ V(\{(D_2, E)\}) &= 2 + (30 - 2) \cdot q(0.5) \quad = 14.18 \end{aligned}$$

That is, at D_2 the decision maker would prefer $\{(D_2, C)\}$ over $\{(D_2, E)\}$ meaning that the decision maker is not dynamically consistent. □

A dilemma

Dynamic consistency is a relation between preference orderings. Let us now examine its implications concerning choice. At any given decision node, the decision maker ranks complete substrategies but can only control the immediate decision. It is generally admitted that, aware of that fact, an intelligent decision maker should at each moment anticipate his future choices and therefore value each immediate decision by the value of the lottery generated by the anticipated ensuing strategy. This is known as *sophisticated behavior*.

From the preceding example, we see that a sophisticated behavior makes a dynamically inconsistent rank dependent utility maximizer choose a strategy which is stochastically dominated at the first order. This is generally considered as irrational.

Example 9.2.2. For the decision problem illustrated in Figure 9.4, the sophisticated strategy is $\{(D_1, A)\}$ which is stochastically dominated by $\{(D_1, B), (D_2, E)\}$ at the first order. This can easily be seen by considering the corresponding cumulative distribution functions. □

Such an irrational choice cannot be observed with an expected utility maximizer. Consider then a dynamically consistent decision maker whose preferences at the root node are expressible by a rank dependent utility criterion. What about his preferences at other decision nodes? Since the optimal strategy depends on the whole tree, so does its restriction to any subtree. Thus preferences at a decision node depend on elements outside the subtree rooted at that node. The same example as above shows it immediately.

Example 9.2.3. Assume that the decision maker is dynamically consistent, i.e., at D_1 the strategy $\{(D_1, B), (D_2, E)\}$ is preferred, which implies that $\{(D_2, E)\}$ is preferred at D_2. However, if the utility 5, in the subtree defined by (D_1, B), is replaced by the utility 9.5 we get:

$$\begin{aligned} V(\{(D_1, B), (D_2, C)\}) &= 9.5 + (10 - 9.5) \cdot q(0.25 + 0.25) + (20 - 10) \cdot q(0.25) \\ &= 12.80 \\ V(\{(D_1, B), (D_2, E)\}) &= 2 + (9.5 - 2) \cdot q(0.5 + 0.25) + (30 - 9.5) \cdot q(0.25) \\ &= 12.70 \end{aligned} \tag{9.9}$$

That is, in this modified decision problem the decision maker would prefer $\{(D_1, B), (D_2, C)\}$ at D_1 and therefore $\{(D_2, C)\}$ at D_2. Hence, the decision maker's choice at D_2 depends on a value outside the subtree having D_2 as root. □

Consequentialism precisely forbids such an influence. Thus the adaptation of rank dependent utility theory to a dynamic setting forces us to face the following dilemma: either accept possibly irrational behavior (choice of a dominated strategy) or renounce consequentialism.

Machina (1989) argues that it is inappropriate to impose consequentialism on non-expected utility behavior since the very notion of a non-expected utility criterion implies a type of non-separability which contradicts the consequentialist assumption. Machina (1989) also provides some examples of situations where genuine non-consequentialist preferences seem extremely natural. Non-consequentialism can also appear through the modification of originally consequentialist preferences, see (McClennen 1990). We shall develop and advocate this alternative approach.

Resolute choice

Resolute choice is a type of behavior introduced and discussed in (McClennen

1990): The resolute decision maker initially commits himself to a strategy, i.e., initially chooses a strategy and never deviates from it later. Conceptually, in this model each decision node is associated with a *Self*, who represents the decision maker at the time and state when the decision is made. The Selves, who primitively may have preference inconsistencies, manage to bend them and achieve a consensus. A restrictive interpretation of resolute choice is that the strategy to which the decision maker commits himself is necessarily the strategy judged optimal, from the very beginning, by the first Self. It has the advantage to straightforwardly endow the resolute strategy with all the qualities ensured by the first Self's criterion.

However, the ability of the decision maker to commit himself, and especially to commit himself to this particular strategy, is a disputable matter. This makes it desirable to explicitly model the underlying process that leads to the determination of a consensual strategy (or fails to reach a consensus); notice that the model, being intended to be prescriptive rather than descriptive, does not necessarily mimic the underlying psychological process of the decision maker.

With this objective, Jaffray (1999) elaborates on resolute choice and suggests a model which can be interpreted as follows: Each Self is associated with a decision node and endowed with a consequentialist preference ordering on the substrategies at that node. Together, these preferences make them dynamically inconsistent. However, the Selves are willing to cooperate. This assumption is quite the reverse of the prevailing one, which is to consider the Selves as noncooperative players of a game (Karni and Safra 1989). As a justification, it can be put forward that the Selves are all part of the same mind and cannot hide intentions from each other; this rules out bluff and betrayal and should therefore facilitate cooperation. The overall goal of the Selves is assumed to be the selection of a non-dominated strategy. For the Self associated with a decision node D, the selection of a non-dominated substrategy in the subtree rooted at D is an obvious goal. For the coalition of all the Selves associated with the immediate decision successors for D, one can argue that it is aware that one of the subcoalitions will in fact be selected by the decision option chosen at D, and therefore wants to avoid the selection of a subcoalition which offers a collectively disadvantageous prospect, i.e., a dominated substrategy. This common goal is the mutually incentive for cooperation between a Self and its immediate successors.

Finally, the willingness of the Selves to cooperate is characterized by the maximum amount of value they are ready to give up, when deviating from their best achievable choices in order to reach a satisfactory consensus. The fact that the Selves are clones makes it possible to assume a common value scale and a common acceptable deviation.

9.3 Investigating strategies

A crucial aspect of the model by Jaffray (1999) is the identification (and subsequent proposal) of a non-dominated strategy that is acceptable to all the Selves. This means that any proposed strategy Δ (represented by a lottery L_Δ) must satisfy:

$$\begin{aligned}\forall \Delta' \neq \Delta, \text{ either } \exists u \in \mathcal{V} \text{ s.t. } EU(L_\Delta)_u > EU(L_{\Delta'})_u \\ \text{or } \forall u \in \mathcal{V} : EU(L_\Delta)_u \geq EU(L_{\Delta'})_u,\end{aligned}$$

where $\mathcal{V}$ is defined as in Theorem 9.2.1. If both of the expressions hold for all $\Delta' \neq \Delta$, then Δ stochastically dominates all other strategies at the first order.

In order to investigate whether or not a strategy Δ satisfies one of the expressions above, we would in principle have to compare Δ with all the other strategies. To overcome this computational difficulty we shall instead look for strategies for which there exists a utility function s.t. the strategy in question is strictly EU-optimal:

$$\exists u \in \mathcal{V} \text{ s.t. } EU(L_\Delta)_u > EU(L_{\Delta'})_u, \forall \Delta' \neq \Delta \tag{9.10}$$

To say it in another way, if there exists a utility function s.t. Δ is an EU-optimal strategy then Δ is non-dominated. The converse is, however, not necessarily true, i.e., the approach is sound but not complete since we cannot infer that Δ is dominated if no such utility function exists.

9.3.1 Restating the problem

In this section we reduce the problem of determining whether a strategy Δ is non-dominated to the problem of finding a solution to a particular system of linear inequalities, i.e., the system of inequalities has a solution if and only if there exists a utility function resulting in Δ being a strictly optimal strategy (see Expression 9.10).

The linear inequalities are found by initially creating the variables which are subject to a constraint. Given a decision tree T these variables are constructed as follows:

- Create a variable y_c for each distinct consequence c in T .
- Create a variable y_D for each decision node D in T.

From these variables the system of inequalities is constructed incrementally by considering the consequences and the decision nodes separately.

The constraints on the consequences correspond to their preference ordering which can be read directly from the (original) utility function. That is, by assuming that the preference ordering over the consequences is given by their indexes, we require:

$$\forall 2 \leq i \leq m : y_{c_i} \geq y_{c_{i-1}} + \alpha, \tag{9.11}$$

where m is the number of distinct consequences and $\alpha > 0$; notice that two consequences c_i and c_j are considered distinct (similar) if and only if $c_i \prec c_j$ or $c_j \prec c_i$ ($c_i \sim c_j$). This set of inequalities ensures that any solution encodes a utility function $u \in \mathcal{U}$, defined by $u(c_i) = y_{c_i}$, which respects the preference ordering over all the consequences.

The remaining constraints are associated with the decision nodes and can be identified (locally) in the decision tree.

Assume that D is a decision node and let $\epsilon > 0$. Then, for each arc (D, X) which is not part of Δ, we require, if X is a chance node:

$$y_D \geq \sum_{l \in \mathcal{L}} p_l y_l + \epsilon, \tag{9.12}$$

where $\mathcal{L}$ is the set of all paths from X to either its immediate decision successors or to value nodes; p_l is the product of the probabilities associated with the path l and y_l is the variable associated with the "end" node of path l. In case X is a decision node or a value node, we simply have $y_D \geq y_X + \epsilon$. The constant $\epsilon > 0$ ensures strict optimality; note that α and ϵ need not be equal.

If (D, X) is part of the strategy Δ and if X is a chance node, we require:

$$y_D \leq \sum_{l \in \mathcal{L}} p_l y_l. \tag{9.13}$$

If X is a decision node or a value node we have $y_D \leq y_X$.

Example 9.3.1. Consider the case where we have to choose between the two lotteries $L_1 = (c_2, 0.4; c_3, 0.2; c_4, 0.4)$ and $L_2 = (c_1, 0.7; c_5, 0.1, c_6, 0.2)$, and assume that the preferences over the consequences is revealed by their index (see Figure 9.5).

Consider the strategy $\Delta = \{(D, L_2)\}$, and assume that we want to determine whether there exists a utility function that respects the preference ordering over the consequences and yields Δ as an optimal strategy.

For e.g. c_4 we construct the inequality (the inequalities for the other consequences are constructed similarly):

$$y_{c_4} \geq y_{c_3} + \alpha.$$

For the decision node D we have:

$$\begin{aligned} y_D &\geq 0.2 \cdot y_{c_3} + 0.4 \cdot y_{c_2} + 0.4 \cdot y_{c_4} + \epsilon \\ y_D &\leq 0.7 \cdot y_{c_1} + 0.1 \cdot y_{c_5} + 0.2 \cdot y_{c_6} \end{aligned}$$

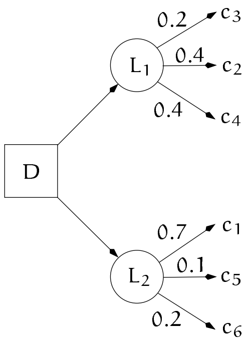

Figure 9.5: A decision tree with one decision node representing the choice between the two lotteries L_1 and L_2; the preference ordering over the consequences correspond to their index.

For $\alpha = 1$ and $\epsilon = 0.1$, an admissible solution to these inequalities encode the utility function $u(c_1) = 0$, $u(c_2) = 1$, $u(c_3) = 2$, $u(c_4) = 3$, $u(c_5) = 4$ and $u(c_6) = 8.5$. This utility function yields $\Delta = \{(D, L_2)\}$ as an optimal strategy ($EU(L_1)_u = 2$ and $EU(L_2)_u = 2.1$). □

Theorem 9.3.1. Let T be a decision tree with consequences $\mathcal{C}$ and let Δ be a strategy in T. Then there exists a utility function $u \in \mathcal{V}$ s.t. Δ is a strictly optimal strategy if and only if the corresponding system of linear inequalities has a solution for some $\alpha > 0$ and $\epsilon > 0$.

Proof. Follows from the construction of the set of inequalities. □

Theorem 9.3.2. Given a decision tree T and a strategy Δ, if the corresponding system of linear inequalities has a solution for some $\alpha > 0$ and $\epsilon > 0$, then the system of linear inequalities has a solution $\forall \alpha' > 0$ and $\forall \epsilon' > 0$.

Proof. The proof is straightforward by observing that a utility function is unique up to an affine transformation, and that the expected utility is linear in the utilities. □

Finding a solution for a system of inequalities can be done with "brute force" by considering all the inequalities simultaneously and then use stage one of the *simplex algorithm*; similarly, we may construct an artificial *objective function*, e.g. $c = 0$, and use an *interior point* algorithm. Notice that the structure of the inequalities does not directly support decomposition schemes like (Dantzig and Wolfe 1960), however, it should be noted that due to the relatively large size of manageable systems the brute force approach might be sufficient for some decision problems.

Alternatively to the brute force approach we can exploit the structure of the decision tree e.g. a solution may be found by recursively solving collections of smaller subsets of inequalities. In Appendix A we give a suggestion for how the simplex algorithm can be modified to exploit such a property.

9.4 Evaluation

In this section we present an algorithm for solving decision problems w.r.t. non-expected utility criteria such as rank dependent utility. The underlying idea of the algorithm is to assume consequentialist preferences and to accept non-consequentialist behavior as described in Section 9.2.1.

The consequentialist preference assumption is reflected in the use of backward induction, whereas the assumption about non-consequentialist behavior follows from a Self being willing to accept any of a set of strategies (whereas one strategy is enough when dealing with expected utility); the actual selection is made by a preceding Self and therefore depends on data outside the consequentially relevant subtree.

9.4.1 Algorithm

The algorithm works itself backwards from the leaves towards the root of the decision tree T. When a decision node (say D) is visited, a set of candidate (sub)-strategies Δ_{D}^{+} in $\mathsf{T}(\mathsf{D})$ is determined; since we work from the leaves towards the root, Δ_{D}^{+} is formed by taking the Cartesian product of the strategies acceptable by the Selves that are associated with the immediate decision successors of D in T. For each strategy in Δ_{D}^{+} we then determine whether or not it passes the non-dominance test; let $\Delta_{\mathsf{D}} \subseteq \Delta_{\mathsf{D}}^{+}$ be the set of strategies which are proven non-dominated. For each strategy Δ in Δ_{D} we calculate $\mathsf{V}(\Delta)$ and finally we remove all strategies with a value inferior to the best achievable one minus the amount (denoted θ) the Self is willing to give up in order to achieve the goal of identifying a non-dominated strategy.

Obviously, if the value of θ is "sufficiently large" then no non-dominated strategies are removed, and at each decision node we may therefore get a set of strategies that is intractable to evaluate. In order to overcome this computational difficulty we impose an upper bound (denoted κ) on the number of non-dominated strategies acceptable by a Self. That is, if each Self is willing to accept at most κ strategies, then at any decision node D we only need to evaluate at most $\kappa^{\omega}|\mathrm{sp}(\mathsf{D})|$ strategies, where ω is the maximum number of decision nodes that immediately succeed D w.r.t. some decision option for D. Notice, that κ is only introduced to ensure the computational feasibility (as opposed to θ which is part of the underlying model).

Finally, it should be emphasized that for each strategy in Δ_D^+ we make sure that it is non-dominated before its value is calculated. This reflects the overall goal of finding a non-dominated strategy that is acceptable by all the Selves, i.e., a dominated (sub-)strategy should have no effect on any acceptable strategy; otherwise a dominated strategy might force a non-dominated strategy to be rejected. Moreover, this also allows an "early" identification of strategies which do not pass the dominance test, as stated by the following theorem.

Theorem 9.4.1. Let T be a decision tree and let D be a decision node in T. If there does not exist a utility function $u \in \mathcal{V}$ s.t. Δ is a unique EU-optimal strategy w.r.t. u in $T(D)$, then for any strategy Δ' that induces Δ there does not exist a utility function $u' \in \mathcal{V}$ s.t. Δ' is a unique EU-optimal strategy w.r.t. u'.

Proof. Follows immediately. □

Before presenting the algorithm we need the following definition.

Definition 9.4.1. Let $T(D)$ be a decision tree with root D, and let $\mathcal{D}_{(D,X)}$ be the set of immediate decision successors for D w.r.t. the decision option $X \in sp(D)$. The set of *candidate strategies* for D is then given by:

$$\Delta_D^+ = \bigcup_{X \in sp(D)^*} \left\{ \{(D,X)\} \cup S \mid S \in \prod_{D' \in \mathcal{D}_{(D,X)}} \Delta_{D'} \right\},$$

where $sp(D)^* \subseteq sp(D)$ s.t. for all $X \in sp(D)^*$ it holds that every $D' \in \mathcal{D}_{(D,X)}$ is associated with a non-empty set of strategies.

The algorithm outlined above is formalized as follows.

Algorithm 9.4.1. Let θ denote a Self's willingness to cooperate, and let κ be the maximum number of strategies acceptable by a Self. To evaluate a decision tree T with root D, do:

1. Initialize T by associating the probability 1 with each consequence (leaf node) of T.
2. Invoke Algorithm 9.4.2 on T w.r.t. θ and κ and let Δ_D be the set of strategies returned.

3. **If** $\Delta_D \neq \emptyset$ **then**
   Return $\arg\max_{\Delta \in \Delta_D} V(\Delta)$.
   else
   No solution exists for T w.r.t. κ and θ.

□

Algorithm 9.4.2. Given a decision tree T, let θ denote a Self's willingness to cooperate, and let κ denote the maximum number of strategies acceptable by a Self. To determine the strategies acceptable by decision node D, do:

1. Invoke Algorithm 9.4.2 on D', $\forall D' \in \mathcal{D}_D$.
2. Let Δ_D^+ be the set of candidate strategies for D in $T(D)$ (see Definition 9.4.1).
3. **For** each chance node $A \in \mathcal{C}_D$
   For each $B \in sp(A)$
   Multiply $P((A, B)|past(A))$ with the probabilities associated with the consequences in $T(B)$.
4. **For** each strategy $\Delta \in \Delta_D^+$
   If Δ is non-dominated in $T(D)$ (see Section 9.3) **then**
   a) Calculate $V(\Delta)$.
   b) Set $\Delta_D := \Delta_D \cup \{\Delta\}$ and let $V_D^{max} = \max_{\Delta \in \Delta_D} V(\Delta)$.
5. **For** each strategy $\Delta \in \Delta_D$
   If $V(\Delta) < V_D^{max} - \theta$ **then**
   Set $\Delta_D := \Delta_D \backslash \{\Delta\}$
6. **If** $|\Delta_D| > \kappa$ **then**
   Repeat
   Set $\Delta_D := \Delta_D \backslash \{\Delta\}$, where $\Delta = \arg\min_{\Delta \in \Delta_D} V(\Delta)$.
   Until $|\Delta_D| = \kappa$
7. Return Δ_D as the set of strategies acceptable by D.

□

During the evaluation the elements in Δ_D should, for reasons of efficiency, be organized in a data structure which is ordered (or accessible) w.r.t. $V(\Delta)$. Furthermore, it should be noticed that step (3) ensures that all probabilities needed to calculate $V(\Delta)$ in step 4a are available. Hence, to calculate $V(\Delta)$ we basically only need to order the possible consequences of Δ w.r.t. their preference ordering and to calculate the cumulative distribution over these consequences; the latter can be done in linear time given the ordering over the consequences.

9.4.2 Complexity

In what follows we consider the complexity of the proposed algorithm under the assumption that we have a decision tree T with a set $\mathcal{C}$ of distinct consequences.[3] Since the model is proposed as an alternative to expected utility theory, it is appropriate to consider the relative complexity. That is:

$$\text{Relative complexity} = \frac{\mathsf{O}(\text{non-EU})}{\mathsf{O}(\text{EU})}$$

In both Algorithm 9.4.1 and the *average-out and fold-back* algorithm by (Raiffa and Schlaifer 1961), we work ourselves backwards from the leaves towards the root. When a chance node A is visited in the:

- expected utility model, it is assigned the sum of the values associated with its children weighted by the probabilities of their outcomes.
- non-expected utility model, the probability $\mathsf{P}((\mathsf{A},\mathsf{X})|\mathsf{past}(\mathsf{A}))$ is multiplied with the probability (initially 1) associated with the consequences in the subtree $\mathsf{T}(\mathsf{X})$, for each $\mathsf{X} \in \mathsf{sp}(\mathsf{A})$.

When a decision node D is visited in the:

- expected utility model, it is associated with the decision option leading to the child having the highest expected utility; the decision node is assigned this value as well.
- non-expected utility model, it is associated with a set of substrategies. This set is formed by i) considering the set of substrategies associated with each of its immediate decision successors, and ii) iteratively removing substrategies s.t. the resulting set has cardinality no larger than κ, and consists of the non-dominated substrategies with highest values.

Compared to the *average-out and fold-back* algorithm, the most computationally intensive step is when determining whether or not a strategy is non-dominated: When visiting a decision node D we consider the decision tree $\mathsf{T}(\mathsf{D})$ rooted at D; obviously, if some decision node D' in $\mathsf{T}(\mathsf{D})$ is not associated with any substrategies then we need not consider the part of $\mathsf{T}(\mathsf{D})$ corresponding to the subtree $\mathsf{T}(\mathsf{D}')$. For each of the candidate substrategies at D we construct a system of linear inequalities. Given that $\mathsf{T}(\mathsf{D})$ contains the decision nodes $\mathcal{U}_{\mathrm{D}}$ we obtain a system of inequalities where the:

[3] Note that the algorithm is also applicable in case we work with the coalesced decision tree (Olmsted 1983). Constructing the system of linear inequalities from the coalesced decision tree would significantly reduce the number of linear inequalities; however, we shall restrict our attention to the traditional decision tree.

- Number of variables $= |\mathcal{U}_{\mathrm{D}}| + |\mathcal{C}|$.
- Number of inequalities $\leq |\mathcal{U}_{\mathrm{D}}| \cdot \mathrm{sp}_{\max} + |\mathcal{C}|$; $\mathrm{sp}_{\max}$ is the maximum number of children for a decision node.

A solution to these inequalities can be found by e.g. employing a simplex algorithm or an interior point algorithm. Although the simplex algorithm can require an exponential running time for some difficult classes of linear problems, the running time has been shown to be polynomial on the average; this also agrees with the complexity observed in practice (see (Schrijver 1986)). On the other hand, the interior point algorithm is proven to have a polynomial running time (Karmarkar 1984).

Now, if $\#_{\mathsf{T}(\mathsf{D})}$ denotes the time for performing the non-dominance test in the subtree $\mathsf{T}(\mathsf{D})$ and $\left|\mathcal{C}^{\mathsf{T}(\mathsf{X})}\right|$ is the maximum number of possible consequences for a strategy in $\mathsf{T}(\mathsf{X})$, then we have:

$$\text{Relative complexity} = \frac{\mathsf{O}\left(\sum_{\mathsf{D}\in\mathcal{U}_{\mathsf{D}}}\left[\kappa^{\omega}\cdot|\mathrm{sp}(\mathsf{D})|\cdot\left(\#_{\mathsf{T}(\mathsf{D})}+\left|\mathcal{C}^{\mathsf{T}(\mathsf{D})}\right|\right)\right]+\sum_{\mathsf{C}\in\mathcal{U}_{\mathsf{C}}}\left|\mathcal{C}^{\mathsf{T}(\mathsf{C})}\right|\right)}{\mathsf{O}\left(\sum_{\mathsf{D}\in\mathcal{U}_{\mathsf{D}}}|\mathrm{sp}(\mathsf{D})|+\sum_{\mathsf{C}\in\mathcal{U}_{\mathsf{C}}}|\mathrm{sp}(\mathsf{C})|\right)}. \tag{9.17}$$

It is important to notice that the complexity of investigating a strategy is strongly dependent on the number of distinct consequences in the decision tree. In real world decision problems it is reasonable to assume that the number of distinct consequences is significantly smaller than the number of possible consequences in the decision tree.

9.5 Empirical results

Two sets of tests have been performed w.r.t. the evaluation of decision trees based on non-expected utility. In the first set of tests, we have determined the average number of non-dominated strategies produced by rolling back the decision tree according to the sophisticated behavior of a rank dependent utility maximizer. In the second set of tests, the algorithm proposed in Section 9.4 was used to identify an average number for κ (the maximum number of strategies acceptable by a Self) needed to produce a non-dominated strategy; the program *PCx* was used to solve the inequalities.[4] It should be noted that considerations regarding the value of θ have been omitted from the tests, since this value should be elicited

[4] *PCx* is an interior-point linear programming package which can be downloaded at *http://www-fp.mcs.anl.gov/otc/Tools/PCx/index.html*.

in relation to the decision maker and is therefore specific to the decision problem (and the decision maker) in question.

The first set of tests was performed on a collection of randomly generated decision trees with a branching factor between 2 and 4 (randomly chosen) and where the number of decision nodes and chance nodes varied from 15 to 7200. For a given number of nodes, 50 different decision trees were generated and for each of these decision trees a sophisticated strategy was determined w.r.t. three different utility functions; the range of these utility functions was either $\{0, 1, \ldots, 20\}$, $\{0, 1, \ldots, 100\}$ and $\{0, 1, \ldots, 1000\}$, i.e., each possible consequence was randomly assigned an integer utility level within these ranges. The average number of non-dominated sophisticated strategies is illustrated in Figures 9.6-9.8, which supports the prevailing assumption that a sophisticated strategy is likely to be dominated. This result also justifies the use of different models and evaluation schemes when considering dynamic decision problems with non-separable preference functions.

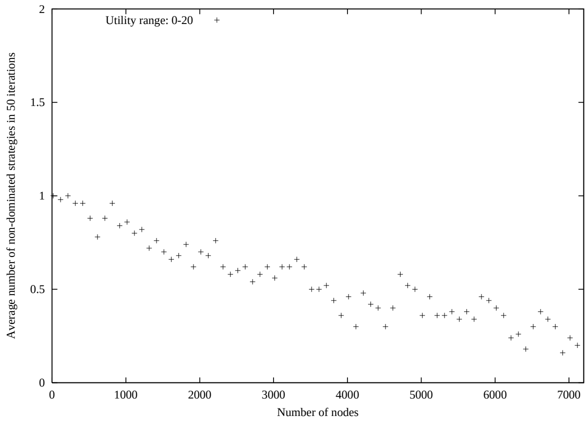

Figure 9.6: The average number of non-dominated sophisticated strategies as a function of the number of chance nodes and decision nodes in the decision tree. The utility levels varied between 0 and 20.

The second set of tests was also performed on a collection of randomly generated decision trees. The number of nodes varied between 15 and 4015, and for a given number of nodes 20 different decision trees were generated. For each of these decision trees the number of strategies acceptable by a Self was incremented (starting with $\kappa = 1$) until a non-dominated strategy was generated. The results of the tests are illustrated in Figures 9.9-9.11.

The second set of tests indicates that the average value for κ (needed to obtain a non-dominated strategy) increases w.r.t. both the size of the decision tree and

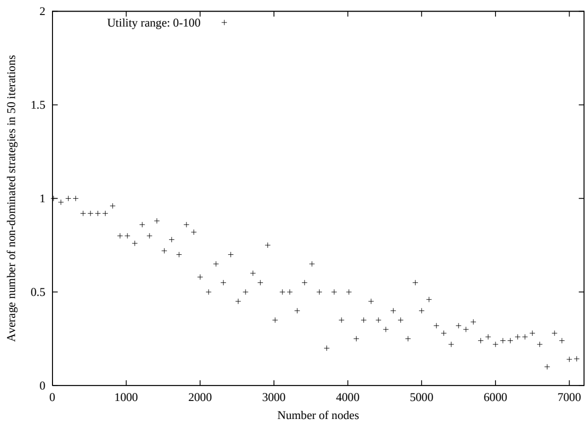

Figure 9.7: The average number of non-dominated sophisticated strategies as a function of the number of chance nodes and decision nodes in the decision tree. The utility levels varied between 0 and 100.

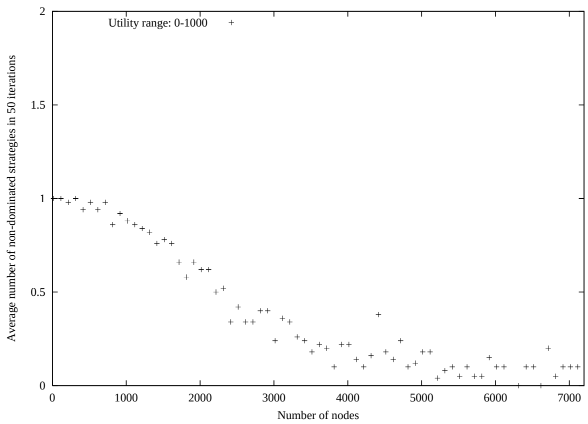

Figure 9.8: The average number of non-dominated sophisticated strategies as a function of the number of chance nodes and decision nodes in the decision tree. The utility levels varied between 0 and 1000.

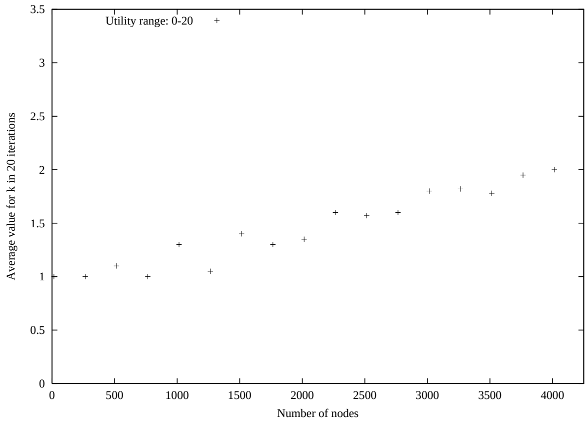

Figure 9.9: The average value of κ which yields a non-dominated strategy as a function of the number of chance nodes and decision nodes in the decision tree. The utility levels varied between 0 and 20.

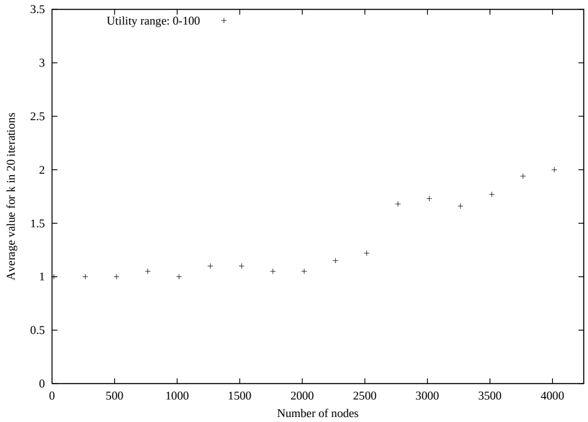

Figure 9.10: The average value of κ which yields a non-dominated strategy as a function of the number of chance nodes and decision nodes in the decision tree. The utility levels varied between 0 and 100.

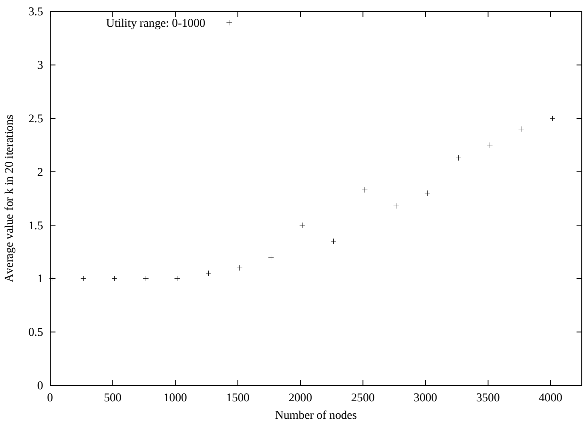

Figure 9.11: The average value of κ which yields a non-dominated strategy as a function of the number of chance nodes and decision nodes in the decision tree. The utility levels varied between 0 and 1000.

the range of the utility function.

9.6 Other aspects

9.6.1 Backtracking

If no non-dominated strategy is found w.r.t. some values for κ and θ we would in principle need to perform the entire evaluation again, with other (larger) values for κ. However, by employing a form of *backtracking* we can re-use many of the calculations initially performed.

Consider a decision tree T with a root decision node D and assume that no non-dominated strategy was produced by the initial evaluation; from the empirical findings above, this problem tends to arise if the value of κ is chosen too "small". An obvious way to accommodate such situations (enlarging the set of candidate strategies for D without evaluating the entire decision tree again) is to use backtracking. This can be realized by storing all (or some) of the sets of substrategies acceptable by the decision successors for D. Given these sets of acceptable substrategies, a new set of candidate strategies for D can easily be generated. For instance, if we just consider the immediate decision successors for D, then the number of candidate strategies which can be generated is given by $sp_{\max} \cdot (sp_{\max} \cdot \kappa^{\omega})^{\omega}$; thus, the size of the initial set of candidate strategies can be enlarged with:

$$sp_{\max} \cdot (sp_{\max} \cdot \kappa^{\omega})^{\omega} - (sp_{\max} \cdot \kappa^{\omega}).$$

Obviously the backtracking procedure can be extended (to create additional candidate strategies), by considering several "steps" back in the decision tree. However, it should be noticed that, in general, backtracking does not come free of charge as the computational complexity increases with the value of κ, or more precisely, with the number of candidate strategies associated with the decision nodes. Note also, that backtracking supports an initial evaluation of the decision tree with an optimistic value for κ, i.e., the value assigned to κ can be guided by the empirical findings given in Section 9.5.

9.6.2 Testing for stochastic dominance of the second order

The investigation of strategies can be modified for investigating strategies w.r.t. stochastic dominance of the second order (see Definition 9.2.3). In this context, any proposed strategy Δ (represented by a lottery L_Δ) must satisfy:

$$\begin{aligned} \forall \Delta' \neq \Delta, \text{ either } & \exists u \in \mathcal{V} \text{ s.t. } EU(L_\Delta)_u > EU(L_{\Delta'})_u \\ \text{or } & \forall u \in \mathcal{V} \text{ s.t. } EU(L_\Delta)_u \geq EU(L_{\Delta'})_u, \end{aligned}$$

where $\mathcal{V} = \{u | [u \text{ is concave}] \text{ and } [\text{if } c_i \prec c_j \text{ then } u(c_i) < u(c_j)]\}$. As for the investigation of stochastic dominance of the first order, this approach would in principle require the comparison of Δ with all other strategies. So we advocate the same approach as for first order stochastic dominance; that is, we look for the strategies for which there exists a utility function s.t. the strategy in question is strictly EU-optimal:

$$\exists u \in \mathcal{V} \text{ s.t. } EU(L_\Delta)_u > EU(L_{\Delta'})_u, \forall \Delta' \neq \Delta.$$

Similarly to first order stochastic dominance, this problem can be restated in terms of a linear programming problem however, we also need to encode that any feasible solution must correspond to a concave utility function. Consider the three consequences c_{i-1}, c_i and c_{i+1}, where $c_{i-1} \prec c_i \prec c_{i+1}$. Any candidate utility function must satisfy:

$$u(c_{i+1}) - u(c_i) \leq [u(c_i) - u(c_{i-1})] \left(\frac{c_{i+1} - c_i}{c_i - c_{i-1}} \right). \tag{9.20}$$

By extending the system of linear inequalities (as defined in Section 9.3) with Inequality 9.20 ($\forall 2 \leq i \leq n-1$) we can investigate strategies w.r.t. to this stricter notion of second order stochastic dominance.

9.7 Discussion

In this chapter we have presented an operational approach to decision aiding based on non-expected utility. The approach was motivated by the fact that decision makers sometimes reject proposed strategies (or models) based on expected utility theory, due to the inability of expected utility theory to account for experimentally well-established certainty/possibility effects. Methods have been proposed to circumvent these shortcomings. For instance, the decision maker could pursue the second best EU-strategy in case the optimal EU-strategy is rejected. However, refusing to undertake the strategy with maximum expected utility implies a violation of the maximum expected utility principle. As this assumption also underlies the computation of the second best strategy, there is no immediate justification for proposing this strategy since it is identified under an assumption which the decision maker is known to violate. Finally, the second best strategy may in fact be dominated, and proposing a dominated strategy is usually considered irrational and therefore unacceptable from a normative point of view.

The algorithm proposed in this chapter is based on the model by Jaffray (1999). The non-expected utility criterion attributed to the decision maker is rank dependent utility (Quiggin 1982), and the common goal among the Selves is to produce a non-dominated strategy. The rank dependent utility criterion is non-separable, meaning that, in general, it fails to support backward induction. However, under the assumption that the decision maker exhibit consequentialist preferences and non-consequentialist behavior we have shown that backwards induction can be facilitated.

A crucial aspect of the proposed algorithm is the identification of non-dominated strategies and, in particular, strategies which are stochastically non-dominated in either the first or second order. We have proposed a method for identifying such strategies by reducing the original problem to the problem of finding a solution to a linear program. Moreover, we have suggested a method for reducing the computational complexity for solving the linear program by exploiting the structure of the decision tree from which the linear program is generated, see Appendix A.

The proposed algorithm was tested on a collection of randomly generated decision trees, and it was furthermore shown (by empirical results) that while simple sophisticated behavior (under a non-separable decision criterion) in general fails to produce a non-dominated strategy, our algorithm manages to exhibit one for reasonable values of the model parameters.

Chapter 10

Conclusion

In this dissertation we have considered the representation, solution and analysis of decision problems with finite horizon.

When representing a decision problem using the influence diagram framework, a linear temporal ordering over the decision nodes is required; this requirement ensures that the influence diagram is well-defined. However, when some of the decisions are conditionally independent it is unnecessary to impose such an ordering. In Chapter 3 we described the framework of partial influence diagrams, i.e., influence diagrams with only a partial temporal ordering over the decision nodes. We presented a set of conditions that are necessary and sufficient for a partial influence diagram to be well-defined; these conditions also revealed an operational characterization of the observed variables which are relevant for a decision variable (see also (Shachter 1999)). We proposed an algorithm for determining whether or not a partial influence diagram is well-defined and, given that it is not well-defined, the algorithm indicates where further refinement of the temporal ordering is needed.

Jensen (2001) takes another interesting approach towards partial influence diagrams: A partial influence diagram represents a decision problem hence, when solving a partial influence diagram we do not only calculate an optimal strategy but also an optimal ordering of the decisions. This view is in fact compatible with the ideas presented in Chapter 3; given an ambiguous partial influence diagram we can resolve the ambiguities by inserting fill-ins s.t. the expected utility of the model is maximized.

In Chapter 4 we presented the asymmetric influence diagram framework for representing asymmetric decision problems. The asymmetric influence diagram is based on the partial influence diagram, and augments the graphical structure with labels associated with nodes and informational arcs. These labels encode structural asymmetry as they indicate under which conditions the associated nodes and informational arcs are part of the decision problem. Hence, information about structural asymmetry can be read directly from the model. Functional asymmetry is encoded by partial probability potentials and utility functions, as well as restrictive functions associated with the decision variables. Thus, the modeler can specify that the possible outcomes of a chance variable vary depending on the conditioning states, and that the legitimate decision options for a decision variable depends on the different information states.

As opposed to traditional frameworks for representing asymmetric decision problems (see e.g. (Smith et al. 1993; Covaliu and Oliver 1995; Shenoy 2000)), the asymmetric influence diagram allows (embedded) cycles in the model. This supports the specification of decision problems in which the relative temporal order of two decision variables depend on previous observations and decisions; the use of cycles has been further pursued by Demirer and Shenoy (2001). A topic for future research is the characterization of the class of asymmetric decision problems that can efficiently be represented in the asymmetric influence diagram framework, i.e., the class of decision problems that can be modeled without introducing any redundancy such as artificial states or duplicated variables.

The evaluation of asymmetric decision problems, formulated in the asymmetric influence diagram framework, were described in Chapter 6. The evaluation is based on a decomposition of the original decision problem into a collection of smaller symmetric subproblems. These subproblems can be solved using existing evaluation methods, and a solution to the original decision problem is found by combining the solutions for the subproblems. The evaluation of the symmetric subproblems was described by advocating the principles of the LAZY evaluation architecture (Madsen and Jensen 1999), where the main idea is to postpone the combination of potentials until needed. This idea may actually be taken one step further: The key observation is that during the evaluation we are only interested in the optimal policy for each of the decision variables hence, probability potentials need not be combined when a chance variable should be marginalized out. Instead we can store the probability potentials in *scripts* s.t. a script is only evaluated when a decision variable should be eliminated (thereby extending the use of on-line triangulation). The use of scripts has the potential benefit that more situations may arise where we can exploit the occurrence of e.g. barren variables; this is a topic for future research.

The evaluation of asymmetric influence diagrams also formed the basis for the definition of a well-defined asymmetric influence diagram. The definition is partly based on the concept of a significant chance variable, adopted from the partial influence diagram framework. The structural characterization of the significant chance variables in a partial influence diagram can, however, not completely be applied as special care should be taken when considering the utility functions being relevant for the so-called split variables. We tentatively indicated how these utility functions should be identified, and a complete formal characterization of these utility functions is a topic for future research.

In Chapter 7 we presented a method for decomposing an influence diagram into a collection of smaller influence diagrams. The decomposition produces an influence diagram for each decision variable in the original decision problem, and each of these smaller influence diagrams contains exactly the variables that are relevant and required for the associated decision variable. From a modeling perspective, this decomposition has the advantages that i) when eliciting the utilities and the probabilities we may focus our attention on the parts of the decision problem that are relevant for a particular decision variable, and ii) we can determine whether some variable provides information which is relevant for a certain decision. Additionally, we indicated that performing such a decomposition may reduce the computational complexity when solving the decision problem, and we showed that the decomposition has the advantage of producing an optimal strategy with no redundant variables.

The proposed decomposition method may also provide a way to simplify the elicitation of the structure of a complex decision problem. That is, it may be possible to characterize conditions that are necessary and sufficient to ensure the consistency of different subproblems based on the existence of a decision problem that decomposes into these subproblems. This is a topic for future research.

In Chapter 8 we addressed sensitivity analysis in influence diagrams. This type of analysis has previously been considered within the medical decision making literature, where the usual approach is to initially assign a probability distribution to each of the uncertain parameters. Based on these probability distributions, we may consider sensitivity indicators such as the probability of obtaining a different optimal strategy (Doubilet et al. 1985) or the expected value of perfect information (Felli and Hazen 1998). However, when the uncertain parameters are not independent (which is generally the case when we have uncertain utility parameters and a preference ordering over the outcomes) we need to assign conditional probability distributions to these parameters. Unfortunately, these conditional distributions can be difficult to elicit, in particular when the utilities have no straightforward interpretation. Moreover, even when the parameters are assumed to be independent the computational complexity can be cumbersome. Hence, Monte Carlo simulations are usually applied to sample values from the distributions associated with the parameters; when the parameters are not independent we will in general need a larger sample set to get a representative sample from the joint distribution. To overcome these problems we presented two methods for doing non-probabilistic sensitivity analysis, i.e., we considered how much a set of parameters can be varied without changing the optimal strategy found w.r.t. the initial values of the parameters. The proposed methods also provide a simple way to take the preference ordering over the outcomes into account, and they are computationally efficient when dealing with uncertain utility parameters. When considering the joint effect from uncertain probability parameters, the computational complexity appears impractical and indicates the use of probabilistic sensitivity analysis; following the standard assumption from the learning community (see e.g. (Cooper and Herskovits 1991)) it seems reasonable to assume that the probability parameters are independent a priori.

In chapter 9 we considered an operational approach to decision making based on non-expected utility theory; specifically, we considered rank dependent utility theory (Quiggin 1982). The motivation for this approach was that a decision maker sometimes reject a proposed strategy based on expected utility theory, because expected utility theory may be unable to simultaneously account for all of the decision maker's choices. In general, non-expected utility theory does not share the separability properties of expected utility theory hence, in sequential decision problems the sophisticated strategy is likely to be dominated. Additionally, since rational non-expected utility behavior is necessarily a non-consequentialist behavior (i.e., the choice in a subtree depends on the rest of the tree) backwards induction is not immediately facilitated. However, we have shown that it is possible to define a method that involves a "rolling back" of the decision tree, and selects a non-dominated strategy. The method, which is based on the model by Jaffray (1999), assumes consequentialist preferences but accepts non-consequentialist behavior. Relative to the average-out and fold-back algorithm, the main computational increase follows from the identification of non-dominated strategies; we reduced this problem to the problem of finding a solution to a linear programming problem. Empirical results confirm the tractability of the proposed method, and they also demonstrate that a sophisticated behavior (under a non-separable decision criterion) in general fails to produce a non-dominated strategy.

Appendix A

Testing for dominance

In this appendix we illustrate how the structure of a linear program, generated by a decision tree according to Section 9.3, can be exploited when testing for dominance.

Before describing this procedure we briefly review stage one of the simplex algorithm based on (Tucker 1964) and (Rothenberg 1979); stage two of the simplex algorithm will not be described as we are only interested in identifying a configuration which satisfies the inequalities.

A.1 The simplex algorithm

In order to apply the *simplex algorithm*, the inequalities must be compiled into a suitable form. This is done by introducing so-called *slack variables* (sometimes referred to a *surplus variables*) to take up the "slack" from the inequalities.

In general, we start with

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \leq b_i$$

and by introducing the slack variable t_i we get

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n + t_i = b_i \tag{A.2}$$

Rewriting Equation A.2 in terms of $-t_i$ gives:

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n - b_i = -t_i \tag{A.3}$$

The set of equalities produced by adding slack variables can be organized in a *tableau*. We shall refer to the tableau consisting of Equation A.3 ($\forall 1 \leq i \leq m$) as the *initial tableau* (see Table A.1).

x_1	x_2	$\cdots$	x_j	$\cdots$	x_n	1	
a_{11}	a_{12}	$\cdots$	a_{1j}	$\cdots$	a_{1n}	$-b_1$	$=-t_1$
a_{21}	a_{22}	$\cdots$	a_{2j}	$\cdots$	a_{2n}	$-b_2$	$=-t_2$
$\vdots$	$\dots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\vdots$
a_{i1}	a_{i2}	$\cdots$	a_{ij}	$\cdots$	a_{in}	$-b_i$	$=-t_i$
$\vdots$	$\dots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\vdots$
a_{m1}	a_{m2}	$\cdots$	a_{mj}	$\cdots$	a_{mn}	$-b_m$	$=-t_m$

Table A.1: The initial tableau.

The aim of stage one is to reach a tableau, where all the $-b_i$'s are non-positive or to identify/produce a row where $-b$ is positive and all the a entries in that row are positive (in the latter case the problem has no solution).

The initial tableau is transformed by a sequence of *pivot transformations*. Let p denote the *pivot entry*, let q denote another entry in the pivot row, let r denote another entry (diffrent from p) in the pivot column and let s denote an entry neither in the same row nor in the same column as p. The pivot transformation is then specified as follows (see Table A.2 for illustration):

1. Replace the pivot p by $1/p$.
2. Multiply the remaining q's in p's row by $1/p$.
3. Multiply the remaining r's in p's column by $-1/p$.
4. Add $-(rq)/p$ to every other entry s.

Notice that if a row has zero in the pivot column then the entries in that row are unchanged by the pivot transformation.

$$\begin{array}{|ccc|}\hline p & \cdots & q \\ \vdots & \ddots & \vdots \\ r & \cdots & s \\ \hline\end{array} \quad \Longrightarrow \quad \begin{array}{|ccc|}\hline \frac{1}{p} & \cdots & \frac{q}{p} \\ \vdots & \ddots & \vdots \\ \frac{-r}{p} & \cdots & s - \frac{rq}{p} \\ \hline\end{array}$$

Table A.2: The table illustrates the pivot transfomation of a tableau where p is the pivot entry.

Assume that the rows are labeled consecutively from top to bottom so that the top-most row is labeled with 1. The pivot entry to be used by the simplex algorithm can then be identified by applying the following algorihm.

Algorithm A.1.1. To determine a pivot entry for a system of linear inequalities do:

1. Let $-b_l$ denote the lowest positive physical entry in the right column.
2. Let a_{lj} be a negative entry in row l; if no such entry is found then no solution exists.
3. **If** $l < m$ **then**
   a) Set $e := b_l/a_{lj}$.
   b) **For** each positive entry a_{ij} $(i > l)$
      If $b_i/a_{ij} < e$ **then**
         Set $e := b_i/a_{ij}$.
   c) Return (k, j), where $e := b_k/a_{kj}$.
4. **If** $l = m$ **then**
   Return (l, j).

□

By iteratively invoking the above algorithm (and making the corresponding pivot transformations) it can be shown that, in the non-degenerate case, the algorithm will lead to a tableau where $-b_l$ is decreased and the $-b_i$'s physically below b_l remain non-postive that is, the $-b_i$'s for which $(i > l)$. However, particular care should be taken in case the tableau is degenerate:

Definition A.1.1. A tableau is said to be a *degenerate tableau* if at least one of the $-b_i$'s is equal to zero.

The problem with a degenerate tableau is that the algorithm might cycle but, in the non-degenerate case we are sure to arrive at a solution in a finite number of steps.

A.2 Modifying the simplex algorithm

In what follows we show how to modify the simplex algorithm to take advantage of the structure of the inequalities when investigating strategies. In particular, we shall exploit the fact that if a row has zero in the pivot column then all entries in that row are unchanged by the pivot transformation.

Given a strategy Δ, consider the set of linear inequalities associated with the nodes of the decision tree T. By introducing slack variables we get (for $y_{c_i} \geq y_{c_{i-1}} + \alpha$):

$$y_{c_i} = y_{c_{i-1}} + s_i + \alpha \Leftrightarrow -s_i = y_{u_{i-1}} - y_{u_i} + \alpha,$$

where $s_i \geq 0$. Inequality 9.12 results in:

$$y_D = \sum_{l \in \mathcal{L}} p_l y_l + \epsilon + t_D \Leftrightarrow -t_D = \sum_{l \in \mathcal{L}} p_l y_l + \epsilon - y_D \tag{A.5}$$

for $t_D \geq 0$. Similarly,

$$y_D = y_x + \epsilon + t_D \Leftrightarrow -t_D = y_x + \epsilon - y_D \tag{A.6}$$

Finally, Inequality 9.13 gives:

$$y_D + t_D = \sum_{l \in \mathcal{L}} p_l y_l \Leftrightarrow -t_D = y_D - \left(\sum_{l \in \mathcal{L}} p_l y_l \right) \tag{A.7}$$

Given a decision node D the set of equalities described by Equality A.5, Equality A.6 and Equality A.7 will be referred to as the *child equalities* associated with D. If D is preceded by another decision node, then the (unique) equality where D occurs on the right-hand side will be referred to as the *parent equality* of D.

Example A.2.1. Consider Figure 9.5, and assume again that L_2 is chosen (i.e. $\Delta = \{(D, L_2)\}$) and that we want to determine whether there exists a utility function that respects the preference ordering over the consequences and yields Δ as an optimal strategy.

In this case the initial tableau is specified as in Table A.3; the boxed entry denotes the pivot element.

y_D	c_1	c_2	c_3	c_4	c_5	c_6	1	
1	-0.7				-0.1	-0.2	0	$-r$
-1		0.4	0.2	0.4			ϵ	$-s$
	1	-1					α	$-t_2$
		1	-1				α	$-t_3$
			1	-1			α	$-t_4$
				1	-1		α	$-t_5$
					1	$\boxed{-1}$	α	$-t_6$

Table A.3: The initial tableau.

Making a pivot transformation w.r.t. the entry in row 7 we get Table A.4.

y_D	c_1	c_2	c_3	c_4	c_5	t_6	1	
1	-0.7				-0.3	-0.2	-0.2α	$-r$
-1		0.4	0.2	0.4			ϵ	$-s$
	1	-1					α	$-t_2$
		1	-1				α	$-t_3$
			1	-1			α	$-t_4$
				1	$\boxed{-1}$		α	$-t_5$
					-1	-1	$-\alpha$	$-c_6$

Table A.4: The tableau obtained from Table A.3 after pivot transformation.

Making a pivot transformation w.r.t. the entry in row 6 we get Table A.5.

y_D	c_1	c_2	c_3	c_4	t_5	t_6	1	
1	-0.7			-0.3	-0.3	-0.2	-0.5α	$-r$
-1		0.4	0.2	0.4			ϵ	$-s$
	1	-1					α	$-t_2$
		1	-1				α	$-t_3$
			1	$\boxed{-1}$			α	$-t_4$
				-1	-1		$-\alpha$	$-c_5$
				-1	-1	-1	-2α	$-c_6$

Table A.5: The tableau obtained from Table A.4 after pivot transformation.

Making a pivot transformation w.r.t. the entry in row 5 we get Table A.6.

y_D	c_1	c_2	c_3	t_4	t_5	t_6	1	
1	-0.7		-0.3	-0.3	-0.3	-0.2	-0.8α	$-r$
-1		0.4	0.6	0.4			$0.4\alpha + \epsilon$	$-s$
	1	-1					α	$-t_2$
		1	$\boxed{-1}$				α	$-t_3$
			-1	-1			$-\alpha$	$-c_4$
			-1	-1	-1		-2α	$-c_5$
			-1	-1	-1	-1	-3α	$-c_6$

Table A.6: The tableau obtained from Table A.5 after pivot transformation.

Making a pivot transformation w.r.t. the entry in row 4 we get Table A.7.

y_D	c_1	c_2	t_3	t_4	t_5	t_6	1	
1	-0.7	-0.3	-0.3	-0.3	-0.3	-0.2	-1.1α	$-r$
-1		1	0.6	0.4			$\alpha + \epsilon$	$-s$
	1	$\boxed{-1}$					α	$-t_2$
		-1	-1				$-\alpha$	$-c_3$
		-1	-1	-1			-2α	$-c_4$
		-1	-1	-1	-1		-3α	$-c_5$
		-1	-1	-1	-1	-1	-4α	$-c_6$

Table A.7: The tableau obtained from Table A.6 after pivot transformation.

Making a pivot transformation w.r.t. the entry in row 3 we get Table A.8.

y_D	c_1	t_2	t_3	t_4	t_5	t_6	1	
1	-1	-0.3	-0.3	-0.3	-0.3	-0.2	-1.4α	$-r$
$\boxed{-1}$	1	1	0.6	0.4			$2\alpha+\epsilon$	$-s$
	-1	-1					$-\alpha$	$-c_2$
	-1	-1	-1				-2α	$-c_3$
	-1	-1	-1	-1			-3α	$-c_4$
	-1	-1	-1	-1	-1		-4α	$-c_5$
	-1	-1	-1	-1	-1	-1	-5α	$-c_6$

Table A.8: The tableau obtained from Table A.7 after pivot transformation.

Making a pivot transformation w.r.t. the entry in row 2 we get Table A.9.

y_D	c_1	t_2	t_3	t_4	t_5	t_6	1	
1	0	0.7	0.3	0.1	-0.3	$\boxed{-0.2}$	$0.6\alpha+\epsilon$	$-r$
-1	-1	-1	-0.6	-0.4			$-2\alpha-\epsilon$	$-s$
	-1	-1					$-\alpha$	$-c_2$
	-1	-1	-1				-2α	$-c_3$
	-1	-1	-1	-1			-3α	$-c_4$
	-1	-1	-1	-1	-1		-4α	$-c_5$
	-1	-1	-1	-1	-1	-1	-5α	$-c_6$

Table A.9: The tableau obtained from Table A.8 after pivot transformation.

The relevant influence by the pivot transformation specified by Table A.9 is restricted to the entry containing -5α (producing $-8\alpha - 5\epsilon$ after the transformation). Thus, the utility function $u(c_1) = 0, u(c_2) = 1, u(c_3) = 2, u(c_4) = 3, u(c_5) = 4, u(c_6) = 8.5$ results in Δ being an optimal strategy; for $\alpha = 1$ and $\epsilon = 0.1$ we get $EU(L_1) = 2$ and $EU(L_2) = 2.1$. □

In what follows we give a suggestion on how to exploit the structure in the tableau when performing the pivot transformations. The method will be illustrated by making references to the example above, and it will be shown that many computations can be done locally. That is, most pivot transformations can be performed by only considering the parent and child equalities of a particular decision node. It should be noticed that if a row describes e.g. the parent equality of some decision node D, then we will keep referring to that row as describing the parent equality for D even after pivot transformations have been performed. Furthermore, we say that a row is *resolved* if it has a negative entry in the $-b$ column.

Conceptually, the method can be seen as a propagation and an absorption of equalities from the leaves towards the root.

Now, assume that the rows in the tableau are indexed from top to bottom so that the top row is indexed with 1. Construct the initial tableau as follows:

i) The equalities encoding the constraints on the consequences are placed at the bottom, i.e., they constitute the rows with highest index (see row 3-7 in Table A.3).

ii) For each decision node D, the parent equality of D is given a lower index than the child equalities of D and the child equalities of D are indexed sequentially (see Figure A.1).

iii) If D' precedes D in T, then all the parent and child equalities of D' are given lower indexes than the parent and child equalities of D e.g. if we augment Figure 9.5 with another decision node D' (preceding D), then the equalities associated with D' would be placed physically above the entries in the initial tableau.

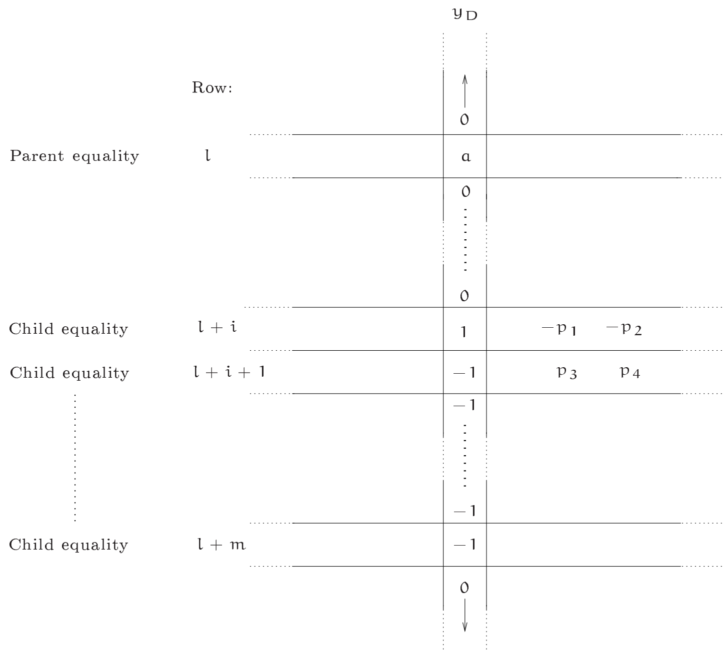

Figure A.1: The figure illustrates the ordering of the rows in the initial tableau.

Now, from the construction of the tableau it follows that we start off by resolving the equalities encoding the constraints on the consequences (see Table A.3-A.8 in Example A.2.1). After resolving these equalities we obtain the general situation outlined in Figure A.1; actually Figure A.1 depicts the ordering of the rows in the initial tableau, however, it is easy to see that this structure is invariant under these initial transformations.

So, assume that the rows depicted in Figure A.1 are the rows placed immediately above the rows encoding the constraints on the consequences; from the construction of the tableau the node D is not followed by any other decision node in T hence, resolving the equalities encoding the constraint on the consequences produces a tableau with a negative entry in the $-\mathsf{b}$ column for each row $\mathsf{j}(\mathsf{l}+\mathsf{i}+1 \leq \mathsf{j} \leq \mathsf{l}+\mathsf{m})$. Furthermore, according to Algorithm A.1.1 the -1 entry in the $(\mathsf{l}+\mathsf{m})$'th row is a legitimate pivot entry; notice that the figure illustrates the situation where D is part of the strategy under investigation.

Making a pivot transformation w.r.t. this entry does not influence the other entries in the y_{D} column as they are multiplied with $-1/-1 = 1$. Moreover, the entries in the tableau which are changed by the pivot transformation are all entries in the rows corresponding to the parent equality and the child equalities of D; all other rows have a zero entry in the y_{D} column. From this it follows that we can continue to pick our pivot entries in this way until we reach row $\mathsf{l}+\mathsf{i}$. Notice that even though each of these pivot transformations changes the row corresponding to the parent equality for D, they do not change the -1 entry in the parent equality (the -1 entry in the column corresponding to the decision node for which the equality occurs as a child equality).

If row $\mathsf{l}+\mathsf{i}$ is resolved by the pivot transformations made w.r.t. to the entries in the rows physically below row $\mathsf{l}+\mathsf{i}$, then we can continue the pivot transformations as described above. So, assume that row $\mathsf{l}+\mathsf{i}$ was not resolved by these pivot transformations. In general, this row does not contain a -1 entry in the y_{D} column thus, we need to identify another negative entry (let $-\mathsf{a}$ denote such an entry if it exists; otherwise the system of inequalities fails to have a solution).

The construction of the tableau does not reveal an immediate way to determine if there exists a positive entry a', physically below $-\mathsf{a}$. So, we should in principle investigate all entries physically below $-\mathsf{a}$. However, suppose that we choose the negative entry $-\mathsf{a}$ such that it occurs in a column which (initially) corresponds to the decision variable D' closest to (and succeeding) D in the decision tree. In this case we need not investigate the equalities physically below the parent and child equalities associated with D', since they all have a zero entry in this column. Finally, making a pivot transformation w.r.t. such an a-entry produces a tableau with a structure consistent with the previous one and, hence, allows us to proceed with the pivot transformations as described above.

It should be noticed that the "problem" with row $\mathsf{l}+\mathsf{i}$ only occurs because D was assumed to be part of Δ that is, if D is not part of Δ then row $\mathsf{l}+\mathsf{i}$ would not be present and we would only need to make pivot transformations w.r.t. the -1 entries in the y_{D} column.

So the computational problem reduces to the set of rows that correspond to the arcs that are part of Δ. For these rows, it is not apparent how to avoid the exponential step, i.e., to resolve row l we can perceive row l together with the

$m - l$ rows physically below row l as a (separate) system of inequalities.

To summarize, by organizing the rows in the initial tableau we can exploit its structure and in particular the occurrence of zero values. This allows us to perform many calculations locally but unfortunately not all of them. However, we may still obtain a substantial reduction in the computational complexity.

Appendix B

Overview of *Sif*

B.1 Introduction

This document is an overview of the *Sif* program for specification, analysis and evaluation of partial influence diagrams. The document is intended to be a relatively superficial guide to the operations supported by the program, since large parts are self explanatory.

The program provides a graphical user interface for specifying partial influence diagrams, and offers various operations for analyzing such models. The program has been developed under Redhat Linux 6.2 and has been implemented using wxWindows (version 2.2.1); wxWindows is a cross-platform API for developing graphical user interfaces.

The terminology in this document will follow that of Chapter 3.

B.2 Description

The *Sif* program contains a graphical user interface for specifying partial influence diagrams. The program also provides its own file format for loading and saving *Sif* models; these models can also be imported/exported in the Hugin net-file format.

Given a partial influence diagram I, the program supports the following types of operations:

- Determine if I is well-defined.

- If I is not well-defined, determine the last chance node under the partial order which is significant for a decision node.
- Resolve possible ambiguities.
- Find the value nodes being relevant for a given decision node.
- Find the nodes being required for a given decision node.
- Find the decision nodes for which a specific set of value nodes are relevant.
- Find the decision nodes for which a specific set of nodes are required.
- Evaluate I w.r.t. the maximum expected utility principle.
- Display the optimal policy for a given decision node.

The rules presented in Chapter 3 are used for determining the nodes relevant and required for a given decision node. Algorithm 3.4.2 is used for determining whether or not a partial influence diagram is well-defined, and the Bayes-ball algorithm (Shachter 1998) has been implemented for testing d-separation. The algorithm for resolving ambiguities is based on the maximum expected utility principle, thereby inserting informational arcs s.t. a chance node which has been identified as being significant (for some decision node) is observed as early as possible in the decision process.

The evaluation of a partial influence diagram is based on variable elimination (see Chapter 5) with the requirement that the partial influence diagram is well-defined. Having evaluated a partial influence diagram, the optimal policy for a decision variable can be displayed. The policy is shown as a tree structure, where the non-leaf nodes constitute the nodes in the required past for the decision variable in question; note that this type of representation requires that the number of nodes in the required past is relatively small.

Appendix C

Dansk resumé

Dette resumé er en direkte oversættelse af indledningen til nærværende afhandling med den danske titel *Grafiske modeller for delvist sekventielle beslutningsproblemer*.

Grafisk modellering i forbindelse med beslutningsstøttesystemer er blevet mere og mere udbredt. For det første er grafiske modeller potentielt stærke, da de oversætter et komplekst problem til en let forståelig form (Almond 1995). Derudover tilbyder de en appellerende måde at tænke på, og kommunikere om, strukturen for det underliggende domæne.

Influensdiagrammer (Howard and Matheson 1981) fungerer som et effektivt modelleringsværktøj for symmetriske beslutningsproblemer med flere beslutninger; styrken ved et influensdiagram, både som analyse- og kommunikationsværktøj, bygger på dets egenskab til koncist og præcist at beskrive strukturen af et beslutningsproblem (Smith et al. 1993). Influensdiagrammer kan dog ikke effektivt repræsentere klassen af asymmetriske beslutningsproblemer. For at repræsentere denne type af beslutningsproblemer, indenfor begrebsrammen influensdiagrammer, skal problemet først "symmetriseres" ved eksempelvis at tilføje kunstige tilstande for at fange asymmetrien. Uheldigvis har denne "symmetrisering" tendens til, at utydeliggøre beslutningsproblemets struktur, og kan desuden medføre en signifikant beregningsmæssig forøgelse (se (Smith et al. 1993; Qi et al. 1994; Covaliu and Oliver 1995; Shenoy 2000)).

Når et beslutningsproblem repræsenteres som et influensdiagram fordres en lineær temporal ordning over beslutningerne; denne forudsætning sikrer at beslutningsproblemet er veldefineret. Et beslutningsproblems struktur kan dog ofte specificere, at nogle beslutninger er betinget uafhængige og i sådanne situationer er det unødvendigt at kræve en sådan ordning. Hvis, for eksempel, der ikke indsamles information mellem to beslutninger, så kan disse beslutninger foretages uafhængigt at hinanden. Denne type af temporal uafhængighed er genkendt i værdibaserede systemer (Shenoy 1992), men temporal uafhængighed kan være mere kompliceret. F.eks. vil observationen af en variabel A, umiddelbart inden en beslutning D foretages, ikke nødvendigvis have nogen indflydelse på denne beslutning (se (Shachter 1999)). Dette indikerer også, at ikke alle observerede variable nødvendigvis har indflydelse på en bestemt beslutningsvariabel, og disse variable kan derfor betragtes som værende irrelevante for denne beslutning. Tilsvarende kan en fremtidig variable også være irrelevant, og for både beregningsmæssige og kommunikationsmæssige årsager er det vigtigt at identificere disse typer irrelevans. Ved, for eksempel, at identificere de dele af et beslutningsproblem som er relevante for en bestemt beslutning (der kan være specielt vigtig) kan vi fokusere på disse dele når vi skal bestemme nytter og sandsynligheder. Denne tilgang kan også suppleres med en analyse af hvor følsom modellen er overfor variationer i sandsynlighederne og nytterne dvs., at når disse værdier skal bestemmes så kan vi fokusere på de værdier som modellen er specielt følsom overfor (se f.eks. (Bielza et al. 2000)).

Når et beslutningsproblem skal løses søger vi typisk en optimal strategi for de involverede beslutninger. Denne strategi beregnes normalt i henhold til teorien om forventet nytte dvs., at strategien der maksimerer den forventede nytte betegnes som den optimale strategi. Uheldigvis har det vist sig, at teorien om forventet nytte har en begrænset værdi som deskriptiv model, se f.eks. (Allais 1979). Fra et normativt synspunkt kan vi ikke blot ignorere denne ufuldkommenhed, da opgaven er at foreslå en strategi som en beslutningstager forventes at ville følge. En beslutningstager er dog typisk ikke villig til blindt at følge en strategi før han har fået tilstrækkelig tillid til den anvendte model. Men da teorien om forventet nytte ikke samtidigt kan redegøre for alle hans valg vil beslutningstageren muligvis forkaste den foreslåede strategi.

I denne afhandling beskriver vi repræsentation, løsning og analyse af beslutningsproblemer i relation til ovenstående emner. For det første betragter vi influensdiagrammer med kun en partiel temporal ordning over beslutningerne (partielle influensdiagrammer), og vi præsenterer en række betingelser som er nødvendige og tilstrækkelige for, at et partielt influensdiagram er veldefineret. Som en generalisering af det partielle influensdiagram foreslår vi en anden begrebsramme, betegnet asymmetriske influensdiagrammer, til at repræsentere og løse asymmetriske beslutningsproblemer; denne begrebsramme repræsenterer hovedsagligt information om asymmetri på det kvalitative niveau, hvorfor denne information kan aflæses direkte fra modellen. I forbindelse med analyse af beslutningsproblemer, foreslår vi en metode til at bestemme de dele af et influensdiagram som er relevante for en bestemt beslutning. I denne kontekst præsenterer vi også to metoder til at bestemme hvor følsomt et influensdiagram er overfor variationer i sandsynlighederne og nytterne. Til slut, betragter vi rationel beslutningstagning baseret på teorier om ikke-forventet nytte, og vi foreslår en metode der giver en kompromisstrategi som ikke er domineret.

C.1 Oversigt over afhandlingen

I Del I betragter vi repræsentation af beslutningsproblemer. Kapitel 2 præsenterer de termer og den notation der anvendes gennem afhandlingen, samt relevante resultater publiseret af andre forfattere. Kapitel 3 præsenterer nødvendige og tilstrækkelige betingelser for at et influensdiagram, med kun en partiel ordning over beslutningerne, er veldefineret. I Kapitel 4 foreslår vi en begrebsramme (betegnet asymmetriske influensdiagrammer) som modelleringssprog for asymmetriske beslutningsproblemer.

Del II omhandler evaluering af beslutningsproblemer. I Kapitel 5 gennemgår vi kort nogle eksisterende evalueringsmetoder til løsning af symmetriske beslutningsproblemer. Disse metoder danner baggrunden for Kapitel 6, hvor vi foreslår en metode til at løse asymmetriske beslutningsproblemer formuleret i begrebsrammen asymmetriske influensdiagrammer.

Del III omhandler emner i forbindelse med analyse af influensdiagrammer. I Kapitel 7 foreslår vi en metode til at dekomponere et influensdiagram til en mængde af mindre influensdiagrammer, så hvert af de mindre influensdiagrammer indeholder præcist de variable som er relevante for en bestemt beslutningsvariabel. Kapitel 8 beskriver to metoder til at foretage følsomhedsanalyse i influensdiagrammer, med fokus på nytteparametre.

Del IV betragter rationel beslutningstagning baseret på teorier om ikke-forventet nytte. I Kapitel 9 præsenterer vi en operationel tilgang til rationel beslutningstagning baseret på rank-dependent utility (Quiggin 1982).

Til sidst giver vi i Kapitel 10 nogle afsluttende bemærkninger om arbejdet dokumenteret i denne afhandling, og skitserer emner for fremtidig forskning.

Bibliography

Allais, M. (1979). The foundation of a positive theory of choice involving risk and a criticism of the postulate and axioms of the American school. In M. Allais and O. Hagen (Eds.), *Expected utility hypotheses and the Allais paradox*, pp. 27–145. Dordrecht, Holland. Original work published in 1952.

Allais, M. (1988). The general theory of random choices in relation to the invariant cardinal utility function and the specific probability function. In B. Munier (Ed.), *Risk, Decision and Rationality*, pp. 233–289. Dordrecht, Holland.

Almond, R. G. (1995). *Graphical Belief Modeling* (first ed.). Chapman and Hall, London, UK. ISBN 0-412-06661-0.

Arnborg, S., D. G. Corneil, and A. Proskurowski (1987, April). Complexity of finding embeddings in a k-tree. *SIAM Journal of Algebraic and Discrete Methods 8*(2), 277–284.

Bellman, R. E. (1957). *Dynamic Programming*. Princeton University Press, Princeton.

Bielza, C., D. Ríos-Insua, and S. Ríos-Insua (1996). Influence diagrams under partial information. In *Bayesian Statistics* 5, pp. 491–497. Oxford University Press.

Bielza, C., S. Ríos-Insua, M. Gómez, and J. A. F. del Pozo (2000). Sensitivity analysis in IctNeo. In *Lecture Notes in Statistics*, Volume 152 of *Robust Bayesian Analysis*, pp. 317–334. Springer.

Bielza, C. and P. P. Shenoy (1998, December). A comparison of graphical techniques for asymmetric decision problems: Supplement to Management Science paper (Bielza and Shenoy 1999). Working paper 282, School of Business, University af Kansas, Summerfield Hall, Lawrence, KS 66045-2003, USA.

Bielza, C. and P. P. Shenoy (1999). A comparison of graphical techniques for asymmetric decision problems. *Management Science 45*(11), 1552–1569.

Boutilier, C., N. Friedman, M. Goldszmidt, and D. Koller (1996). Context-specific independence in Bayesian networks. In E. Horvitz and F. V. Jensen

(Eds.), *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, San Francisco, pp. 115–123. Morgan Kaufmann Publishers.

Call, H. J. and W. A. Miller (1990). A comparison of approaches and implementations for automating decision analysis. *Reliability Engineering and System Safety 30*, 115–162.

Cano, A., S. Moral, and A. Salmerón (2000a, July). Lazy evaluation in penniless propagation over join trees. Technical Report DESCAI-00-02-15, Department of Computer Science and Artificial Intelligence, E.T.S de Ingeniería Informática, Universidad de Granada, Spain.

Cano, A., S. Moral, and A. Salmerón (2000b). Penniless propagation in join trees. *International Journal of Intelligent Systems 15*, 1027–1059.

Castillo, E., J. M. Gutiérrez, and A. S. Hadi (1997). Sensitivity analysis in discrete Bayesian networks. *IEEE Transactions on Systems, Man and Cybernetics 27*(4), 412–423.

Chajewska, U. and D. Koller (2000). Utilities as random variables: Density estimation and structure discovery. In C. Boutilier and M. Goldszmidt (Eds.), *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pp. 63–71.

Chateauneuf, A. and M. Cohen (1984). Risk seeking with diminishing marginal utility in a non-expected utility model. *Journal of Risk and Uncertainty 9*, 77–91.

Chew, S., E. Karni, and Z. Safra (1987). Risk aversion in the theory of expected utility with rank dependent preferences. *Journal of Economic Theory 42*, 370–381.

Cooper, G. F. (March 1990). The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence 42*(2-3), 393–405.

Cooper, G. F. and E. Herskovits (1991). A Bayesian method for constructing Bayesian belief networks from databases. In B. D. D'Ambrosio, P. Smets, and P. P. Bonissone (Eds.), *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, pp. 86–94.

Coupé, V. M. H. (2000). *Sensitivity analysis of decision-theoretic networks.* Ph. D. thesis, Department of Public Health, Erasmus University Rotterdam, 3000 DR Rotterdam, The Netherlands.

Covaliu, Z. and R. M. Oliver (1995). Representation and solution of decision problems using sequential decision diagrams. *Management Science 41*(12), 1860–1881.

Cowell, R. G., A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter (1999). *Probabilistic Networks and Expert Systems.* Springer-Verlag.

Currim, I. S. and R. K. Sarin (1989). Prospect versus utility. *Management Science 35*(1), 22–41.

Dantzig, G. B. and P. Wolfe (1960). Decomposition principle of linear programs. *Operations Research 8*(1), 101–111.

Dechter, R. (1996). Bucket elimination: A unifying framework for probabilistic inference. In E. Horvitz and F. V. Jensen (Eds.), *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers.

Demirer, R. and P. P. Shenoy (2001). Sequential valuation networks: A new graphical technique for asymmetric decision problems. In *Lecture Notes in Artificial Intelligence*, pp. 252–265. Springer.

Dittmer, S. L. and F. V. Jensen (1997). Myopic value of information in influence diagrams. In D. Geiger and P. P. Shenoy (Eds.), *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers.

Doubilet, P., C. B. Begg, M. C. Weinstein, P. Braun, and B. J. McNeil (1985). Probabilistic sensitivity analysis using Monte Carlo simulation. *Medical Decision Making 5*(2), 157–177.

Fagiuoli, E. and M. Zaffalon (1998). A note about redundancy in influence diagrams. *International Journal of Approximative Reasoning 19*(3-4), 231–246.

Felli, J. C. and G. B. Hazen (1998). Sensitivity analysis and the expected value of perfect information. *Medical Decision Making 18*, 95–109.

Felli, J. C. and G. B. Hazen (1999). Do sensitivity analysis really capture problem sensitivity? An empirical analysis based on information value. *Risk, Decision and Policy 4*(2), 79–98.

Geiger, D., T. Verma, and J. Pearl (1990a). d-separation: From theorems to algorithms. In *Uncertainty in Artificial Intelligence 5*.

Geiger, D., T. Verma, and J. Pearl (1990b, August). Identifying independence in Bayesian networks. *Networks 20*(5), 507–534. Special Issue on Influence Diagrams.

Hadar, J. and W. Russell (1969). Rules for ordering uncertain prospects. *American Economic Review 59*(1), 97–122.

Højsgaard, S. (1998). *Split models and contingency tables*. Ph. D. thesis, Ministry of Food, Agrigultures and Fisheries, Danish Institute of Agricultural Sciences, Biometry Research Unit, Denmark.

Horwitz, E. and M. Barry (1995). Display of information for time-critical decision making. In P. Besnard and S. Hanks (Eds.), *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, pp. 296–305. Morgan Kaufmann Publishers.

Howard, R. A. (1962). The used car buyer. In R. A. Howard and J. E. Matheson (Eds.), *The Principles and Applications of Decision Analysis*, Volume 2, Chapter 36, pp. 691–718. Strategic Decision Group.

Howard, R. A. and J. E. Matheson (1981). Influence diagrams. In R. A. Howard and J. E. Matheson (Eds.), *The Principles and Applications of Decision Analysis*, Volume 2, Chapter 37, pp. 721–762. Strategic Decision Group.

Jaffray, J.-Y. (1999). Rational decision making with imprecise probabilities. In 1*st International Symposium on Imprecise Probabilities and Their Applications*, pp. 183–188. Morgan Kaufmann Publishers.

Jensen, F., F. V. Jensen, and S. L. Dittmer (1994). From influence diagrams to junction trees. In R. L. de Mantaras and D. Poole (Eds.), *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pp. 367–373. Morgan Kaufmann Publishers.

Jensen, F. V. (1996). *An introduction to Bayesian Networks*. UCL Press Limited, London, UK. ISBN: 1-85728-332-5 HB.

Jensen, F. V. (2001). Graphical models as languages for computer assisted diagnosis and decision making. In *Lecture Notes in Artificial Inteligence*. Springer. 1-15.

Jensen, F. V. and F. Jensen (1994). Optimal junction trees. In R. L. de Mantaras and D. Poole (Eds.), *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pp. 360–366. Morgan Kaufmann Publishers.

Jensen, F. V., S. L. Lauritzen, and K. G. Olesen (1990). Bayesian updating in causal probabilistic networks by local computations. *Computational Statistics Quarterly 4*, 269–282.

Kahneman, D. and A. Tversky (1979). Prospect theory: An analysis of decision under risk. *Econometrica 47*, 263–291.

Karmarkar, N. (1984). A new polynomial-time algorithm for linear progamming. *Combinatorica 4*, 373–395.

Karni, E. and Z. Safra (1989). Ascending bid auctions with behaviorally consistent bidders. *Annals of Operations Research 19*, 435–446.

Kirkwood, C. W. (1993, July). An algebraic approach to formulating and solving large models for sequential decision under uncertainty. *Management Science 39*(7), 900–913.

Kjærulff, U. (1993, April). *Aspects of efficiency improvement in Bayesian networks*. Ph. D. thesis, Department of Computer Science, Aalborg University, Denmark.

Kjærulff, U. and L. C. van der Gaag (2000). Making sensitivity analysis computationally efficient. In C. Boutilier and M. Goldszmidt (Eds.), *Proceedings* *of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pp. 317–325. Morgan Kaufmann Publishers.

Laskey, K. B. (1995). Sensitivity analysis for probability assessments in Bayesian networks. *IEEE Transactions on Systems, Man and Cybernetics 25*, 901–909.

Lauritzen, S. L., A. P. Dawid, B. N. Larsen, and H. G. Leimer (1988, October). Independence properties of directed Markov fields. Research Report F 88-32, Institute for Electronic Systems, Aalborg University, Denmark.

Lauritzen, S. L., A. P. Dawid, B. N. Larsen, and H.-G. Leimer (1990, August). Independence properties of directed Markov fields. *Networks 20*(5), 491–505.

Lauritzen, S. L. and D. J. Spiegelhalter (1988). Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society 50*(2), 157–224.

Machina, M. J. (1989). Dynamic consistency and non-expected utility models of choice under uncertainty. *Journal of Economic Literature 27*, 1622–1688.

Madsen, A. L. and F. V. Jensen (1999). Lazy evaluation of symmetric Bayesian decision problems. In K. B. Laskey and H. Prade (Eds.), *Proceedings of the Fifthteenth Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers.

Madsen, A. L. and D. Nilsson (2001). Solving influence diagrams using HUGIN, Shafer-Shenoy and Lazy propagation. In J. Breese and D. Koller (Eds.), *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, pp. 337–345. Morgan Kaufmann Publishers.

Madsen, A. L., K. G. Olesen, and S. L. Dittmer (2001, october). Practical modeling of Bayesian decision problems - exploiting deterministic relations. IEEE Transactions on Systems, Man, and Cybernetics, To appear.

McClennen, E. F. (1990). *Rationality and Dynamic choice: Foundational explorations*. Cambridge University Press, Cambridge.

McCord, M. R. and R. de Neufville (1984). Utility dependence on probability: An empirical demonstration. *Journal of Large Scale Systems 6*, 91–103.

Ndilikilikesha, P. C. (1994). Potential influence diagrams. *International Journal of Approximate Reasoning 10*, 251–285.

Nielsen, T. D. (2001). Decomposition of influence diagrams. In *Lecture Notes in Artificial Intelligence*, pp. 144–155. Springer.

Nielsen, T. D. and J.-Y. Jaffray (2001a). An operational approach to rational decision making based on rank dependent utility. Technical Report R-01-5001, Department of Computer Science, Fredrik Bajers 7E, 9220 Aalborg, Denmark.

Nielsen, T. D. and J.-Y. Jaffray (2001b). An operational approach to rational decision making based on rank dependent utility. Submitted for publication.

Nielsen, T. D. and F. V. Jensen (1999a). Representing and solving asymmetric Bayesian decision problems. Technical report, Department of Computer Science, Fredrik Bajers 7C, 9220 Aalborg, Denmark. R-99-5010.

Nielsen, T. D. and F. V. Jensen (1999b). Welldefined decision scenarios. In K. B. Laskey and H. Prade (Eds.), *Proceedings of the Fifthteenth Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers.

Nielsen, T. D. and F. V. Jensen (2000). Representing and solving asymmetric Bayesian decision problems. In C. Boutilier and M. Goldszmidt (Eds.), *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers.

Nielsen, T. D. and F. V. Jensen (2001a). Sensitivity analysis in influence diagrams. Submitted for publication.

Nielsen, T. D. and F. V. Jensen (2001b). Welldefined decision scenarios. Technical Report R-01-5002, Department of Computer Science, Fredrik Bajers 7E, 9220 Aalborg, Denmark.

Nilsson, D. and F. V. Jensen (1998). Probabilities of future decisions. In *Proceedings from the International Conference on Informational Processing and Management of Uncertainty in Knowledge-Based Systems*, pp. 1454–1461. Reprinted in Information, Uncertainty and Fusion (2000), 161-171.

Nilsson, D. and S. L. Lauritzen (2000). Evaluating influence diagrams using LIMIDs. In C. Boutilier and M. Goldszmidt (Eds.), *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pp. 436–345.

Olmsted, S. M. (1983). *On representing an solving decision problems*. Ph. D. thesis, Department of Engineering-Economic Systems, Stanford University.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. Networks of Plausible Inference. San Mateo California: Morgan Kaufmann Publishers. ISBN 0-934613-73-7.

Poh, K. L. and E. J. Horvitz (1993). Reasoning about the value of decision-model refinement: Methods and applications. In D. Heckerman and A. Mamdani (Eds.), *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers.

Qi, R., N. L. Zhang, and D. Poole (1994). Solving asymmetric decision problems with influence diagrams. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pp. 491–497. Morgan Kaufmann Publishers.

Quiggin, J. (1982). A theory of anticipated utility. *Journal of Economic Behavior and Organisation 3*(4), 323–343.

Raiffa, H. (1968). *Decision Analysis, Introductory Lectures on Choices under Uncertainty*. Massachusetts: Addison-Wesley.

Raiffa, H. and R. Schlaifer (1961). *Applied Statistical Decision Theory*. MIT press, Cambridge.

Rose, D. J., R. E. Tarjan, and G. S. Lueker (1976). Algorithmic aspects of vertex elimination on graphs. *SIAM Journal on Computing 5*(2), 266–283.

Rothenberg, R. I. (1979). *Linear Programming*. Elsevier North Holland. ISBN: 0-444-00325-8.

Schick, F. (1986). Dutch bookies and money pumps. *Journal of Philosophy 83*, 112–119.

Schrijver, A. (1986). *Theory of Linear and Integer Programming*. Wiley, N-Y.

Shachter, R. D. (1998). Bayes ball: The rational pastime (for determining irrelevance and requisite information in belief networks and influence diagrams). In G. F. Cooper and S. Moral (Eds.), *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, pp. 480–487. Morgan Kaufmann Publishers.

Shachter, R. D. (1999). Efficient value of information computation. In K. B. Laskey and H. Prade (Eds.), *Proceedings of the Fifthteenth Conference on Uncertainty in Artificial Intelligence*, pp. 594–601. Morgan Kaufmann Publishers.

Shachter, R. D. (February 1986). Evaluating influence diagrams. *Operations Research Society of America 34*(6), 79–90.

Shenoy, P. P. (1992). Valuation-based systems for Bayesian decision analysis. *Operations Research 40*(3), 463–484.

Shenoy, P. P. (2000). Valuation network representation and solution of asymmetric decision problems. *European Journal of Operations Research 121*(3), 579–608.

Skaanning, C. (2000). A knowledge acquisition tool for Bayesian-network troubleshooters. In C. Boutilier and M. Goldszmidt (Eds.), *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pp. 549–557. Morgan Kaufmann Publishers.

Smith, J. E., S. Holtzman, and J. E. Matheson (March/April 1993). Structuring conditional relationships in influence diagrams. *Operations research 41*(2), 280–297.

Tarjan, R. E. and M. Yannakakis (1984). Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal of Computing 13*(3), 566–579.

Tatman, J. A. and R. D. Shachter (March/April 1990). Dynamic programming and influence diagrams. *IEEE Transactions on Systems, Man and Cybernetics 20*(2), 365–379.

Tucker, A. W. (1964). Combinatorial algebra of matrix games and linear programs. In E. F. Beckenbach (Ed.), *Applied Combinatorial Mathematics*, Chapter 11, pp. 320–347. New York: John Wiley and Sons.

von Neumann, J. and O. Morgenstern (1944). *Theory of Games and Economic Behavior* (First ed.). John Wiley & Sons, New York.

Wakker, P. P. (1998). Non-expected utility as aversion of information. *Journal of Behavioral Decision Making 1*, 169–175.

Wang, H. and M. J. Druzdzel (2000). User interface tools for navigating in conditional probability tables and elicitation of probabilities in Bayesian networks. In C. Boutilier and M. Goldszmidt (Eds.), *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pp. 617–625. Morgan Kaufmann Publishers.

Wen, W. X. (1990). Optimal decomposition of belief networks. In *Proceedings of the Sixth Workshop on Uncertainty in Artificial Intelligence*, Cambridge, MA, pp. 537–552.

Wu, G. and R. Gonzalez (1996). Curvature of the probability weighting function. *Management Science 42*(12), 1676–1690.

Yaari, M. (1987). The dual theory of choice under risk. *Econometrica 55*(1), 95–115.

Yannakakis, M. (1981). Computing the minimum fill-in is NP-complete. *SIAM Journal on Algebraic and Discrete Methods 2*(1), 77–79.

Zhang, N. L. (1994). *A computational theory of decision networks*. Ph. D. thesis, Department of Computer Science, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong.

Zhang, N. L. (1998). Probabilistic inference in influence diagrams. In G. F. Cooper and S. Moral (Eds.), *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, pp. 514–522. Morgan Kaufmann Publishers.

Index