AALBORG UNIVERSITY

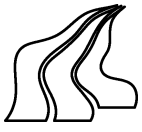
An operational approach to rational decision making based on rank dependent utility

by

Thomas D. Nielsen and Jean-Yves Jaffray

R-01-5001 February 2001

Department of Computer Science
Decision Support Group
Fredrik Bajers Vej 7E — DK 9220 Aalborg — Denmark
Tel.: +45 96 35 80 80 — Fax +45 98 15 98 89



An operational approach to rational decision making based on rank dependent utility.

Thomas D. Nielsen
Department of Computer science
Fredrik Bajers Vej 7E, 9220 Aalborg Øst
Denmark

Jean-Yves Jaffray
Université Paris 6
8, rue du Capitaine Scott 75015 Paris
France

February 2001

Abstract

Non-expected utility (non-EU) theories, such as rank dependent utility (RDU) theory, have been proposed as alternative models to EU theory in decision making under risk. These models do not share the separability property of EU theory hence, in dynamic decision problems, the sophisticated strategy is likely to be dominated w.r.t. stochastic dominance.

Although a rational non-EU 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 EU 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 RDU criterion, confirms the computational tractability of the model.

1 Introduction

The standard normative model for decision making under risk, i.e., under uncertainty represented by probabilities, is *expected utility* (EU) *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* (DM), the so-called *von Neumann-Morgenstern* (vNM) *utility function*. Unfortunately, it has turned out that EU 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 [Chajewska and Koller, 2000]) 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 EU theory. The Analyst's task is to propose a solution which the DM is presumed to undertake. However, the DM is usually not prepared to blindly follow the Analyst's advice before he has gained sufficient confidence in the model used. Since the DM 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 DM is likely to reject the model as a result of the inability of EU 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 EU theory. This problem has motivated the consideration of various alternative models, generically known as non-EU theories. Most of these models depart from EU theory (which they nonetheless include as a special case) by using criteria which not only have lost the additive separability of the EU 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 EU criteria; EU criteria support *backward induction* (*dynamic programming*), a procedure justified by the validity of Bellman's principle. When this principle is invalid, which is generally the case with non-separable criteria, one appears bound to directly evaluate and compare all strategies, a frequently impossible task!

However, a closer look at the implications of non-separability shows that non-EU 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-EU theories have also been criticized from the normative point of view, as they appear to make the DM open to manipulations that can induce sure losses or more generally, make the DM 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-EU criteria rest on the assumption that the DM 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 influenced neither by the past nor by counterfactuals. [Machina, 1989] claims that it is inappropriate to impose the property of consequentialism on non-EU behavior since the very notion of a non-EU criterion implies a type of non-separability which contradicts the consequentialist assumption.

By renouncing consequentialism, the DM 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 DM 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 DM 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 DM's multiple *Selves* (associated with the DM'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 paper we present an operational approach to the model by [Jaffray, 1999], where the non-EU criterion attributed to the DM 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 EU model [Raiffa and Schlaifer, 1961].

The remainder of the paper is organized as follows. In Section 2 we briefly review some basic concepts and notation relating to decision making based on both expected and non-expected utility.

Additionally we describe the framework suggested by [Jaffray, 1999] and outline the structure of the proposed algorithm. The algorithm is based on the identification of dominated strategies which is considered in Section 3. In Section 4 the proposed algorithm is specified and in Section 5 we present some preliminary results. In Section 6 we discuss various extensions to the proposed algorithm.

2 Preliminaries

2.1 Definitions and Notation

2.1.1 Decision trees

A *graph* $G = (\mathcal{N}, \mathcal{E})$ consists of a finite set of nodes $\mathcal{N}$ and a finite set of edges $\mathcal{E}$. An edge is a pair of nodes (X, Y), where $X, Y \in \mathcal{N}$ and $X \neq Y$; if X and Y are ordered then the edge is said to be *directed* otherwise it is *undirected* (if (X, Y) is directed from X to Y, then X is said to be the *parent* of Y and Y is said to be the *child* of X). If all edges in $\mathcal{E}$ are directed then G is said to be a *directed graph* otherwise it is an *undirected graph*.

A *path* from a node X to a node Y in a graph $G = (\mathcal{N}, \mathcal{E})$ is a set of edges $\mathcal{E}' = \{(X_1, X_2), (X_2, X_3), \ldots, (X_{n-1}, X_n)\}$, where $\mathcal{E}' \subseteq \mathcal{E}$ and $X = X_1$ and $Y = X_n$; the *length* of $\mathcal{E}'$ is the number of edges in $\mathcal{E}'$. The *path* $\mathcal{E}'$ is a *directed path* if (X_i, X_{i+1}) is directed and X_i is the parent of X_{i+1}, $\forall 1 \leq i \leq n-1$.

A *subgraph* G' of a graph $G = (\mathcal{N}, \mathcal{E})$ induced by a set of nodes $\mathcal{N}' \subseteq \mathcal{N}$ is the graph $G = (\mathcal{N}', \mathcal{E}')$, where $\mathcal{E}' = \{(X, Y) | \{X, Y\} \subseteq \mathcal{N}' \text{ and } (X, Y) \in \mathcal{E}\}$. A directed graph is said to be a *tree* if each pair of distinct nodes is connected by exactly one path and there exists exactly one node (termed the *root*) to which there does not exist a directed path. A node X is said to *precede* another node Y in a tree T if there exists a directed path from X to Y (analogously, Y is said to *succeed* X in T). The *subtree* of $T = (\mathcal{N}, \mathcal{E})$ having a node X as root (denoted $T(X) = (\mathcal{N}_{T(X)}, \mathcal{E}_{T(X)})$) is the subgraph of T induced by X and the successors of X in T.

A tree T is said to be a *decision tree* if the nodes $\mathcal{N}$ can be partitioned into three disjoint subsets: *decision nodes* $\mathcal{N}_D$ (drawn as rectangles), *chance nodes* $\mathcal{N}_C$ (drawn as circles) and *value nodes* (or *consequences*) $\mathcal{C}$ satisfying that there is a directed path from a node X if and only if $X \notin \mathcal{C}$; $\mathcal{C}$ are the *leaves* in the decision tree. 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 $\mathcal{D}_D$. 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 of a decision node D is denoted $\mathcal{C}_D$. The past of a node X (denoted past(X)) is the configuration specified by the path from the root to X. Finally, we denote by sp_X the set of all children of X in T.

Definition 1. Let T be a decision tree, and let $\mathcal{N}^\Delta \subseteq \mathcal{N}$ be a set of nodes containing:

i) The root node.

ii) Exactly one child of each $X \in \mathcal{N}_D^\Delta = \mathcal{N}_D \cap \mathcal{N}^\Delta$.

iii) All children of each $X \in \mathcal{N}_C^\Delta = \mathcal{N}_C \cap \mathcal{N}^\Delta$.

Then the set of directed edges $\Delta = \{(Y, Z) | Y \in \mathcal{N}_D^\Delta, Z \in \mathcal{N}^\Delta\} \subseteq \mathcal{E}$ is a *strategy*.

The restriction of a strategy to a subtree $T(X)$, which in fact is a strategy in $T(X)$, is called a *substrategy*. Notice that the definition of a strategy is not as the definition in game theory, where a strategy is a set of directed edges consisting of exactly one directed edges for each $X \in \mathcal{N}_D$.

In the remainder of this paper we assume that a decision tree always has a decision node as root node.

2.1.2 Decision making under risk

The directed edges $(X, Y), Y \in sp_X$ emanating from a chance node X are associated with the alternative states of knowledge, or *events*, that may be achieved at X.

For every chance node X, we assume that a probability distribution $\{P((X, Y)|past(X)), Y \in sp_X\}$ specifies the probabilities of the achievable events, conditioned on the events and decisions belonging to the path from the root to X, past(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 leaf (consequence) from X given past(X). A situation where any (sub)strategy generates a probability distribution on the consequence set is known as a situation of *decision making under risk*.

2.2 Single stage decision making

2.2.1 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 1).

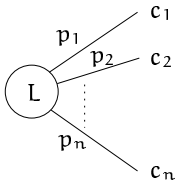


Figure 1: A graphical representation of the single stage lottery $L = (c_1, p_1; c_2, p_2; \ldots; c_n, p_n)$.

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 DM'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 DM'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:

$$\begin{aligned} V(L) > V(L') &\Leftrightarrow L \succ L' \\ V(L) = V(L') &\Leftrightarrow L \sim L' \end{aligned}$$

Having established this preference ordering, the reduction of compound lotteries can be illustrated as in Figure 2.

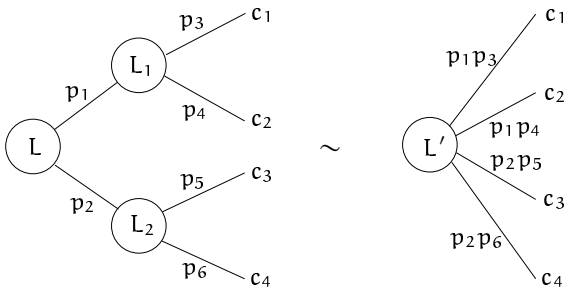


Figure 2: The figure illustrates a compound lottery and its probabilistically equivalent single stage lottery; the DM is assumed to be indifferent between L and L'.

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.

2.2.2 Expected utility

Under EU theory, the preference function $V(\cdot) = EU(\cdot)$ on a lottery L takes on the form:

$$EU(L)_u = EU(c_1, p_1; ...; c_i, p_i; ...; c_n, p_n.)_u = \sum_{i=1}^{n} u(c_i)p_i, \tag{1}$$

where the function $u(\cdot)$ is a personal characteristic of the DM, his von Neumann-Morgenstern (vNM) utility function. The identification of a constant lottery with its unique consequence makes $u(\cdot)$ a utility representing the DM'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.[1]

The preference function of EU 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:

$$EU(L_\Delta)_u \geq EU(L_{\Delta'})_u, \forall \Delta' \neq \Delta$$

EU theory can be justified as an axiomatic model. The cornerstone of the axiom system is the following property which is clearly implied by the linearity of the preference function w.r.t. the probabilities.

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 DMs who have the same preference, $\succsim$, under certainty, hence with $u_i(\cdot)$ in $\mathcal{U} = \{u(\cdot) : \mathcal{C} \to \Re | u(c) \geq u(c') \Leftrightarrow c \succsim c'\}$. Although the model allows lotteries to be ranked differently by different DMs, it is generally agreed that consistency with the following partial ordering is a natural rationality requirement.

[1] 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.

Definition 2 (First order stochastic dominance: FSD). 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 $k \in \Re$ and
- $\sum_{\{i:u(c_i)\leq k\}} p_i < \sum_{\{i:u(c_i)\leq k\}} p'_i$, for some $k \in \Re$.

Note that this relation does not depend on the choice of $u(\cdot)$ in $\mathcal{U}$. 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 (pointwise dominance).

EU criteria are necessarily consistent with FSD. Actually, this property provides a characterization of FSD, as stated in the following theorem due to [Hadar and Russel, 1969], which is exploited later in this paper.

Theorem 1. Let $\mathcal{U}$ be the set of utility functions representing the DM's ordering on the consequence set $\mathcal{C}$, and let L and L' be two lotteries. Then the following statements are equivalent:

i) L FSD L'.

ii) $\forall u \in \mathcal{U} : EU(L)_u \geq EU(L')_u$ and $\exists u \in \mathcal{U} : EU(L)_u > EU(L')_u$.

The theorem also implies that if there exists a utility function $u \in \mathcal{U}$ 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 $u \in \mathcal{U}$ it holds that $EU(L)_u \geq EU(L')_u$.

Another important type of stochastic dominance is *second order stochastic dominance (SSD)* which is involved in a possible definition of *risk aversion*.

Definition 3 (Second order stochastic dominance: SSD). 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 $k \in \Re$ 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 $k \in \Re$.

As for FSD, this relation does not depend on the choice of $u(\cdot)$ in $\mathcal{U}$ and, like FSD, the definition of SSD can be formulated in terms of a specific class of utility functions; for SSD we consider the same set of utility functions as for FSD except that the utility functions are also required to be concave.

Theorem 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 SSD L'.

ii) For every concave $u \in \mathcal{U} : EU(L)_u \geq EU(L')_u$, and there exists a concave $u \in \mathcal{U} : EU(L)_u > EU(L')_u$.

In the remainder of this paper we shall mainly consider stochastic dominance of the first order thus, if not stated otherwise, any future reference to dominance will implicitly refer to FSD.

2.2.3 A non-EU model: Rank Dependent Utility (RDU)

Motivated by experimental observations of phenomena inconsistent with EU theory, in particular the *Allais Paradox* [Allais, 1979] and [Kahneman and Tversky, 1979], several authors have suggested an alternative to EU theory, now most often called *Rank Dependent Utility* (RDU) 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; the preference function in the RDU model has the form of a *Choquet integral* [Choquet, 1953].

In the RDU model, there are two parameters: i) a utility function $u(\cdot)$ which, as the vNM utility function of EU theory, is cardinal and represents the DM'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 RDU 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 DM 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 RDU theory, preferences are consistent with FSD; the function $q(\cdot)$ is strictly increasing. It can be shown that consistency with *strong risk aversion* (L SSD L' 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 Gonzales, 1996]). Examples include:

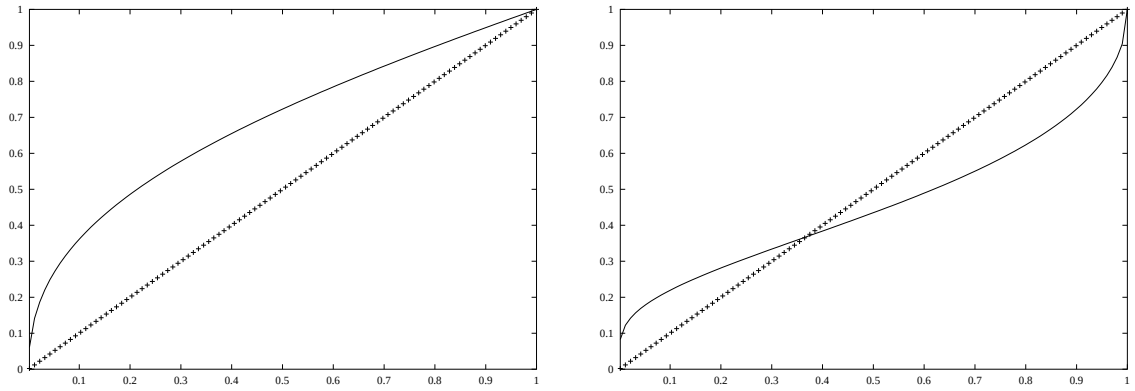
$$\begin{aligned} q(p) &= \frac{p^\gamma}{(p^\gamma + (1-p)^\gamma)^\delta} \text{ (See Figure 3a)} \\ q(p) &= \exp(-(-\ln(p))^\omega)) \text{ (See Figure 3b)} \end{aligned}$$

Note that RDU theory no longer satisfies Axiom 1.

2.3 Dynamic Decision Making

2.3.1 Dynamic consistency

Consider a decision problem represented by a decision tree, and suppose that at each decision node the DM has a preference ordering on the substrategies that are feasible at that node. We say that the DM is *dynamically consistent* if substrategies of optimal strategies are optimal substrategies.



(a) (b)

Figure 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.

More precisely, assume that Δ_0 is an optimal strategy of the subtree rooted at decision node D_0 given the preference ordering at D_0. If decision node D_1 is reachable from D_0 with Δ_0, then substrategy Δ_1, the restriction of Δ_0 to the subtree rooted at D_1, should itself be optimal for the preference ordering at that node.

A DM who is an EU 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 DM who is an RDU maximizer at each decision node (always with the same vNM utility function and probability weighting function). The following example shows that this DM can be dynamically inconsistent; this problem was initially addressed by [Schick, 1986].

Example 1. Consider the decision tree depicted in Figure 4 and assume that the DM's preferences are expressed by an RDU preference function, where the probability weighting function is given by:

$$q(p) = \exp(-(-\ln(p))^{0.5})$$

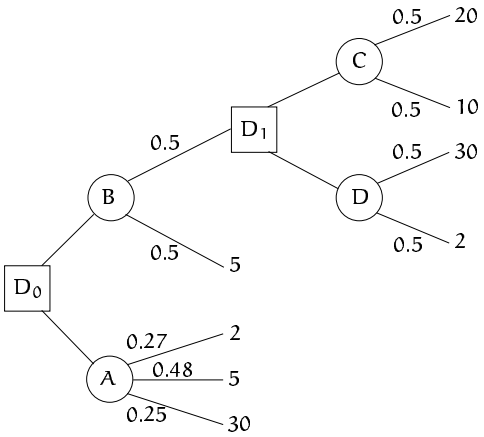


Figure 4: A decision tree representation of a dynamic decision problem with two decisions. The values associated with the leaf nodes correspond to the utilities of these consequences.

At D_0 the possible strategies are $\{(D_0, A)\}$, $\{(D_0, B), (D_1, C)\}$ and $\{(D_0, B), (D_1, D)\}$, where:

$$\begin{aligned} V(\{(D_0, A)\}) &= 2 + (5-2) \cdot q(0.48 + 0.25) + (30-5) \cdot q(0.25) = 11.41 \\ V(\{(D_0, B), (D_1, C)\}) &= 5 + (10-5) \cdot q(0.25 + 0.25) + (20-10) \cdot q(0.25) = 10.26 \\ V(\{(D_0, B), (D_1, D)\}) &= 2 + (5-2) \cdot q(0.5 + 0.25) + (30-5) \cdot q(0.25) = 11.46 \end{aligned}$$

Thus, at D_0 the DM would prefer $\{(D_0, B), (D_1, D)\}$ over $\{(D_0, B), (D_1, C)\}$ and $\{(D_0, A)\}$ however, at D_1 we have:

$$\begin{aligned} V(\{(D_1, C)\}) &= 10 + (20-10) \cdot q(0.5) = 14.35 \\ V(\{(D_1, D)\}) &= 2 + (30-2) \cdot q(0.5) = 14.18 \end{aligned}$$

That is, at D_1 the DM would prefer $\{(D_1, C)\}$ over $\{(D_1, D)\}$ meaning that the DM is not dynamically consistent. □

2.3.2 A dilemma

Dynamic consistency is a relation between preference orderings. Let us now examine its implications concerning choice. At any given decision node, the DM ranks complete substrategies but can only control the immediate decision. It is generally admitted that, aware of that fact, an intelligent DM 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*.

Going back to the preceding example, we see that a sophisticated behavior makes our dynamically inconsistent RDU maximizer choose an FSD-dominated strategy. This is generally considered as irrational.

Example 2. For the decision problem illustrated in Figure 4, the sophisticated strategy is $\{(D_0, A)\}$ which is FSD-dominated by $\{(D_0, B), (D_1, D)\}$. This can easily be seen by considering the corresponding cumulative distribution functions. □

Such an irrational choice cannot be observed with an EU maximizer. Consider then a dynamically consistent DM whose preferences at the root node are expressible by a RDU 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 3. Assume that the DM is dynamically consistent, i.e., at D_0 the strategy $\{(D_0, B), (D_1, D)\}$ is preferred, which implies that $\{(D_1, D)\}$ is preferred at D_1. However, if the utility 5, in the subtree defined by (D_0, B), is replaced by the utility 9.5 we get:

$$\begin{aligned} V(\{(D_0, B), (D_1, C)\}) &= 9.5 + (10-9.5) \cdot q(0.25 + 0.25) + (20-10) \cdot q(0.25) = 12.80 \\ V(\{(D_0, B), (D_1, D)\}) &= 2 + (9.5-2) \cdot q(0.5 + 0.25) + (30-9.5) \cdot q(0.25) = 12.70 \end{aligned}$$

That is, in this modified decision problem the DM would prefer $\{(D_0, B), (D_1, C)\}$ at D_0 and therefore $\{(D_1, C)\}$ at D_1. Hence, the DM's choice at D_1 depends on a value outside the subtree having D_1 as root. □

Consequentialism precisely forbids such an influence. Thus the adaptation of RDU 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-EU behavior since the very notion of a non-EU 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.

2.3.3 Resolute choice

Resolute choice is a type of behavior introduced and discussed in [McClennen, 1990]: The resolute DM 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 DM 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 DM 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 DM 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 DM.

With this objective, [Jaffray, 1999] elaborates on resolute choice and suggests a model which can be interpreted along the following lines: 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 non-cooperative 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.

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:

$$\begin{aligned}\forall \Delta' \neq \Delta, \text{ either } \exists \mathsf{u} \in \mathcal{U} \text{ s.t. } \mathsf{EU}(\mathsf{L}_\Delta)_\mathsf{u} > \mathsf{EU}(\mathsf{L}_{\Delta'})_\mathsf{u} \\ \text{or } \forall \mathsf{u} \in \mathcal{U} : \mathsf{EU}(\mathsf{L}_\Delta)_\mathsf{u} \geq \mathsf{EU}(\mathsf{L}_{\Delta'})_\mathsf{u},\end{aligned}$$

where $\mathcal{U}$ is defined as in Theorem 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 expression above, we 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:

$$\exists u \in \mathcal{U} \text{ s.t. } \mathsf{EU}(L_\Delta)_u > \mathsf{EU}(L_{\Delta'})_u, \forall \Delta' \neq \Delta \tag{2}$$

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.

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 2).

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:

$$\forall 2 \leq i \leq m : y_{c_i} \geq y_{c_{i-1}} + \alpha, \tag{3}$$

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:

$$y_D \geq \sum_{l \in \mathcal{L}} p_l y_l + \epsilon, \tag{4}$$

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:

$$y_D \leq \sum_{l \in \mathcal{L}} p_l y_l. \tag{5}$$

If X is a decision node or a value node we have $y_D \leq y_X$.

Example 4. 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 5).

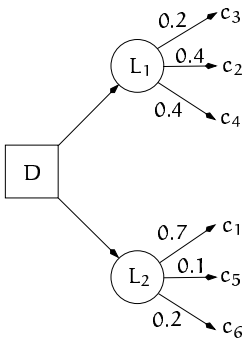


Figure 5: A decision tree with one decision node representing the choice between the two lotteries L_1 and L_2; the preference ordering over the consequences correspond to their index.

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:

$$\begin{aligned} y_D &\geq 0.2 \cdot y_{c_3} + 0.4 \cdot y_{c_2} + 0.4 \cdot y_{c_4} + \epsilon \\ y_D &\leq 0.7 \cdot y_{c_1} + 0.1 \cdot y_{c_5} + 0.2 \cdot y_{c_6} \end{aligned}$$

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 3. Let T be a decision tree with consequences $\mathcal{C}$ and let Δ be a strategy in T. Then there exists a utility function $u \in \mathcal{U}$ 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 4. 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.

4 Evaluation

In this section we present an algorithm for solving decision problems w.r.t. non-EU criteria such as RDU. The underlying idea of the algorithm is to assume consequentialist preferences and to accept non-consequentialist behavior as described in Section 2.2.3.

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.

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 $V(\Delta)$ 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 5. Let T be a decision tree and let D be a decision node in T. If there does not exist a utility function $\mathfrak{u} \in \mathcal{U}$ s.t. Δ is a unique EU-optimal strategy w.r.t. $\mathfrak{u}$ in $\mathsf{T}(\mathsf{D})$ then for any strategy Δ' that induces Δ, there does not exist a utility function $\mathfrak{u}' \in \mathcal{U}$ s.t. Δ' is a unique EU-optimal strategy w.r.t. $\mathfrak{u}'$.

Proof. Follows immediately. □

Before presenting the algorithm we need the following definition.

Definition 4. Let $\mathsf{T}(\mathsf{D})$ be a decision tree with root D, and let $\mathcal{D}_{(\mathsf{D},\mathsf{X})}$ be the set of immediate decision successors for D w.r.t. the decision option $\mathsf{X} \in \mathrm{sp}_{\mathsf{D}}$. The set of *candidate strategies* for D is then given by:

$$\Delta_{\mathsf{D}}^{+} = \bigcup_{\mathsf{X} \in \mathrm{sp}_{\mathsf{D}}^{*}} \left\{ \{(\mathsf{D},\mathsf{X})\} \cup \mathsf{S} \mid \mathsf{S} \in \prod_{\mathsf{D}' \in \mathcal{D}_{(\mathsf{D},\mathsf{X})}} \Delta_{\mathsf{D}'} \right\},$$

where $\mathrm{sp}_{\mathsf{X}}^{*} \subseteq \mathrm{sp}_{\mathsf{X}}$ s.t. for all $\mathsf{X} \in \mathrm{sp}_{\mathsf{X}}^{*}$ it holds that every $\mathsf{D}' \in \mathcal{D}_{(\mathsf{D},\mathsf{X})}$ is associated with a non-empty set of strategies.

The algorithm outlined above is formalized as follows.

Algorithm 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* Evaluation *(Algorithm 2) on T w.r.t. θ and κ and let Δ_{D} be the set of strategies returned.*
3. **If** $\Delta_{\mathsf{D}} \neq \emptyset$ **then**
   Return $\arg\max_{\Delta \in \Delta_{\mathsf{D}}} \mathsf{V}(\Delta)$.
   else
   No solution exists for T w.r.t. κ and θ.

Algorithm 2 (Evaluation). *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* Evaluation *on D', $\forall \mathsf{D}' \in \mathcal{D}_{\mathsf{D}}$.*
2. *Let Δ_{D}^{+} be the set of candidate strategies for D in $\mathsf{T}(\mathsf{D})$ (see Definition 4).*
3. **For** *each chance node $\mathsf{A} \in \mathcal{C}_{\mathsf{D}}$*
   For *each $\mathsf{B} \in sp_{\mathsf{A}}$*
   Multiply $\mathsf{P}((\mathsf{A},\mathsf{B})|past(\mathsf{A}))$ with the probabilities associated with the consequences in $\mathsf{T}(\mathsf{B})$.
4. **For** *each strategy $\Delta \in \Delta_{\mathsf{D}}^{+}$*
   If *Δ is non-dominated in $\mathsf{T}(\mathsf{D})$ (see Section 3)* **then**
   a) Calculate $\mathsf{V}(\Delta)$.
   b) Set $\Delta_{\mathsf{D}} := \Delta_{\mathsf{D}} \cup \{\Delta\}$ and let $\mathsf{V}_{\mathsf{D}}^{\max} = \max_{\Delta \in \Delta_{\mathsf{D}}} \mathsf{V}(\Delta)$.
5. **For** *each strategy $\Delta \in \Delta_{\mathsf{D}}$*
   If $\mathsf{V}(\Delta) < \mathsf{V}_{\mathsf{D}}^{\max} - \theta$ **then**
   Set $\Delta_{\mathsf{D}} := \Delta_{\mathsf{D}} \backslash \{\Delta\}$
6. **If** $|\Delta_{\mathsf{D}}| > \kappa$ **then**
   repeat
   Set $\Delta_{\mathsf{D}} := \Delta_{\mathsf{D}} \backslash \{\Delta\}$, where $\Delta = \arg\min_{\Delta \in \Delta_{\mathsf{D}}} \mathsf{V}(\Delta)$.
   Until $|\Delta_{\mathsf{D}}| = \kappa$
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. $\mathsf{V}(\Delta)$. Furthermore, it should be noticed that step (3) ensures that all probabilities needed to calculate $\mathsf{V}(\Delta)$ in step 4a are available. Hence, to calculate $\mathsf{V}(\Delta)$ 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.

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.[2] Since the model is proposed as an alternative to EU-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 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:

- EU model, it is assigned the sum of the values associated with its children weighted by the probabilities of their outcomes, i.e., $\sum_{\mathsf{X}\in \mathrm{sp}_{\mathsf{A}}} \mathsf{P}((\mathsf{A},\mathsf{X})|\mathrm{past}(\mathsf{A})) \cdot \mathsf{EU}(\mathsf{X})$.
- non-EU model, the probability $\mathsf{P}((\mathsf{A},\mathsf{X})|\mathrm{past}(\mathsf{A}))$ is multiplied with the probability (initially 1) associated with the consequences in the subtree $\mathsf{T}(\mathsf{X})$, for each $\mathsf{X} \in \mathrm{sp}_{\mathsf{A}}$.

When a decision node D is visited in the:

- EU model, it is associated with the decision option leading to the child having the highest expected utility; the decision node is assign this value as well.
- non-EU 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 $\mathcal{N}_{\mathsf{D}}$ we obtain a system of inequalities where the:

- Number of variables $= |\mathcal{N}_{\mathsf{D}}| + |\mathcal{C}|$.
- Number of inequalities $\leq |\mathcal{N}_{\mathsf{D}}| \cdot \mathrm{sp}_{\max} + |\mathcal{C}|$; $\mathrm{sp}_{\max}$ is the maximum number of children for a decision node.

[2] Note that the algorithm is also applicable in case we work with the *coalesced decision tree*[Olmsted, 1983], i.e., the graph formed from the decision tree by collapsing identical subtrees; two subtrees are identical if they agree on both structure and numeric information. 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.

A solution to these inequalities can be found by e.g. employing a simplex algorithm or an interior point algorithm. The simplex algorithm is bounded by the number of *basic solutions* which is given by:

$$\text{Upper bound} = \frac{(m+n)!}{m!n!},$$

where n is the number of variables and m is the number of inequalities.

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[3] in the subtree $\mathsf{T}(\mathsf{D})$ and $\left|\mathcal{C}^{\mathsf{T}(\mathsf{D})}\right|$ is the maximum number of possible consequence for a strategy in $\mathsf{T}(\mathsf{D})$, then we have:

$$\text{Relative complexity} = \frac{\mathsf{O}\left(\sum_{\mathsf{D}\in\mathcal{N}_{\mathsf{D}}}\left[\kappa^{\omega}\cdot|\mathrm{sp}_{\mathsf{D}}|\cdot\left(\#_{\mathsf{T}(\mathsf{D})}+\left|\mathcal{C}^{\mathsf{T}(\mathsf{D})}\right|+1\right)\right]+\sum_{\mathsf{C}\in\mathcal{N}_{\mathsf{C}}}\left|\mathcal{C}^{\mathsf{T}(\mathsf{D})}\right|\right)}{\mathsf{O}\left(\sum_{\mathsf{D}\in\mathcal{N}_{\mathsf{D}}}|\mathrm{sp}_{\mathsf{D}}|+\sum_{\mathsf{C}\in\mathcal{N}_{\mathsf{C}}}|\mathrm{sp}_{\mathsf{C}}|\right)}$$

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. For instance, most people are usually almost indifferent between receiving 1000$ or 1005$ thus, these consequences should not be given different utilities.

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 an RDU maximizer. In the second set of tests, the algorithm proposed in Section 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 in relation to the DM and is therefore specific to the decision problem (and the DM) 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 6-8, 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.

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

[3] When using the interior point algorithm the non-dominance test runs in polynomial time.

[4] *PCx* is an interior-point predictor-corrector linear programming package which can be downloaded at *http://www-fp.mcs.anl.gov/otc/Tools/PCx/index.html*.

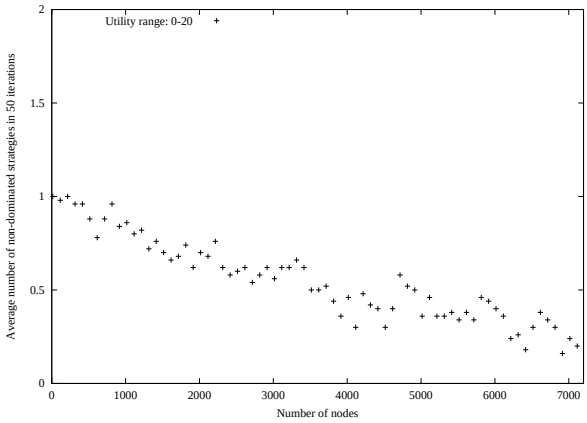


Figure 6: The average number of non-dominated sophisticated strategies as a function of the number of chance nodes and decision nodes in the decision tree. The utility levels varied between 0 and 20.

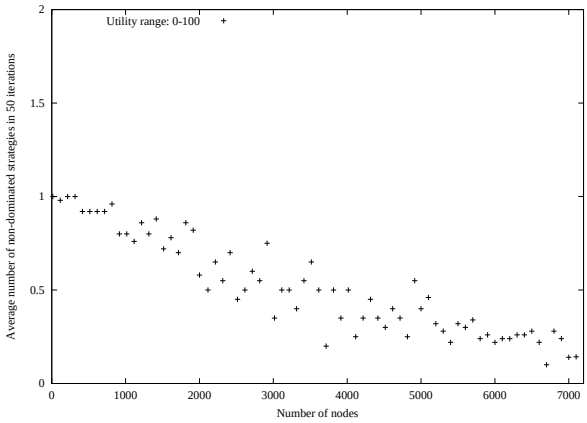


Figure 7: The average number of non-dominated sophisticated strategies as a function of the number of chance nodes and decision nodes in the decision tree. The utility levels varied between 0 and 100.

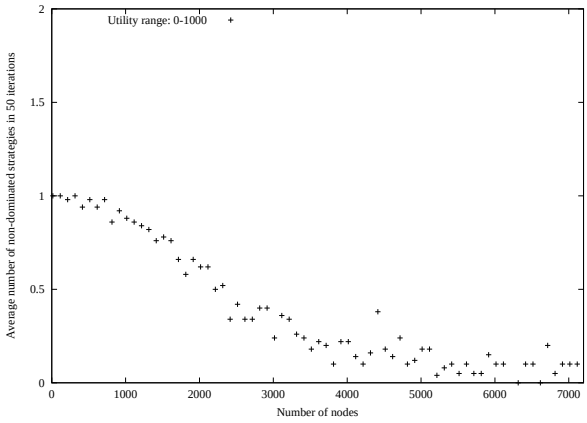


Figure 8: The average number of non-dominated sophisticated strategies as a function of the number of chance nodes and decision nodes in the decision tree. The utility levels varied between 0 and 1000.

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 9-11.

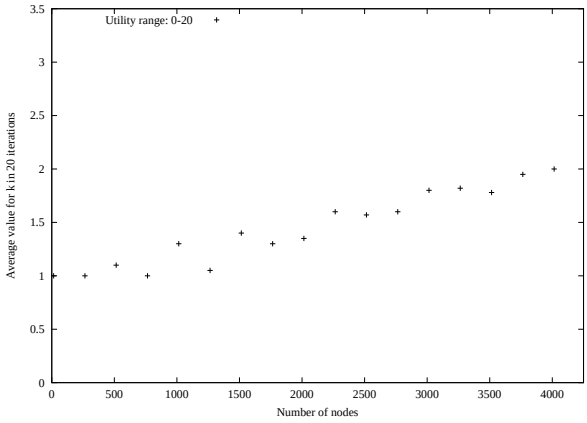


Figure 9: The average value of κ which yields a non-dominated strategy as a function of the number of chance nodes and decision nodes in the decision tree. The utility levels varied between 0 and 20.

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 the range of the utility function.

6 Discussion

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.

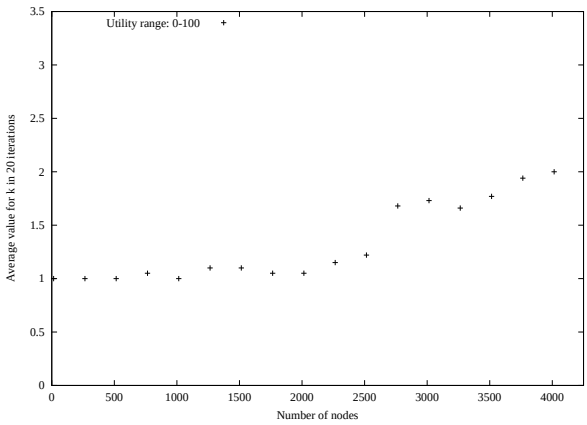


Figure 10: The average value of κ which yields a non-dominated strategy as a function of the number of chance nodes and decision nodes in the decision tree. The utility levels varied between 0 and 100.

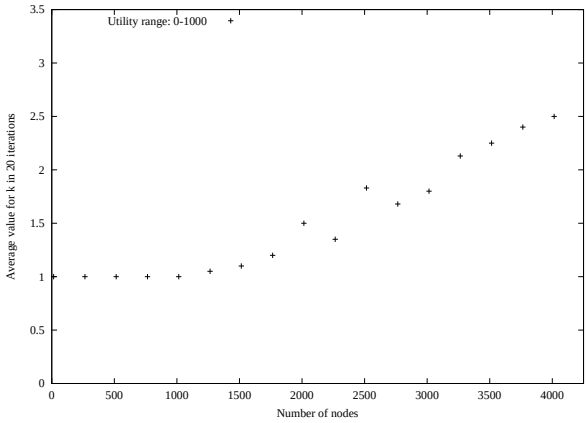


Figure 11: The average value of κ which yields a non-dominated strategy as a function of the number of chance nodes and decision nodes in the decision tree. The utility levels varied between 0 and 1000.

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 5.

6.2 Testing for stochastic dominance of the second order

The investigation of strategies can easily be modified for investigating strategies w.r.t. stochastic dominance of the second order (see Definition 3). In this context, any proposed strategy Δ (represented by a lottery L_{Δ}) must satisfy:

$$\begin{aligned} \forall \Delta' \neq \Delta, \text{ either } \exists u \in \mathcal{U} \text{ s.t. } EU(L_{\Delta})_u > EU(L_{\Delta'})_u \\ \text{or } \forall u \in \mathcal{U} \text{ s.t. } EU(L_{\Delta})_u \geq EU(L_{\Delta'})_u, \end{aligned}$$

where $\mathcal{U} = \{u | [u \text{ is concave}] \text{ and } [\text{if } c_i \prec c_j \text{ then } u(c_i) < u(c_j)]\}$. 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 FSD that is, we look for the strategies for which there exists a utility function s.t. the strategy in question is strictly EU-optimal:

$$\exists u \in \mathcal{U} \text{ s.t. } EU(L_{\Delta})_u > EU(L_{\Delta'})_u, \forall \Delta' \neq \Delta$$

Similarly to FSD, 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:

$$u(c_{i+1}) - u(c_i) \leq [u(c_i) - u(c_{i-1})] \left(\frac{c_{i+1} - c_i}{c_i - c_{i-1}} \right) \tag{6}$$

By extending the system of linear inequalities (as defined in Section 3) with Inequality 6 ($\forall 2 \leq i \leq n-1$) we can investigate strategies w.r.t. to this stricter notion of SSD.

Moreover, this extension does not compromise the investigation of strategies through the modified version of the simplex algorithm, as suggested in Appendix A.

7 Conclusion

In this paper we have presented an operational approach to decision aiding based on non-expected utility. The approach was motivated by the fact that DMs sometimes reject proposed strategies

(or models) based on EU theory, due to the inability of EU theory to account for experimentally well-established certainty/possibility effects. Methods have been proposed to circumvent these shortcomings. For instance, the DM could pursue e.g. 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 DM 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 paper is based on the model by [Jaffray, 1999]. The non-EU criterion attributed to the DM is Rank Dependent Utility [Quiggin, 1982], and the common goal among the Selves is to produce a non-dominated strategy. The RDU decision is non-separable meaning that in general, it fails to support backward induction. However, under the assumption that the DM 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.

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 at producing a non-dominated strategy, our algorithm manages to exhibit one for reasonable values of the model parameters.

Acknowledgement

This research was performed while Thomas D. Nielsen was associated with Université Paris 6, France as a visiting scholar; in the same period Thomas D. Nielsen was a Ph.D.-student at the Department of Computer Science, Aalborg University, Denmark. The research was partly supported by CIT project number 87.2 - "Automated Decision Support for Customer Support Operations - Phase 2".

A Testing for dominance

In this appendix we illustrate how the structure of a linear program, generated by a decision tree according to Section 3, can be exploited when testing for dominance.

Before describing how this can be done 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 satisfy 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

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n + t_i = b_i \tag{7}$$

Rewriting Equation 7 in terms of $-t_i$ gives:

$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n - b_i = -t_i \tag{8}$$

The set of equalities produced by adding slack variables can be organized in a *tableau*. We shall refer to the tableau consisting of Equation 8 ($\forall 1 \leq i \leq m$) as the *initial tableau* (see Table 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}	$-b_2$	$= -t_2$
$\vdots$	...	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\vdots$
a_{i1}	a_{i2}	$\cdots$	a_{ij}	$\cdots$	a_{in}	$-b_i$	$= -t_i$
$\vdots$	...	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\vdots$
a_{m1}	a_{m2}	$\cdots$	a_{mj}	$\cdots$	a_{mn}	$-b_m$	$= -t_m$

Table 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 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.

p	$\cdots$	q	$\Longrightarrow$	$\frac{1}{p}$	$\cdots$	$\frac{q}{p}$
$\vdots$	$\ddots$	$\vdots$		$\vdots$	$\ddots$	$\vdots$
r	$\cdots$	s		$\frac{-r}{p}$	$\cdots$	$s - \frac{rq}{p}$

Table 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 3. *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 5. 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 for investigating strategies

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 4 results in:

$$y_D = \sum_{l \in \mathcal{L}} p_l y_l + \epsilon + t_D \Leftrightarrow -t_D = \sum_{l \in \mathcal{L}} p_l y_l + \epsilon - y_D \tag{9}$$

for $t_D \geq 0$. Similarly,

$$y_D = y_x + \epsilon + t_D \Leftrightarrow -t_D = y_x + \epsilon - y_D \tag{10}$$

Finally, Inequality 5 gives:

$$y_D + t_D = \sum_{l \in \mathcal{L}} p_l y_l \Leftrightarrow -t_D = y_D - \left(\sum_{l \in \mathcal{L}} p_l y_l \right) \tag{11}$$

Given a decision node D the set of equalities described by Equality 9, Equality 10 and Equality 11 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 5. Consider Figure 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 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	[-1]	α	$-t_6$

Table 3: The initial tableau.

Making a pivot transformation w.r.t. the entry in row 7 we get Table 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	[-1]		α	$-t_5$
					−1	−1	$-\alpha$	$-c_6$

Table 4: The tableau obtained from Table 3 after pivot transformation.

Making a pivot transformation w.r.t. the entry in row 6 we get Table 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	[-1]			α	$-t_4$
				−1	−1		$-\alpha$	$-c_5$
				−1	−1	−1	-2α	$-c_6$

Table 5: The tableau obtained from Table 4 after pivot transformation.

Making a pivot transformation w.r.t. the entry in row 5 we get Table 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	[-1]				α	$-t_3$
			−1	−1			$-\alpha$	$-c_4$
			−1	−1	−1		-2α	$-c_5$
			−1	−1	−1	−1	-3α	$-c_6$

Table 6: The tableau obtained from Table 5 after pivot transformation.

Making a pivot transformation w.r.t. the entry in row 4 we get Table 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	-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 7: The tableau obtained from Table 6 after pivot transformation.

Making a pivot transformation w.r.t. the entry in row 3 we get Table 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	$-r$
-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 8: The tableau obtained from Table 7 after pivot transformation.

Making a pivot transformation w.r.t. the entry in row 2 we get Table 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	-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 9: The tableau obtained from Table 8 after pivot transformation.

The relevant influence by the pivot transformation specified by Table 9 is restricted to the entry containing -5α (producing $-8\alpha - 0.5\epsilon$ 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 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 12).

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 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.

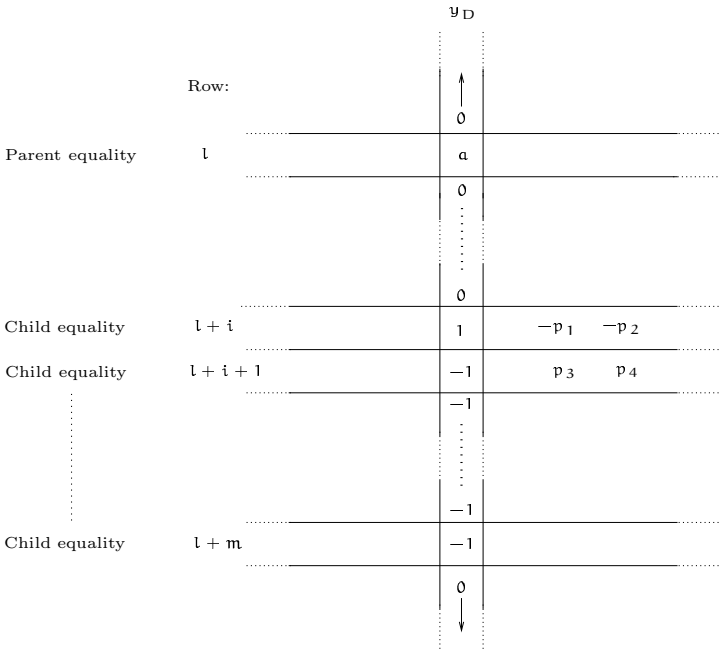


Figure 12: The figure illustrates the ordering of the rows in the initial tableau.

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 3-8 in Example 5 for illustration). After resolving these equalities we obtain the general situation outlined in Figure 12; actually Figure 12 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 12 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 positive entry in the $-b$ column for each row $j(l+i+1 \leq j \leq l+m)$. Furthermore, according to Algorithm 3 the -1 entry in the $(l+m)$'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 $l+i$. 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 $\mathtt{l}+\mathtt{i}$ is resolved by the pivot transformations made w.r.t. to the entries in the rows physically below row $\mathtt{l}+\mathtt{i}$, then we can continue the pivot transformations as described above. So, assume that row $\mathtt{l}+\mathtt{i}$ was not resolved by these pivot transformations. In general, this row does not contain a -1 entry in the $\mathtt{y}_{\mathsf{D}}$ column thus, we need to identify another negative entry (let $-\mathtt{a}$ 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 $\mathtt{a}'$, physically below $-\mathtt{a}$. So, we should in principle investigate all entries physically below $-\mathtt{a}$. However, suppose that we choose the negative entry $-\mathtt{a}$ 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 $\mathtt{a}$-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 $\mathtt{l}+\mathtt{i}$ only occurs because D was assumed to be part of Δ that is, if D is not part of Δ then row $\mathtt{l}+\mathtt{i}$ would not be present and we would only need to make pivot transformations w.r.t. the -1 entries in the $\mathtt{y}_{\mathsf{D}}$ column.

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. Anyhow we may still obtain a substantial reduction in the computational complexity.

A.3 Complexity

In what follows we consider how the modifications to the simplex algorithm impact on the complexity.

Consider solving a system of linear inequalities as described above. Given a decision tree containing $\mathcal{N}_{\mathsf{D}}$ decision nodes, we obtain an initial tableau where:

- Number of columns $= |\mathcal{N}_{\mathsf{D}}| + |\mathcal{C}|$.
- Number of rows $\leq |\mathcal{N}_{\mathsf{D}}| \cdot \mathrm{sp}_{\max} + |\mathcal{C}|$; $\mathrm{sp}_{\max}$ is the maximum number of children for a decision node.

If $\mathcal{N}_{\mathsf{D}}^{\Delta} \subseteq \mathcal{N}_{\mathsf{D}}$ is the set of decision nodes contained in the strategy Δ, then according to the section above, we make one (and only one) pivot transformation for i) each child equality associated with a node in $\mathcal{N}_{\mathsf{D}} \backslash \mathcal{N}_{\mathsf{D}}^{\Delta}$ and ii) each equality associated with a node in $\mathcal{C}$. That is, we need at most $\left(\mathrm{sp}_{\max} \cdot |\mathcal{N}_{\mathsf{D}} \backslash \mathcal{N}_{\mathsf{D}}^{\Delta}|\right) + |\mathcal{C}|$ pivot transformations to resolve the equalities associated with these nodes. For the set of nodes $\mathcal{N}_{\mathsf{D}}^{\Delta}$, we make one (and only one) pivot transformation for each arc not included in Δ that is, we need at most $(\mathrm{sp}_{\max} - 1) \cdot |\mathcal{N}_{\mathsf{D}}^{\Delta}|$ pivot transformations to resolve these equalities.

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 $\mathtt{l}$ we can perceive row $\mathtt{l}$ together with the $\mathtt{m}-\mathtt{l}$ rows physically below row $\mathtt{l}$ as a (separate) system of inequalities.

References

[Allais, 1979] 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 *Expected utility hypotheses*

and the Allais paradox, pages 27–145. Dordrecht, Holland. Original work published in 1952.

[Allais, 1988] Allais, M. (1988). The general theory of random choices in relation to the invariant cardinal utility function and the specific probability function. *Risk, Decision and Rationality*, pages 233–289. Dordrecht, Holland.

[Chajewska and Koller, 2000] Chajewska, U. and Koller, D. (2000). Utilities as Random Variables: Density Estimation and Structure Discovery. In Boutilier, C. and Goldszmidt, M., editors, *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, pages 63–71.

[Chateauneuf and Cohen, 1984] Chateauneuf, A. and Cohen, M. (1984). Risk seeking with diminishing marginal utility in a non-expected utility model. *Journal of Risk and Uncertainty*, 9:77–91.

[Chew et al., 1987] Chew, S., Karni, E., and Safra, Z. (1987). Risk aversion in the theory of expected utility with rank dependent preferences. *Journal of Economic Theory*, 42:370–381.

[Choquet, 1953] Choquet, G. (1953). Theory of capacities. *Annales institut Fourier*, 5.

[Currim and Sarin, 1989] Currim, I. S. and Sarin, R. K. (1989). Prospect Versus Utility. *Management Science*, 35(1):22–41.

[Dantzig and Wolfe, 1960] Dantzig, G. B. and Wolfe, P. (1960). Decomposition principle of linear programs. *Operations Research*, 8(1):101–111.

[Hadar and Russel, 1969] Hadar, J. and Russel, W. (1969). Rules for ordering uncertain prospects. *American economic review*, 59(1):97–122.

[Jaffray, 1999] Jaffray, J.-Y. (1999). Rational decision making with imprecise probabilities. In 1*st International Symposium on Imprecise Probabilities and Their Applications*, pages 183–188. Morgan Kaufmann Publishers.

[Kahneman and Tversky, 1979] Kahneman, D. and Tversky, A. (1979). Prospect Theory: An Analysis of Decision under Risk. *Econometrica*, 47:263–291.

[Karmarkar, 1984] Karmarkar, N. (1984). A new polynomial-time algorithm for linear progamming. *Combinatorica*, 4:373–395.

[Karni and Safra, 1989] Karni, E. and Safra, Z. (1989). Ascending Bid Auctions With Behaviorally Consistent Bidders. *Annals of Operations Research*, 19:435–446.

[Machina, 1989] Machina, M. J. (1989). Dynamic consistency and non-expected utility models of choice under uncertainty. *Journal of Economic Literature*, 27:1622–1688.

[McClennen, 1990] McClennen, E. F. (1990). *Rationality and Dynamic choice: Foundational explorations.* Cambridge University Press, Cambridge.

[McCord and de Neufville, 1984] McCord, M. R. and de Neufville, R. (1984). Utility dependence on probability: an empirical demonstration. *Journal of Large Scale Systems*, 6:91–103.

[Olmsted, 1983] Olmsted, S. M. (1983). *On representing and solving decision problems.* PhD thesis, Department of Engineering-Economic Systems, Stanford University.

[Quiggin, 1982] Quiggin, J. (1982). A theory of anticipated utility. *Journal of Economic Behavior and Organisation*, 3(4):323–343.

[Raiffa and Schlaifer, 1961] Raiffa, H. and Schlaifer, R. (1961). *Applied Statistical Decision Theory.* MIT press, Cambridge.

[Rothenberg, 1979] Rothenberg, R. I. (1979). *Linear Programming.* Elsevier North Holland. ISBN: 0-444-00325-8.

[Schick, 1986] Schick, F. (1986). Dutch Bookies and Money Pumps. *Journal of Philosophy*, 83:112–119.

[Schrijver, 1986] Schrijver, A. (1986). *Theory of Linear and Integer Programming*. Wiley, N-Y.

[Tucker, 1964] Tucker, A. W. (1964). Combinatorial algebra of matrix games and linear programs. In Beckenbach, E. F., editor, *Applied Combinatorial Mathematics*, chapter 11, pages 320–347. New York: John Wiley and Sons.

[Wakker, 1998] Wakker, P. P. (1998). Non-expected utility as aversion of information. *Journal of Behavioral decision making*, 1:169–175.

[Wu and Gonzales, 1996] Wu, G. and Gonzales, R. (1996). Curvature of the Probability Weighting Function. *Management Science*, 42(12):1676–1690.

[Yaari, 1987] Yaari, M. (1987). The dual thoery of choice under risk. *Econometrica*, 55(1):95–115.