THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*
Cotutelle internationale *avec l'Institut Supérieur de Gestion de Tunis*

Présentée et soutenue le *31 Octobre 2017* **par :**

Zeineb EL KHALFI

Lexicographic Refinements in Possibilistic Sequential Decision-Making Models

JURY

Nahla BEN AMOR	Professeur à l'Institut Supérieur de Gestion de Tunis	Directrice de thèse
Didier DUBOIS	Directeur de Recherche CNRS à l'Institut de Recherche en Informatique de Toulouse	Examinateur
Zied ELOUEDI	Professeur à l'Institut Supérieur de Gestion de Tunis	Examinateur
Hélène FARGIER	Directeur de Recherche CNRS à l'Institut de Recherche en Informatique de Toulouse	Directrice de thèse
Jérôme LANG	Directeur de Recherche CNRS à Paris-Dauphine	Rapporteur
Régis SABBADIN	Directeur de Recherche à l'Institut National de la Recherche Agronomique de Toulouse	Invité
Olivier Spanjaard	Maître de Conférences à l'Université Pierre et Marie Curie	Rapporteur

École doctorale et spécialité :
MITT : Domaine STIC : Intelligence Artificielle
Unité de Recherche :
Institut de Recherche en Informatique de Toulouse (UMR 5505)
Directrice(s) de Thèse :
Hélène FARGIER et *Nahla BEN AMOR*
Rapporteurs :
Jérôme LANG et *Olivier Spanjaard*

This thesis is dedicated to my beloved parents, who always believed in me

Remerciements

À la fin de la rédaction de cette thèse, je suis convaincue que la thèse est loin d'être un travail solitaire. En effet, je n'aurais jamais pu réaliser ce travail sans le soutien d'un grand nombre de personnes dont la bonne humeur et l'intérêt manifestés à l'égard de ma thèse m'ont permis de progresser dans cette phase délicate.

Je voudrais tout d'abord remercier grandement mes directrices de thèse, Nahla Ben Amor et Hélène Fargier, pour toutes leurs aides. Je suis ravie d'avoir travaillé en leur compagnie car outre leurs appuis scientifiques, elles ont toujours été là pour me soutenir et me conseiller au cours de l'élaboration de cette thèse. Leur contact a d'ailleurs été très enrichissant tant au niveau humain qu'au niveau professionnel. Il peut être assuré de mon sincère respect et de ma profonde gratitude. Je remercie également Régis Sabaddin avec qui j'ai eu la chance de pouvoir travailler. Sa rigueur, sa capacité d'analyse des problèmes et ses très nombreuses connaissances m'ont permis de progresser et de répondre à plusieurs de mes préoccupations. Les conseils et le soutien reçus étaient essentiels à la réussite du projet.

Je tiens à remercier Jérome Lang et Olivier Spanjaard qui m'ont fait l'honneur d'être rapporteurs de ma thèse. Leurs remarques m'ont permis de voir mon travail sous un autre angle. J'adresse aussi mes remerciements à Didier Dubois et Zied Elouadi pour avoir accepté de participer à mon jury de thèse et pour leurs apports scientifiques et l'honneur qu'ils me font d'être dans mon jury thèse.

Je remercie également l'ensemble des membres de l'IRIT et du LARODEC pour leur soutien logistique et moral ainsi que pour la très bonne ambiance que j'ai toujours trouvée. Je remercie particulièrement : Fayçal, Maroua, Pierre-François, Wafa, Ismail, Rami, Sleh, Manel...

Heureusement que mes parents et mes soeurs sont là pour me changer les idées. Ils ont tous cru en moi et maintenant j'y suis ! Alors merci à vous tous : mon père Abdrahmen et ma mère Aicha, mes soeurs Emna, Sabra et Rahma, nos petits bonheurs zaydoun, ayouta et lili. Un grand merci à mes amies - chercheuses Héla, Fatma et Sabrine, elles m'ont beaucoup aidé et sont devenues des amies à qui je souhaite tout le courage qu'elles m'ont apporté. Merci à ma chère Marwa pour sa gentillesse et les délicieux plats tunisiens made in Toulouse, et finalement à mon très cher ami et frère Moez, il a toujours fait tout son possible pour m'aider.

Je tiens également à remercier mon cher époux pour son soutien quotidien indéfectible et son enthousiasme contagieux à l'égard de mes travaux comme de la vie en général. Notre couple a grandi en même temps que mon projet scientifique, le premier servant de socle solide à l'épanouissement du second.

Ces remerciements ne peuvent s'achever, sans une pensée pour ma première fan : ma mère Aicha. Sa présence et ses encouragements sont pour moi les piliers fondateurs de ce que je suis et de ce que je fais.

Encore un grand merci à tous pour m'avoir conduit à ce jour mémorable.

Résumé

Ce travail contribue à la théorie de la décision possibiliste et plus précisément à la prise de décision séquentielle dans le cadre de la théorie des possibilités, à la fois au niveau théorique et pratique. Bien qu'attrayante pour sa capacité à résoudre les problèmes de décision qualitatifs, la théorie de la décision possibiliste souffre d'un inconvénient important: les critères d'utilité qualitative possibilistes comparent les actions avec les opérateurs min et max, ce qui entraîne un effet de noyade. Pour surmonter ce manque de pouvoir décisionnel, plusieurs raffinements ont été proposés dans la littérature. Les raffinements lexicographiques sont particulièrement intéressants puisqu'ils permettent de bénéficier de l'arrière-plan de l'utilité espérée, tout en restant «qualitatifs». Cependant, ces raffinements ne sont définis que pour les problèmes de décision non séquentiels.

Dans cette thèse, nous présentons des résultats sur l'extension des raffinements lexicographiques aux problèmes de décision séquentiels, en particulier aux Arbres de Décision et aux Processus Décisionnels de Markov possibilistes. Dans un premier temps, nous présentons des relations de préférence lexicographiques optimistes et pessimistes entre les politiques avec et sans utilités intermédiaires, qui raffinent respectivement les utilités possibilistes optimistes et pessimistes. Nous prouvons que les critères proposés satisfont le principe de l'efficacité de Pareto ainsi que la propriété de monotonie stricte. Cette dernière garantit la possibilité d'application d'un algorithme de programmation dynamique pour calculer des politiques optimales. Nous étudions tout d'abord l'optimisation lexicographique des politiques dans les Arbres de Décision possibilistes et les Processus Décisionnels de Markov à horizon *fini*. Nous fournissons des adaptations de l'algorithme de programmation dynamique qui calculent une politique optimale en temps polynomial. Ces algorithmes sont basés sur la comparaison lexicographique des matrices de trajectoires associées aux sous-politiques. Ce travail algorithmique est complété par une étude expérimentale qui montre la faisabilité et l'intérêt de l'approche proposée. Ensuite, nous prouvons que les critères lexicographiques bénéficient toujours d'une fondation en termes d'utilité espérée, et qu'ils peuvent être capturés par des utilités espérées infinitésimales.

La dernière partie de notre travail est consacrée à l'optimisation des politiques dans les Processus Décisionnels de Markov (éventuellement *infinis*) stationnaires. Nous proposons un algorithme d'itération de la valeur pour le calcul des politiques optimales lexicographiques. De plus, nous étendons ces résultats au cas de l'horizon infini. La taille des matrices augmentant exponentiellement (ce qui est particulièrement problématique dans le cas de l'horizon infini), nous proposons un algorithme d'approximation qui se limite à la partie la plus intéressante de chaque matrice de trajectoires, à savoir les premières lignes et colonnes. Enfin, nous rapportons des résultats expérimentaux qui prouvent l'efficacité des algorithmes basés sur la troncation des matrices.

Mots clefs: Décision séquentielle, théorie de possibilités, critères lexicographiques, arbres de décision, processus décisionnels de Markov

Abstract

This work contributes to possibilistic decision theory and more specifically to sequential decision-making under possibilistic uncertainty, at both the theoretical and practical levels. Even though appealing for its ability to handle qualitative decision problems, possibilisitic decision theory suffers from an important drawback: qualitative possibilistic utility criteria compare acts through min and max operators, which leads to a drowning effect. To overcome this lack of decision power, several refinements have been proposed in the literature. Lexicographic refinements are particularly appealing since they allow to benefit from the expected utility background, while remaining "qualitative". However, these refinements are defined for the non-sequential decision problems only.

In this thesis, we present results on the extension of the lexicographic preference relations to sequential decision problems, in particular, to possibilistic Decision trees and Markov Decision Processes. We first present optimistic and pessimistic lexicographic preference relations between policies with and without intermediate utilities that refine the optimistic and pessimistic qualitative utilities respectively. We prove that these new proposed criteria satisfy the principle of Pareto efficiency as well as the property of strict monotonicity. This latter guarantees that dynamic programming algorithm can be used for calculating lexicographic optimal policies. Considering the problem of policy optimization in possibilistic decision trees and *finite*-horizon Markov decision processes, we provide adaptations of dynamic programming algorithm that calculate lexicographic optimal policy in polynomial time. These algorithms are based on the lexicographic comparison of the matrices of trajectories associated to the sub-policies. This algorithmic work is completed with an experimental study that shows the feasibility and the interest of the proposed approach. Then we prove that the lexicographic criteria still benefit from an Expected Utility grounding, and can be represented by infinitesimal expected utilities.

The last part of our work is devoted to policy optimization in (possibly *infinite*) stationary Markov Decision Processes. We propose a value iteration algorithm for the computation of lexicographic optimal policies. We extend these results to the infinite-horizon case. Since the size of the matrices increases exponentially (which is especially problematic in the infinite-horizon case), we thus propose an approximation algorithm which keeps the most interesting part of each matrix of trajectories, namely the first lines and columns. Finally, we reports experimental results that show the effectiveness of the algorithms based on the cutting of the matrices.

Keywords: Sequential decision theory, possibility theory, lexicographic criteria, decision trees, Markov decision processes

Contents

List of Figures

List of Tables

List of Algorithms

INTRODUCTION

In classical decision-making under uncertainty frameworks, one has to select an action among several alternatives according to Expected utility [Neumann and Morgenstern, 1944, Savage, 1954], considering that uncertainty is modeled via a probability distribution.

When the decisions are spread over time, a decision maker can choose between several punctual actions with incapacity to predict, with certainty, the outcome event. This problem of sequential decision-making under uncertainty exists in multiple domains such as automatic control, robot control, medical diagnosis, in-time stock management, modeling games etc. Several representation formalisms can be used for sequential decision problems, such as influence diagrams [Howard and Matheson, 1984], Markov decision processes [Bellman, 1957b] and decision trees [Raiffa, 1968] etc.

Despite its success, the expected utility framework presents some limitations concerning the representation of total ignorance and the handling of qualitative information. It is appropriate when all probabilities are available or can be easily elicited from the decision-maker, which is not always possible. Several generalizations or alternatives to classical probability theory have been suggested in order to deal with imperfect information, including imprecise probabilities [Walley, 1991], evidence theory [Shafer, 1976] and possibility theory [Zadeh, 1978]. In the present work, we are interested in possibility theory that offers a natural and simple framework to handle qualitative information, and especially in possibilistic qualitative decision theory [Giang and Shenoy, 2005, Weng, 2005, Dubois et al., 2001b, Godo and Zapico, 2001, Dubois et al., 1998a, Dubois and Prade, 1995, Godo and Zapico, 2001]. The development of possibilistic decision theory has led to the proposition of a series of *possibilistic decision criteria*, and in particular: optimistic and pessimistic possibilistic qualitative criteria (the qualitative counterparts of expected utility) [Dubois and Prade, 1995], possibilistic likely dominance [Dubois et al., 2003], binary possibilistic utility [Giang and Shenoy, 2001] and possibilistic Choquet integrals [Rebille, 2006, Dubois and Rico, 2016a].

Possibilistic (qualitative) decision theory is relevant, among other fields, for applications to

sequential decision-making under uncertainty, where a suitable policy (i.e. a set of actions) is to be found w.r.t. a qualitative decision criterion, starting from a qualitative description of the initial world, of the available actions, of their uncertain effects and of the goal to reach (see e.g. [Bauters et al., 2016, Ben Amor et al., 2014, Drougard et al., 2014, Drougard et al., 2013, Sabbadin, 2001, Sabbadin et al., 1998]). It is important to note that, in compact sequential decision models, like Markov decision processes and influence diagrams, the set of potential policies is combinatorial and it may grow exponentially. The computation of an optimal policy for a given representation and a given decision criterion is an algorithmic issue. In this thesis, we are interested to the optimization of the possibilistic counterparts of *decision trees* and *Markov decision processes*, which assume that the uncertain effects of actions can be represented by possibility distributions and that utilities are qualitative.

Even though appealing for its ability to handle qualitative problems, possibilistic decision theory suffers from an important drawback: the lack of discrimination power of its decision criteria. As many of possibilistic decision criteria, optimistic and pessimistic qualitative utilities, compare acts, and policies in sequential decision problems, through $\min$ and $\max$ operators, which leads to a *drowning effect*: plausible enough bad or good consequences may blur the comparison between acts that would otherwise be clearly differentiable. As a consequence, it appears that these criteria do not respect the Pareto efficiency and the sure thing principle.

To overcome the lack of decision power of possibility theory, several refinements of possibilistic decision criteria have been proposed for the non-sequential case decision problems [Dubois et al., 2000, Giang and Shenoy, 2001, Fargier and Sabbadin, 2003, Fargier and Sabbadin, 2005, Weng, 2005]. In particular, [Fargier and Sabbadin, 2003, Fargier and Sabbadin, 2005] have proposed lexicographic refinements of possibilistic qualitative utilities that are appealing since they allow to benefit from the Expected Utility background, while remaining "qualitative". However, the latter is limited to non-sequential decision problems, and cannot take into account the fact that the drowning effect can also appear due to the reduction of compound possibilistic policies into simple possibility distributions on the consequences.

The aim of this thesis is to study and solve the problem of drowning effect in sequential decision problems, in particular in possibilistic decision trees and possibilistic Markov decision processes. We propose to extend the lexicographic refinements of qualitative utilities to sequential problems-thus providing lexicographic possibilistic decision criteria that compare full policies (and not simply their reductions).

This thesis is decomposed into two main parts:

The first part offers necessary background on sequential decision-making under uncertainty in various aspects:

1. Chapter 1 recalls the expected utility decision model and introduces the basic concepts relative to decision theory. We present possibility theory and we detail optimistic and pessimistic utilities as well as the shortcoming of these criteria. Then we evoke their refine-

ments in non-sequential decision-making: first, we present a brief overview of approaches that try to remedy the drowning effect problem, then EU-based refinements and lexicographic refinements are developed.

2. Chapter 2 is devoted to sequential decision-making models. We especially focus on possibilistic decision trees and possibilistic Markov decision processes. These graphical models are presented in the possibilistic (qualitative) version and their solving algorithms w.r.t. qualitative criteria are described.

The second part of the thesis represents our main contributions. It is structured as follows:

1. Chapter 3 presents the drowning effect in possibilistic decision trees, finite-horizon Markov decision processes and also in stationary Markov decision processes. We then define lexicographic refinements of possibilistic decision criteria on policies without intermediate utilities and with intermediate utilities. Besides, we study the properties of these criteria. Indeed, we prove that these criteria satisfy Pareto efficiency as well as strict monotonicity.

2. Chapter 4 extends the lexicographic comparisons to possibilistic decision trees and possibilistic finite-horizon Markov decision processes. We propose an algorithmic solution, based on Dynamic Programming approach, for each model. Moreover, we provide an experimental study to validate and discuss the proposed algorithms.

3. The primary aim of chapter 5 is to show how to relate the theory of expected utility with that of possibilistic qualitative utilities by refinement relations in the case of sequential decision-making (obviously when considering finite-horizon possibilistic decision trees and possibilistic finite-horizon). We propose a special form of expected utility (based on big-stepped probability distributions) as a refinement of possibilistic qualitative utilities, using a transformation function of the scale. Then we establish formal results of equivalence between lexicographic refinements of qualitative utilities and these expected utility criteria proposed. Thus we prove that it is possible to construct stochastic models inducing an order on policies that refines the order induced on the same policies in the given possibilistic model. We define a dynamic programming algorithm, for calculating optimal policies with respect to the so-obtained criteria with an experimental study in the end.

4. Finally, Chapter 6 is devoted to stationary Markov decision processes in which the set of states, the available actions and the transition functions are assumed not do depend on the stage (time step) of the problem. First, we propose a lexicographic variant of the value iteration algorithm for the finite-horizon case, with an approximation algorithm in order to decrease the complexity of the fist. We present then an experimental comparative analysis of these algorithms. In addition, we provide a value iteration algorithm to compute an approximate lexicographic optimal policy when the horizon is infinite.

The main results of this thesis are published in [Ben Amor et al., 2015, Ben Amor et al., 2016b, Ben Amor et al., 2016a, Ben Amor et al., 2017].

CHAPTRE 1

One-stage Possibilistic Decision-Making

Contents

1.1 Introduction

Decision theory is a multidisciplinary domain that concerns economy, psychology, social sciences, operational research and artificial intelligence. Many important problems involve decision-making under uncertainty, that is choosing an act (also called decision or action) or, in sequential decision-making, a policy among many different available alternatives, considering decision maker's limited knowledge about states of nature (states of the world) and his preferences.

In real world several types of uncertainty should be considered. Most available decision models refer to probability theory for the representation of uncertainty [Neumann and Morgenstern, 1944, Savage, 1954]. Despite its success, probability theory is appropriate when all numerical information is available or can be easily elicited. When information about uncertainty cannot be quantified in a probabilistic way, several non-classical theories of uncertainty can be considered in order to deal with imperfect, ordinal information namely, fuzzy sets theory [Zadeh, 1965], evidence theory [Shafer, 1976] and possibility theory [Zadeh, 1978, Dubois and Prade, 1988] etc. In

this work, we will focus on qualitative decision-making, especially on possibility theory suitable for handling uncertain and imprecise knowledge.

This chapter provides the theoretical background of possibilistic decision theory: Section 1.2 presents classical probabilistic decision theory. Section 1.3 reviews the possibilistic decision theory, in particular, it introduces the basics of possibility theory and details the most commonly used possibilistic decision criteria: optimistic and pessimistic qualitative utilities. Section 1.4 exposes the drowning effect problem in these latter criteria. Finally, Section 1.5 defines refinements of qualitative utilities proposed for the one-step decision problems.

1.2 Decision-making under probabilistic uncertainty

Decision-making under uncertainty is primarily the identification and the choice of some alternatives, that most commonly are expressed implicitly, based on preferences of the decision maker. Thus, solving a decision problem amounts to providing an optimal act with respect to the available knowledge about the environment and the decision maker preferences relative to possible consequences of different alternatives.

In this Section, we present the probabilistic decision theory, considered as the standard quantitative decision model, and also the Expected Utility (EU) criterion first introduced by Bernoulli [Bernoulli, 1738], and then axiomatized by Von Neumann and Morgenstern [Neumann and Morgenstern, 1944] and Savage [Savage, 1954]. We first detail subjective expected utility and Savage's axiomatic system, then we present the expected utility criterion in the Von Neumann and Morgenstern's framework and its axiomatization.

1.2.1 Subjective Expected Utility: Savage's approach

Expected utility theory based on subjective probability has been well developed and axiomatized by Savage [Savage, 1954]. In Savage's framework, subjective probability is used to model uncertainty. Formally, a decision problem under uncertainty using the Savage approach is defined by a 4-tuple $(S, X, A, \succeq)$, where:

- S is the set of states of nature,
- X is the set of consequences, formally, the preference between two consequences x and $y \in X$ is denoted by $x \succeq y$. It means that "x is at least as good as y" for the decision maker.
- $A = X^S$ is the set of possible acts, an act is thus a function $f\colon S \mapsto X$,
- $\succeq$ is a preference relation on A satisfying two main crucial properties:

- Completeness: $\forall f, g \in A$ either $f \succeq g$ or $g \succeq f$.
- Transitivity: If f is at least as good as g and g is at least as good as q, then f is at least as good as q.

$$\text{If } (f \succeq g \text{ and } g \succeq q) \text{ then } (f \succeq q).$$

$f \succeq g$ denotes the preference between two acts f and g. It means that "f is at least as good as g" for the decision maker. $\succ$ denotes the asymmetric part of $\succeq$ and $\sim$ its symmetrical part:

- The strict preference relation $\succ$ is defined by $f \succ g$ if and only if $f \succeq g$ and $g \not\succeq f$ and it means that" f is strictly preferred to g ".
- The indifference relation $\sim$ is defined by $f \sim g$ if and only if $f \succeq g$ and $g \succeq f$ and it means that " f and g are indifferently preferred ".

Considering probability theory, the information pertaining to the state of nature and the preferences on X are encoded as follows: uncertainty is represented by a probability distribution p over S and the preferences on X are encoded by a utility function $\mu : X \mapsto U$.

Acts are then ranked (for any two acts, either one is better than the other, or the two are equivalent) by *Subjective Expected Utility*, denoted by SEU [Savage, 1954]:

Definition 1.1. *Given a probability distribution p over S and a utility function μ on X, the SEU of an act h is defined by:*

$$SEU(h) = \sum_{s \in S} p(s) \,.\, \mu(h(s)). \tag{1.1}$$

Example 1.1. *Suppose that you are thinking about taking out fire insurance on your home. Perhaps it costs \$100 to take out insurance on a house worth \$100,000, and you ask: Is it worth it? Let us formalize this decision problem: Let $S = \{Severe\ fire, Minor\ fire, No\ fire\}$ be the set of states of nature s.t.:*

- *$Severe\ fire$ is the state in which your house catches on a severe fire,*
- *$Minor\ fire$ is the state in which your house catches on a negligible fire,*
- *$No\ fire$ is the state in which your house doesn't catch on fire.*

Using fire occurrence data, we have $p(Severe\ fire) = 0.2$, $p(Minor\ fire) = 0.3$ and $p(No\ Fire) = 0.5$. Consider the two acts $Insurance$ (take an insurance) and $No\ insurance$ (do not take an insurance) and the consequences given in Table 1.1.

Acts/States	$Severe\ fire$	$Minor\ fire$	$Not\ fire$
$Insurance$	No house+($100)	House+($100-$50)	House+(-$100)
$Noinsurance$	No house+($0)	House+(-$50)	House+($0)

Table 1.1: Consequences of acts $Insurance$ and $No\ insurance$

The utility value associated to each consequence is defined as follows:

- $\mu(No\ house + \$100)$ = *3,* $\mu(House + (\$100 - \$50))$=*6,* $\mu(House + (-\$100))$=*7,*
- $\mu(No\ house + \$0)$=*0,* $\mu(House + (-\$50))$=*5,* $\mu(House + \$0)$=*10.*

Using Equation 1.1, we have:

- $SEU(Insurance) = (0.2 \times 3) + (0.3 \times 6) + (0.5 \times 7) = 5.9$ *and*
- $SEU(No\ insurance) = (0.2 \times 0) + (0.3 \times 5) + (0.5 \times 10) = 6.5.$

So, the act $No\ insurance$ *is preferred to* $Insurance$*.*

Savage has provided an axiomatic system that gives necessary conditions that should be satisfied by a preference relation $\succeq$ between acts to be represented by an EU [Savage, 1954]. First, we define fAh as the act which gives the same consequence f on $A \subseteq S$ and as h on $S \setminus A$. A constant act is defined by:

Definition 1.2. *(Constant act) A constant act* $f_x \in A$ *provides the same consequence* $x \in X$*, whatever the state of nature i.e.:*

$$\forall s \in S, f_x(s) = x.$$

Savage's axiomatic system is based on the five following axioms [Savage, 1954]:

Axiom. *(*$SAV1$*: Complete Pre-order) The preference relation* $\succeq$ *is complete and transitive.*

Axiom. *(*$SAV2$*: Sure Thing Principle (STP)) For all acts* $f, g, h, h' \in A$ *and for every event* $E \subseteq S$*:*

$$fEh \succeq gEh \text{ iff } fEh' \succeq gEh'.$$

Axiom. *(*$SAV3$*: Conditioning over constant acts) Thus, for any not null event* $E \subseteq S$*, and any constant acts* f_x*,* $g_y \in A$ *it holds that:*

$$\forall E \subseteq S, f_x \succeq g_y \text{ iff } \forall h \in A, f_xEh \succeq g_yEh.$$

Axiom. *($SAV4$: Projection from acts over events) For any consequences $x, y, x', y' \in X$, for any constant acts $f_x, g_y, f'_{x'}, g'_{y'} \in A$. If $x \succ y$ and $x' \succ y'$, then $\forall E, D \subseteq S$ we have:*

$$f_x E g_y \succeq f_x D g_y \text{ iff } f'_{x'} E g'_{y'} \succeq f'_{x'} D g'_{y'}.$$

Axiom. *($SAV5$: Non triviality) $\exists\, f, g \in A$, such that $f \succ g$.*

The principle axiom of Savage is the Sure Thing Principle (SAV2), it is interpreted by the fact that if an act is preferred to another when an event E is occurred then it will still preferred whatever the act in the case of complementary event.

If a preference relation $\succeq$ satisfies axioms $SAV1$ to $SAV5$, as well as two technical axioms of continuity and monotonicity, then this preference relation can be represented by an expected utility from the set of act to the reals:

Theorem 1.1. *If the preference relation $\succeq$ satisfies Savage axioms then it exists a utility function μ: $X \mapsto \mathcal{R}$ and a probability distribution p deduced from the preference relation between acts such that :*

$$\forall f, g \in A,\ f \succeq g \ \Leftrightarrow\ SEU(L) \geq SEU(L'). \tag{1.2}$$

1.2.2 Expected Utility: Von Neumann and Morgenstern's approach

In the present Section, we present the expected utility model in the Von Neumann and Morgenstern's framework [Neumann and Morgenstern, 1944], the most known framework to deal with decision-making problems under risk. In this framework, an act is represented by a probability distribution over the set of possible outcomes. It is called a *simple probabilistic lottery* and it is denoted by $L = \langle \lambda_1/x_1, \ldots, \lambda_n/x_n \rangle$, where $\lambda_i = p(x_i)$ is the probability that the decision leads to outcome x_i. A utility μ function maps each outcome x_i to a utility value in a totally ordered numerical set U. This function models the attractiveness of each outcome for the decision maker. Thus, a simple probabilistic lottery can be seen also as a probability distribution over the set of utilities denoted by $L = \langle \lambda_1/\mu_1, \ldots, \lambda_n/\mu_n \rangle$.

A probabilistic compound lottery denoted by $\langle\ \lambda_1/L_1, \ldots, \lambda_m/L_m\ \rangle$ is a probability distribution over a set of lotteries where λ_i is the probability to obtain lottery L_i.

Formally, a decision-making problem under risk can be represented using:

- X the set of consequences,
- $\mathcal{L}$ the set of probabilistic lotteries, where each lottery L_i is a probability distribution p over the set of consequences X,

- $\mu : X \mapsto U$ the utility function, a mapping from the set of consequences X to a numerical scale.

Note that, it is possible to transform a Savage act into a lottery, since we can calculate the probability p of getting each utility level from the probabilities of states:

$$\forall \mu_i \in U, \;\; p(\mu_i) = \sum_{s \in S, \mu(f(s))=\mu_i} p(s).$$

We get then a probability distribution over the set of utilities, i.e. simple probabilistic lottery.

Solving a decision problem under risk amounts to evaluating alternatives and choosing an optimal one among them. The computation of the expected utility of a lottery L is performed as follows:

Definition 1.3. *Given a probabilistic lottery $L = \langle \lambda_1/x_1, \ldots, \lambda_n/x_n \rangle$ and a utility function u, the expected utility of L (denoted by $EU(L)$) is computed by:*

$$EU(L) = \sum_{x_i \in X} \lambda_i \, . \, \mu(x_i). \tag{1.3}$$

Example 1.2. *Consider the same decision problem as in Example 1.1. The two acts $Insurance$ and $No\ insurance$ can be represented, respectively, using the two following probabilistic lotteries $L = \langle 0.2/3, 0.3/6, 0.5/7 \rangle$ and $L' = \langle 0.2/0, 0.3/5, 0.5/10 \rangle$.*

Using Equation 1.3, we have the same values as in Example 1.1 i.e. $EU(Insurance) = 5.9$ and $EU(No\ insurance) = 6.5$.

Von Neumann and Morgenstern have provided an axiomatic system to characterize a preference relation $\succeq$ between probabilistic lotteries [Neumann and Morgenstern, 1944]:

Axiom. *($VNM1$: Weak order) The preference relation $\succeq$ is complete and transitive.*

Axiom. *($VNM2$: Continuity) Let L, L' and L'' be three probabilistic lotteries, if L is at least good as L' and L' is at least good as L'' then there is a probability p for which the decision maker will be indifferent between lottery L' and the compound lottery in which L comes with probability p, and L'' with probability $(1-p)$. Formally:*

$$L \succeq L' \succeq L'' \Rightarrow \exists \, p \; \in]0,1[\; s.t. \; \langle p/L, (1-p)/L'' \rangle \sim L'.$$

Axiom. *($VNM3$: Independence) Let L, L' and L'' be three probabilistic lotteries, we have:*

$$L \succeq L' \Leftrightarrow \exists \, p \in]0,1] \; s.t. \; \langle p/L, (1-p)/L'' \rangle \; \succeq \; \langle p/L', (1-p)/L'' \rangle.$$

The fundamental axiom of the (objective) expected utility model is the independence axiom. It can be interpreted as follows: If the decision maker prefers L to L' and he has to choose between $\langle p/L, (1-p)/L''\rangle$ and $\langle p/L', (1-p)/L''\rangle$ then he will prefer the compound lottery $\langle p/L, (1-p)/L''\rangle$ to $\langle p/L', (1-p)/L''\rangle$ whatever the probability of the event that happens.

Thus, if the preference relation $\succeq$ satisfying completeness, transitivity, continuity and independence axioms, then it can be represented by an expected utility. This is the meaning of the following representation theorem:

Theorem 1.2. *If the preference relation $\succeq$ satisfies VNM axioms, then it is exists a utility function μ: $X \mapsto \mathcal{R}$ over the set of lotteries $\mathcal{L}$ such that:*

$$\forall L, L',\ L \succeq L' \ \Leftrightarrow\ EU(L) \geq EU(L'). \tag{1.4}$$

1.2.3 Beyond Expected Utility decision theories

Despite its success, the Expected Utility theory has some limitations such as:

- The behavior of total ignorance is represented by equiprobability, so it formalizes randomness instead of ignorance.
- EU is not able to capture certain behaviors that appear rational such as those shown in the paradoxes of Allais [Allais, 1953] and Ellsberg [Ellsberg, 1961]: In 1953, Allais paradox has shown that the independence axiom of the VNM's system is violated [Allais, 1953] and in 1961, Ellsberg has shown the behavior of people in the face of ignorance is in contradiction with the Sure Thing Principle of SEU.
- EU-theory supposes that all the numerical information is available or that it can be elicited from the decision maker. In some situations, decision makers are unable to express their uncertainty and preferences numerically for all comparisons.

In order to overcome EU-based model limitations, some extensions of EU, that use quantitative representation of uncertainty, have been developed. Most of them are based on the most well-known criterion Choquet integral [Choquet, 1954] which is based on generalized measure of uncertainty. Quiggin has developed the Rank-Dependent Utility criterion (RDU) (which is a particular case of Choquet integral) that have received an axiomatic justification [Quiggin, 1982].

When uncertainty and/or preferences are ordinal in nature, one can use alternatives to probability theory. So, considering qualitative preferences but remaining within a probabilistic quantification of uncertainty has led to quantile-based approaches [Montes et al., 2014, Szörényi et al., 2015]. In several cases, especially when the representation of both uncertainty and preferences by

additive quantitative values is inappropriate, purely ordinal approaches have been considered to handle qualitative uncertainty and preferences. Qualitative decision models have been introduced with emergence of the rule of Sugeno integrals [Sugeno, 1974], that is considered as an ordinal counterpart of Choquet integrals.

1.3 Possibilistic decision theory

Possibility theory [Zadeh, 1978, Dubois and Prade, 1995] is the fundamental purely ordinal uncertainty theory. For many years, it has received much interest in the Artificial Intelligence community [Pearl, 1993, Dubois and Prade, 1995, Dubois et al., 1998a, Dubois et al., 2001b, Godo and Zapico, 2001, Giang and Shenoy, 2005, Weng, 2005].

In this Section, we are interested in qualitative decision theory by using possibility theory for the representation of uncertainty. The development of possibilistic decision theory has led to the proposition of several possibilistic decision criteria, in particular qualitative utilities : optimistic and pessimistic utilities. These criteria are used to identify a preference relation on simple possibilistic lotteries, associated to acts, defined in what follows.

1.3.1 Possibility theory

Possibility theory, issued from Fuzzy Sets theory, offers a natural and flexible model to represent and handle uncertain information, especially qualitative uncertainty and total ignorance. It was introduced by Zadeh [Zadeh, 1978] and further developed by Dubois and Prade [Dubois and Prade, 1988].

The basic component of possibility theory is the notion of possibility distribution. It is a representation of the knowledge of an agent regarding the state of the world. A possibility distribution is denoted by π and it is a mapping from the universe of discourse S to a finite ordinal scale $V = \{\alpha_0 = 0_V < \alpha_1 < \ldots < \alpha_l = 1_V\}$, we denote the function by: $\pi : S \rightarrow V$.

For each state $s \in S$, $\pi(s) = 1$ means that realization of the state s is totally possible and $\pi(s) = 0$ means that s is an impossible state. $\pi(s) > \pi(s')$ expresses that s is preferred to s' (or is more plausible).
It is generally assumed that there exist at least one state s which is totally possible i.e. $\pi(s) = 1$: π is said then to be *normalized*.

The possibilistic scale V can be interpreted in two manners:

- **Quantitative** manner, or numerical, using possibility degrees i.e. the values of the possibility distribution make sense in the possibilistic scale.

- **Qualitative** manner, or ordinal, using total pre-order on the universe of discourse. This order can be represented by numerical values which have no sense but which express only the order.

In the possibilistic framework, extreme forms of partial knowledge can be captured, namely:

- *C*omplete knowledge i.e. $\exists\ s$ s.t. $\pi(s) = 1$ and $\forall\ s' \neq s, \pi(s') = 0$.
- *T*otal ignorance i.e. $\forall s \in S, \pi(s) = 1$ (all states in S are possible).

In possiblistic theory, for any used scale V, there are two essential measures:

- **Possibility measure:** $\Pi(A) = \max_{\omega \in A} \pi(\omega)$.
  $\Pi(A)$ is the possibility degree evaluating at which level the event A is consistent with the knowledge represented by π.
- **Necessity measure:** $N(A) = 1 - \Pi(\bar{A}) = 1 - \sup_{\omega \notin A} \pi(\omega)$.
  N(A) expresses the necessity degree evaluating at which level the event A is certainly implied by the knowledge.

Example 1.3. *Let us consider the possibility distribution π which shows the opinion of a doctor concerning the diagnosis of a patient. The universe of discourse related to this problem is a set of three diseases and a healthy case: $S = \{d1, d2, d3, h\}$.*

$$\pi(d1) = 0.5, \quad \pi(d2) = 1, \quad \pi(d3) = 0.7, \quad \pi(h) = 0.$$

Note that this distribution is normalized since $max(0.5, 1, 0.7, 0) = 1$.

If we consider the event A: "The patient suffers from $d1$ or $d3$", then we have:
$\Pi(A) = max(0.5, 0.7) = 0.7$ *and* $N(A) = 1 - max(1, 0) = 0.0$.

1.3.2 Possibilistic lotteries

Dubois et al. [Dubois and Prade, 1995, Dubois et al., 1998a] have proposed a possibilistic counterpart of VNM's notion of lottery to represent a one stage decision problem. In the possibilistic framework, an act can be represented by a possibility distribution on $V = \{u_1, \ldots, u_p\}$, also called a possibilistic lottery, and denoted by $\langle \lambda_1/u_1, \ldots, \lambda_p/u_p \rangle$ i.e. $\lambda_i = \pi(u_i)$ is the possibility that the decision leads to an outcome of utility u_i.

A *possibilistic compound lottery* $\langle \lambda_1/L_1, \ldots, \lambda_r/L_r \rangle$ is a normalized possibility distribution over the set of lotteries. The possibility $\pi_{i,j}$ of getting an utility degree $u_j \in V$ from one of its sub-lotteries L_i depends on the possibility λ_i of getting L_i and on the conditional possibility $\lambda_j^i = \Pi(u_j \mid L_i)$ of getting u_j from L_i i.e. $\Pi_{i,j} = min(\lambda_j, \lambda_j^i)$.

Thus, [Dubois and Prade, 1995,Dubois et al., 1998a] have proposed to reduce a compound lottery (over a set of simple lotteries) $\langle \lambda_1/L_1, ..., \lambda_r/L_r \rangle$ into an equivalent simple lottery. Formally we have:

$$Reduction(\langle \lambda_1/L_1, \ldots, \lambda_r/L_r \rangle) = \langle \max_{j=1..r}(min(\lambda_1^j, \lambda_j))/u_1, \ldots, \max_{j=1..r}(min(\lambda_p^j, \lambda_j))/u_p \rangle. \tag{1.5}$$

Example 1.4. *Let* $L_1 = \langle 1/0.5, 0.7/0.3 \rangle$ *and* $L_2 = \langle 1/0.5, 0.6/0.3 \rangle$ *be two simple possibilistic lotteries, and let* $L' = \langle 1/L_1, 0.8/L_2 \rangle$ *be th compound lottery represented in Figure 1.1 (a). The reduction of* L' *into a simple lottery* L''*, presented by (b) in Figure 1.1, can be calculated using Equation 1.5 as follows:*

- $L''(0.5) = \max(\min(1, 1), \min(0.8, 0.6)) = 1$ *and*
- $L''(0.3) = \max(\min(1, 0.7), \min(0.8, 1)) = 0.8$,

So, L'' = *Reduction* $(L') = \langle 1/0.5, 0.8/0.3 \rangle$.

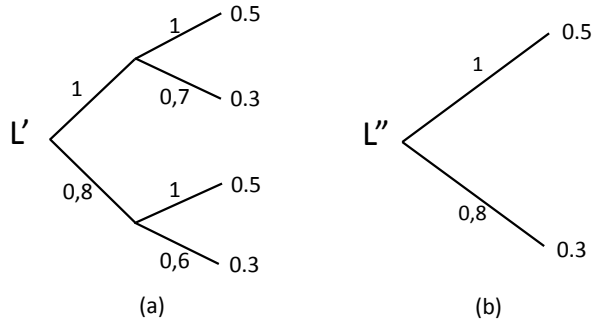


Figure 1.1: A possibilistic compound lottery L' (a) and its reduction L'' (b)

Note that, the reduction of a simple lottery is the simple lottery itself. Furthermore, the reduction of a compound lottery is a polynomial operation, because min and max are also polynomial operations.

The development of possibility theory has led to the emergence of several possibilistic criteria depending on the nature of the utility scale. These criteria allow to compare simple lotteries associated to acts as detailed in what follows.

1.3.3 Possibilistic decision criteria

Possibilistic decision criteria can be gathered into three classes:

- **Utility and possibility scales are commensurate and purely ordinal:** contains possibilistic qualitative criteria such as optimistic utility (denoted by u_{opt}) and pessimistic utility (denoted by u_{pes}) [Dubois and Prade, 1995, Dubois et al., 1999] that are qualitative counterparts of the EU criterion, besides possibilistic binary utilities (denoted by PU) [Giang and Shenoy, 2001].
- **Utility and possibility scales are commensurate and utilities are quantitative:** in this class we have possibility-based Choquet integrals (denoted by Ch_{Π}) and Necessity-based Choquet integrals (denoted by Ch_N) [Rebille, 2006, Ben Amor et al., 2010, Dubois and Rico, 2016b].
- **Utility and possibility scales are not commensurate:** we have possibility-based likely dominance (denoted by $L\Pi$) and Necessity-based likely dominance (denoted by LN) [Fargier and Perny, 1999, Dubois et al., 2003].

In this Thesis, we are interested in the first class, especially in the well-known qualitative utilities: optimistic and pessimistic utilities that are qualitative counterparts of the EU criterion.

1.3.4 Possibilistic qualitative utilities

Under the assumption that the utility scale and the possibility scale are commensurate and purely ordinal, Dubois and Prade [Dubois and Prade, 1995,Dubois et al., 1999] have proposed pessimistic and optimistic criteria. These two qualitative criteria are actually particular cases of a more general criterion based on the Sugeno integral [Sugeno, 1974].

1.3.4.1 Pessimistic utility

The pessimistic criterion was originally defined by Whalen [Whalen, 1984] and it generalizes the Wald criterion [Wald, 1971]. It supposes that the decision maker is happy when bad consequences are hardly plausible i.e. considers the bad and plausible consequences. It estimates to what extent it is certain (i.e. necessary according to measure N) that L reaches a good utility.

Definition 1.4. *Let $L = \langle \lambda_1/u_1, \ldots, \lambda_n/u_n \rangle$ be a possibilistic lottery, the pessimistic utility of L, denoted by u_{pes} is computed as follows:*

$$u_{pes}(L) = \min_{i=1..n} \max(u_i, n(\lambda_i)). \tag{1.6}$$

where n is an order reversing function (e.g. $n(\lambda_i) = 1 - \lambda_i$).

Example 1.5. *Let $L = \langle 1/0.2, 0.7/0.5, 0.4/0.6 \rangle$ and $L' = \langle 1/0.8, 0.3/0.7, 0.5/0.9 \rangle$ be two possibilistic lotteries. Using Equation 1.6 we have:*

- $u_{pes}(L) = min(max(0.2, 0), max(0.5, 0.3), max(0.6, 0.6)) = 0.2$
- $u_{pes}(L') = min(max(0.8, 0), max(0.7, 0.7), max(0.9, 0.5)) = 0.7$

which means that $L' \succ_{u_{pes}} L$.

Dubois and Prade have proposed an adaptation of qualitative utilities to evaluate acts in the style of Savage [Dubois et al., 1998b]. Using this framework, the definition of the pessimistic decision rule is as follows:

Definition 1.5. *Given a possibilistic distribution* π *over* S *and a utility function* u *on the set of consequences* X*, the pessimistic utility of an act* f *is defined by:*

$$u_{pes}(f) = \min_{s_i \in S} \max(u(f(s_i)), 1 - \pi(s_i)). \tag{1.7}$$

Therefore, we can compare two acts f and $g$on the basis of their pessimistic utilities:

$$f \succeq_{u_{pes}} g \Leftrightarrow u_{pes}(f) \geq u_{pes}(g).$$

1.3.4.2 Optimistic utility

The optimistic criterion was originally proposed by Yager [Yager, 1979, Yager, 1997] and it captures the optimistic behavior of the decision maker that makes at least one of the good consequences highly plausible. So, this criterion estimates to what extent it is possible that a possibilistic lottery reaches a good utility.

Definition 1.6. *Let* $L = \langle \lambda_1/x_1, \ldots, \lambda_n/x_n \rangle$ *be a possibilistic lottery, the optimistic utility of* L*, denoted by* u_{opt} *is computed as follows:*

$$u_{opt}(L) = \max_{i=1..n} \min(u_i, \lambda_i). \tag{1.8}$$

Example 1.6. *Let* $L = \langle 1/0.2, 0.7/0.5, 0.4/0.6 \rangle$ *and* $L' = \langle 1/0.8, 0.3/0.7, 0.5/0.9 \rangle$ *be two possibilistic lotteries. Using Equation 1.8 we have:*

- $u_{opt}(L)$ = *max(min(0.2,1),min(0.5,0.7),min(0.4,0.6)) = 0.5.*
- $u_{opt}(L')$ = *max(min(1,0.8),min(0.3,0.7),min(0.5,0.9)) = 0.8.*

which means that $L' \succ_{u_{opt}} L$.

Similar to its pessimistic counterpart, this criterion was also defined in Savage framework to choose among acts rather than lotteries. Its definition is as follows:

Definition 1.7. *Given a possibilistic distribution π over a set S and a utility function u on a set of consequences X, the optimistic utility of an act f is defined by:*

$$u_{opt}(f) = \max_{s_i \in S} \min(u(f(s_i)), \pi(s_i)). \tag{1.9}$$

The pessimistic and optimistic utilities represent particular cases of Sugeno integrals, a more general criterion, in the context of possibility theory [Sugeno, 1974, Grabisch et al., 2000, Dubois et al., 2001a, Marichal, 2001]:

$$S_{\gamma,u}(L) = \max_{\lambda \in [0,1]} \min(\lambda, \gamma(F_\lambda)). \tag{1.10}$$

where $F_\lambda = \{s_i \in S, u(f(s_i)) \geq \lambda\}$, is a set of preferred states for act f. γ captures knowledge (necessity or possibility measures) and reflects the decision maker attitude toward uncertainty.

u_{opt} is recovered when γ is the *possibility measure* Π and u_{pes} is recovered when γ corresponds to *necessity measure* N.

1.4 Limitations of possibilistic qualitative utilities

Qualitative decision criteria in general, and qualitative utilities in particular, suffer from a lack of discrimination called the "drowning effect" due to the use of idempotent operations-max and min. This shortcoming is explicit when considering the Savage's formalism. Indeed, two acts f and g can be considered as indifferent even if they give an identical and extreme (either good or bad) consequence in some plausible state. As a consequence the principle of Pareto efficiency also called strict Pareto dominance: $\forall\ s,\ u(f(s)) \geq u(g(s))$ and $\exists\ s^*,\ \pi(s^*) > 0\ and\ u(f(s^*)) > u(g(s^*))$ does not imply that f is strictly preferred to g (i.e. $g \succ f$), as shown in the following example borrowed from [Fargier and Sabbadin, 2005]:

Example 1.7. *Let $S = \{s_1, s_2\}$ and $V = \{0, 0.1, 0.2, 0.3, 0.4, 1\}$. Let f and g be two acts whose utility consequences in the states s_1 and s_2 are listed in the following table, as well as the degrees of possibility of s_1 and s_2:*

Table 1.2: Drowning effect

	s_1	s_2
$u(f(s))$	0.3	0.4
$u(g(s))$	0.3	0.1
π	1	0.2

Thus $u_{opt}(f) = u_{opt}(g) = 0.3$ (and $u_{pes}(f) = u_{pes}(g) = 0.3$) although f strictly dominates g in the state s_2 and that the two acts are equivalent in s_1. This is due to the fact that the good

performance of the two acts on the most plausible state ($\min(u(f(s_1)), \pi(s_1)) = \min(3, 5) = \min(u(g(s_1)), \pi(s_1)) = 3$*) blurs the comparison.*

Moreover, it has been shown that optimistic and pessimistic qualitative utilities do not fully satisfy the fundamental property of decision criteria: the Sure Thing Principle (STP) (axiom. SAV2) that insures that identical consequences do not influence the relative preference between two acts. In fact, it has been shown that qualitative utilities respect the STP only if there is no uncertainty at all [Fargier and Sabbadin, 2005]. Note that when $\succeq$ is complete and transitive, the principle of Pareto efficiency is a direct consequence of the Sure thing principle.

1.5 Refinements of possibilistic qualitative utilities

Contrary to the EU approach, possibilistic qualitative preference relations u_{opt} and u_{pes} suffer from a lack of decisiveness, since they do not satisfy the STP. In this section, we present some approaches that try to remedy this problem.

1.5.1 Refining qualitative decision criteria

As a solution to qualitative criteria's drawbacks, it seems to be interesting to stay in the pure qualitative framework and escape the drowning effect by satisfying the STP. For this purpose, many papers have proposed some tools to cope with the drowning effect of possibilistic decision criteria. The idea is to propose discriminating criteria i.e. refinements of the possibilistic criteria. Formally:

Definition 1.8. *(Refinement) Let $\succeq$ and $\succeq'$ be any two complete preference relations on the set of acts A. We say that $\succeq'$ refines $\succeq$ if and only if $\forall\ f, g \in A, f \succ g \Rightarrow f \succ' g$, where $\succ$ and $\succ'$ are the strict parts of $\succeq$ and $\succeq'$ respectively.*

Definition 1.9. *A relation $\succeq'$ is said to be more specific than a relation $\succeq$ if and only if $\succeq'$ refines $\succeq$ and $\succeq \neq \succeq'$.*

Thus, a refinement $\succeq'$ agrees with $\succeq$ when $\succeq$ provides a strict preference. So, the two preference relations are perfectly compatible. Obviously, when $\succeq'$ refines $\succeq$, either they provide the same decisions, or the former is more specific.

The idea of refining qualitative possibilistic criteria has first been proposed by [Dubois et al., 2000]: when two acts result indifferent w.r.t. a possibilistic criterion, the ties are broken by sequentially considering the ranking w.r.t. another qualitative criteria. Hence, the refined ordering is obtained by sequentially applying additional qualitative criteria to the original one. For instance, when the pessimistic utility criterion is indifferent, it is possible to decide based on the optimistic

utility ordering or a T-norm aggregation. This refinement allows to evaluate acts based on a lexicographic ordering of a set of different qualitative criteria, instead of an order induced by one criterion. In fact, [Dubois et al., 2000]'s approach is purely qualitative, since it uses min and max but it does not obey the STP.

Then, in [Giang and Shenoy, 2001], the authors have improved the discrimination power of possibilistic utilities by proposing Binary possibilistic utility in order to compare special kind of qualitative lotteries: totally ordered set of possibility measures on a two element set $\{0, 1\}$ containing the utilities (i.e. the scale of utilities) of the best and the worst consequences. Such possibility distribution represents a qualitative lottery. Note that the pessimistic and optimistic utilities are special cases of this bipolar criterion. The decision rule of [Giang and Shenoy, 2001] can capture the lexicographic use of the optimistic and pessimistic utilities when the pessimistic one cannot discriminate and vice-versa [Godo and Zapico, 2005]. Besides, the proposed criterion satisfies the STP but it has a major shortcoming: Consider two acts with contrasted consequences, i.e. respectively a bad or neutral one, and a good or neutral one, and that the two acts have maximal possibility. It appears that the binary possibilistic utility may be indifferent because of the drowning effect of min and max: neutral consequences (possibility and utility equal to 1) hide all other consequences and make all such lotteries equivalent. A refinement of binary possibilistic utility has been proposed in [Weng, 2005]. In this work indeed, two similar possibilities of the same lottery are merged, using new operators, in a lexicographical ordered sequence of values instead of one possibility degree that we get after the reduction. The resulting criterion thus suffers from a drowning effect since it deletes all doubles of branches in each lottery.

In [Lehmann, 2002], Lehmann adds some qualitative notions to the classical expected utility framework. He has axiomatized a refined criterion, based on the qualitative maximin criterion of [Wald, 1971], in the VNM style: it breaks ties between equivalent worst states by considering their respective likelihoods (the probabilities). This criterion takes the form of an expected utility criterion with standard probabilities and qualitative infinitesimal utilities. In this model, two situations can be considered: nonstandard (lexicographic) probabilities with standard utilities or standard probabilities with nonstandard (lexicographic) utilities. Note that in this model the lexicographic characteristic is used only on one of the two dimensions (either the probability level, or the utility level).

In addition, [Fargier and Sabbadin, 2000, Fargier and Sabbadin, 2002] proposed refinements of monotonic utilities (they are based on Sugeno integral): when comparing two acts, every state in which both acts give identical results being neglected (by forcing the two acts to take a fixed value α). This refinement obeys the STP but it claims to forget the transitivity of the indifference relation.

In order to overcome the lack of decisiveness of the possibilistic optimistic and pessimistic utilities, Fargier and Sabbadin show that possibilistic utilities can be refined by an EU criterion [Fargier and Sabbadin, 2003, Fargier and Sabbadin, 2005]. Choosing an EU-model is advantageous, since EU satisfies the STP and the principle of Pareto dominance. The authors show

that these EU-based Refinements of qualitative utilities can be understood as a generalization of the leximin and leximax criteria. We have chosen these refinements since they ensure optimal discrimination in accordance with Pareto efficiency, especially they respect the STP. In the sequel of this Chapter, we detail these latter refinements that constitute the basis of our contribution.

1.5.2 Expected utility-based refinements of qualitative utilities

The EU-based refinements of qualitative utilities criteria proposed in [Fargier and Sabbadin, 2003, Fargier and Sabbadin, 2005] satisfy the Sure Thing Principle - and thus escape the drowning effect. They are based on the use of a probability distribution p and a utility function μ such that the obtained preference relation $\succeq_{EU,p,\mu}$ refines the preference relation that corresponds to the decision rule u_{opt} when considering a possibility distribution π and a utility function u, denoted in this Section $\succeq_{OPT,\pi,u}$. The idea is to construct the expected utility criterion by a transformation function $\chi : V \rightarrow [0,1]$ which transforms each possibility distribution π into a probability distribution p:

Definition 1.10. *(Probabilistic transformation of an ordinal scale)*
A transformation of the scale V is a strictly increasing function $\chi : V \rightarrow [0,1]$ such that $\chi(0_V) = 0$; χ is a probabilistic transformation with respect to the possibility distribution π if $\sum_{s\in S} \chi(\pi(s)) = 1$.

Note that $\chi(0_V) = 0$ means that the impossibility of an event is expressed by a null probability. However the most plausible events (possibility degrees of 1_V, do not receive probability degree 1, since they are mutually exclusive. Note that χ is a unique function for transforming the scale V, thus both p and μ will be built upon this transformation. This is due to the fact that preference and uncertainty levels are commensurate and belong to the same scale. In the refined decision model, no undesirable arbitrary information is introduced by χ, so, p and μ are as close as possible to the original information about π and u: transformations of V is said to be "unbiased". Formally:

Definition 1.11. *(Unbiased transformation of a scale)*
χ is unbiased iff $\forall\, \alpha, \alpha' \in V,\ \ \alpha \geq \alpha' \ \Leftrightarrow\ \chi(\alpha) \geq \chi(\alpha')$.

This property ensures that π and $p = \chi \circ \pi$, the transformation of the distribution π using χ, (resp. u and $\mu = \chi \circ u$) are ordinarily equivalent.

From χ, it is possible to define a decision model of the expected utility type, transformed from the optimistic one$\langle S, X, A, \succeq_{EU,\chi\circ\pi,\chi\circ u}\rangle$, by exhibiting a sufficient condition on the χ function (named H) so that the obtained preference relation refines the optimistic possibilistic preference relation:

$$H\ : \forall \alpha, \alpha'\ \in V \textit{ such that } \alpha > \alpha' : \chi(\alpha)^2 > |S|.\ \chi(1_V).\chi(\alpha').$$

Then formally, $\succeq_{EU,\chi\circ\pi,\chi\circ u}$ refines $\succeq_{OPT,\pi,u}$ whenever χ satisfies H.

If χ respects H, the probability distribution $p = \chi \circ \pi$ obtained by transforming π is a big-step probability [Snow, 1999, Benferhat et al., 1999], i.e verifying: $\forall s \in S, p(s) > \sum_{s', p(s)>p(s')} p(s')$.

Note that the condition H concerns only the scale V and the size of the state space and not the distributions π or utilities u, hence, p and μ will be built using the transformation function of the scale. If $\langle S, X, A, \succeq_{OPT,\pi,u}\rangle$ ranks an optimistic possibilistic model, all expected utility type preference relations defined from χ satisfying H are equivalent.

Example 1.8. *Let us return to Example 1.7 and write* $\chi^* : V = (\alpha_0 = 5 > ... > \alpha_k = 0)$ *the probabilistic transformation defined by:* $\chi^*(0) = 0$*;* $\chi^*(\alpha_i) = \frac{v}{N^{2^{i+1}}}$, $i = 0, k-1$*; where* $v = \frac{1}{\sum_{i=0,k-1} \frac{n^i}{N^{2^{i+1}}}}$, $N = |S|$ *and* $n_i = |\{s \in S, \pi(s) = i\}|$. *In Example 1.7,* $N = 2$ *and* $L = \{5, 4, 3, 2, 1, 0\}$.

$\chi^*(V)$ *is the series* $\frac{v}{N^2}, \frac{v}{N^4}, \frac{v}{N^8}, \frac{v}{N^{16}}, \frac{v}{N^{32}}, 0$*; where* $v = \frac{1}{\frac{1}{N^2}+\frac{1}{N^{16}}}$.

So:

- $EU(f) = \chi^*(5).\chi^*(3) + \chi^*(4).\chi^*(2) = \frac{v^2}{N^{10}} + \frac{v^2}{N^{20}}$,
- $EU(g) = \chi^*(5).\chi^*(3) + \chi^*(2).\chi^*(1) = \frac{v^2}{N^{10}} + \frac{v^2}{N^{48}}$.

Hence, $f \succ_{EU,\chi^*\circ\pi,\chi^*\circ u} g$

In the sequel of this Section, χ^* will denote the transformation function $\chi^*(\alpha_i) = \frac{v}{N^{2^{i+1}}}$, $i = 0, k-1$ obtained with $v = \frac{1}{\sum_{i=0,k-1} \frac{n^i}{N^{2^{i+1}}}}$. Fargier et Sabbadin have shown that χ^* is not a unique generator for the expected utilities: there may exist other unbiased and specific functions, for instance χ^{**}, that attach different numbers to states i.e. $(\chi^{**} \circ \pi \neq \chi^* \circ \pi)$ or to consequences $(\chi^{**} \circ u \neq \chi^* \circ u)$. The two obtained models are ordinally equivalent i.e. they make the same decisions and rank acts in the same way. The following theorem presents a summary for optimistic transformation:

Theorem 1.3. *(Optimistic transformation) For any qualitative possibilistic model, there exists a probabilistic transformation* χ^* *of* V *such that:*

- $\succeq_{EU,\chi^*\circ\pi,\chi^*\circ u}$ *is an unbiased, optimistic refinement of* $\succeq_{OPT,\pi,u}$.
- *If* $\succeq_{EU,p,\mu}$ *is an unbiased and optimistic refinement* $\succeq_{OPT,\pi,u}$ *then* $\succeq_{EU,p,\mu} = \succeq_{EU,\chi^*\circ\pi,\chi^*\circ u}$.

It has been shown that an EU-based refinement of the same type (but somewhat more complex to express) could be obtained in the case of the pessimistic utility criterion, since $\succeq_{OPT,\pi,u}$ and $\succeq_{PES,\pi,u}$ are dual relations. It is possible to use the transformation χ^* (that allows to get probability degrees) coupled with a transformation function ψ^* to get the utilities s.t. $\psi^*(\alpha) = \chi^*(1_V) - \chi^*(n(\alpha))$.

This lead to the following pessimistic counterpart of Theorem 1.3:

Theorem 1.4. *(Pessimistic transformation) For any qualitative possibilistic model, there exists at least one pair of transformations (χ^*, ψ^*) of V satisfying H such that:*

- $\succeq_{EU,\chi^*\circ\pi_i,\psi^*\circ u}$ *is an unbiased refinement of* $\succeq_{PES,\pi,u}$.
- *If* $\succeq_{EU,p,\mu}$ *is an unbiased and pessimistic refinement* $\succeq_{pes,\pi,u}$ *then* $\succeq_{EU,p,\mu}$ *is equivalent to* $\succeq_{EU,\chi^*\circ\pi_i,\psi^*\circ u}$.

1.5.3 Lexicographic refinements of qualitative utilities

The possibilistic qualitative utilities can be refined using the leximin and leximax orderings proposed by [Moulin, 1988] to compare vectors [Fargier and Sabbadin, 2005]. In fact, the preference relation leximin (resp. leximax) has been proposed as refinement relation of min (resp. max), as well as discrimin order [Dubois et al., 1996, Cayrol et al., 2014] defined below: Let $\vec{f}$ denotes a vector of N elements of V i.e. $\vec{f} \in V^N$. $\forall i \in 1..N$, f_i denotes the i^{th} element of $\vec{f}$.

Definition 1.12. *To compare two vectors $\vec{u}, \vec{v} \in V^N$ (two vectors of N elements of V) using discrimin order, we first delete all pairs (u_i, v_j) from $\vec{u}$ and $\vec{v}$ s.t. $u_i = v_i$. let D be the set of indices of elements not deleted. Then, $\vec{u} \succ_{discrimin} \vec{v}$ iff $\min_{i\in D} u_i > \min_{i\in D} v_i$.*

Hence, leximax denoted $\succeq_{lmin}$ (resp. leximax denoted $\succeq_{lmax}$) allows to escape the lack of decisive power of $\succeq_{min}$ and $\succeq_{discrimin}$ orderings (resp. $\succeq_{max}$ order):

Definition 1.13. *(Leximax, Leximin)*
Let us consider $\vec{u}, \vec{v} \in V^N$ (vectors of N elements of V), and let $\forall i \in 1..N$, u_i (resp. v_i) denotes the i^{th} element of $\vec{u}$ (resp. $\vec{v}$). The leximax and the leximin relations on $\vec{u}$ and $\vec{v}$ denoted $\geq_{lmax}$ and $\geq_{lmin}$, respectively are defined as follows:
$\vec{u} \succeq_{lmax} \vec{v} \Leftrightarrow (\forall j, u_{(j)} = v_{(j)})$ *or* $(\exists\, i\ s.t.\ \forall\, j < i,\ u_{(j)} = v_{(j)}$ *and* $u_{(j)} > v_{(j)})$,
$\vec{u} \succeq_{lmin} \vec{v} \Leftrightarrow (\forall j, u_{(j)} = v_{(j)})$ *or* $(\exists\, i\ s.t.\ \forall\, j > i,\ u_{(j)} = v_{(j)}$ *and* $u_{(j)} > v_{(j)})$.

where for any $\vec{w} \in V^N$, $w_{(k)}$ is the k^{th} biggest element of $\vec{w}$ $(i.e.\ w_{(1)} \geq ... \geq w_{(N)})$.

Indeed, the leximax (resp. leximin) comparison consists in ordering both vectors in increasing (resp. decreasing) order and then lexicographically comparing them element by element. Moreover, both relations escape the drowning effect and are very efficient: the only pair of ties is vectors that are identical up to a permutation of their elements.

Example 1.9. *Let $\vec{u} = (3, 2, 6)$ and $\vec{v} = (2, 2, 6)$.*
$\vec{u} \succ_{lmax} \vec{v}$ *since* $u_{(1)} = v_{(1)} = 6$ *and* $u_{(2)} = 3 > v_{(2)} = 2$.
$\vec{u} \succ_{lmin} \vec{v}$ *since* $u_{(3)} = v_{(3)} = 2$ *and* $u_{(2)} = 3 > v_{(2)} = 2$.

Then, when S is finite, the comparison of acts can indeed be seen as a comparison of vectors of pairs of elements of V using 2 dimensions $(V^M)^N$, instead of simple vectors in (V^N). Hence, an act can be represented by a matrix with N lines of pairs, s.t. N is the number of states, and $M = 2$ columns (one for possibility degree and one for the corresponding utility):

Definition 1.14. *The representative matrix of any act $f \in A$ is:*
$\vec{f} = ((\pi_1, u_1), ..., (\pi_i, u_i), ..., (\pi_N, u_N))$,
where π_i corresponds to $\pi(s_i)$ and u_i to $u(f(s_i))$.

Example 1.10. *Let us return to Example 1.7. The representative matrices of f and g are:* $F = ((1, 0.3), (0.2, 0.4))$ *and* $G = ((1, 0.3), (0.2, 0.1))$.

Therefore, [Fargier and Sabbadin, 2003, Fargier and Sabbadin, 2005] proposed an extension of the lexicographic relations to acts using a complete pre-order $\trianglerighteq$ on lines of V^2 instead of the classical relation $\succeq$ on V. So, to compare two matrices representing two acts it is sufficient to order the lines of each matrix according to $\trianglerighteq$ and to apply then any of the two previous leximax and leximin procedures:

First, let F denotes a $N \times M$ matrix of elements of V i.e. $F \in (V^M)^N$ (s.t. $M \geq 2$). $\forall i \in 1..M; \forall j \in 1..N$, f_{ij} denotes an element (or coefficient) of F in line i and column j.

Definition 1.15. *(Leximax($\trianglerighteq$); Leximin($\trianglerighteq$))*
Let $\trianglerighteq$ be a complete pre-order on V^M, $\triangleright$ is its strict part and $\cong$ is its symmetric part. Let F, G be two matrices of $(V^M)^N$, the leximax and leximin relations on F and G denoted $\succeq_{lmax(\trianglerighteq)}$ and $\succeq_{lmin(\trianglerighteq)}$, respectively, are defined as follows:

- $F \succeq_{lmax(\trianglerighteq)} G \Leftrightarrow (\forall j, f_{(\trianglerighteq,j)} \cong g_{(\trianglerighteq,j)})$ *or* $(\exists\, i \; s.t. \; \forall\, j < i, f_{(\trianglerighteq,j)} = g_{(\trianglerighteq,j)}$ *and* $f_{(\trianglerighteq,i)} \triangleright g_{(\trianglerighteq,i)})$
- $F \succeq_{lmin(\trianglerighteq)} G \Leftrightarrow (\forall j, f_{(\trianglerighteq,j)} \cong g_{(\trianglerighteq,j)})$ *or* $(\exists\, i \; s.t. \; \forall\, j > i, f_{(\trianglerighteq,j)} = g_{(\trianglerighteq,j)}$ *and* $f_{(\trianglerighteq,i)} \triangleright g_{(\trianglerighteq,i)})$

where $\forall i \in 1, .., N$, $f_{(\trianglerighteq,i)}$ (resp. $g_{(\trianglerighteq,i)}$) is the i^{th} biggest line of F (resp. G) according to $\trianglerighteq$.

When $\trianglerighteq = \succeq_{lmin}$, the comparison consists in first ordering the elements of each line in increasing order w.r.t $\succeq_{lmin}$, then in ordering the lines in decreasing order (w.r.t. $\succeq_{lmax}$). It is then enough to lexicographically compare the two new matrices. This preference relation, denoted $\succeq_{lmax(lmin)}$, is a refinement of $\succeq_{max(min)}$ and also $\succeq_{OPT,\pi,u}$.

Example 1.11. *Let us consider Example 1.10. We can compare the representative matrices of f ($F = ((1, 0.3), (0.2, 0.4))$) and g ($G = ((1, 0.3), (0.2, 0.1))$) using $lmax(lmin)$ as follows: We have* $(1, 0.3) \cong_{lmin} (1, 0.3)$ *and* $(0.4, 0.2) \succ_{lmin} (0.2, 0.1)$ *so* $f \succ_{max(min)} g$.

For pessimistic utility, Fargier et Sabbadin have proposed a lexicographic refinement $\succeq_{lmin(lmax)}$ to the π-reverse matrix. More precisely, given a matrix F, it π-reverse matrix is denoted $n\ \ F$ such that: $n\ \ F = ((n(\pi(s_1)), u(s_1)), ..., (n(\pi(s_N)), u(s_N)))$. So, refining u_{pes} leads to the application of leximin(leximax) comparison to π-reverse matrix. However, refining u_{opt} leads to leximax(leximin) comparison directly to the representative vectors.

Finally, they have proved that the ordinal lexicographic refinements are equivalent to probabilistic EU-based refinements defined in Section 1.5.2, More formally:

Theorem 1.5. *For any possibilistic model, it holds that:*

- $\succeq_{EU,\chi^*\circ\pi,\chi^*\circ u} \equiv \succeq_{lmax(lmin)}$
- $\succeq_{EU,\chi^*\circ\pi,\psi^*\circ u} \equiv \succeq_{lmin(lmax)}$

where χ^* and ψ^* are two transformation functions as shown in Section 1.5.2.

Hence, these lexicographic refinements obey to the Sure Thing Principle.

In this thesis, we aim to extend these efficient refinements (leximax(leximin) and leximin(leximax)) to sequential decision-making problems, presented in the next Chapter.

1.6 Summary

In this Chapter, we have presented an overview of probabilistic decision model that is well developed and axiomatized. This framework is appropriate when all numerical data are available or can be elicited from the decision maker, which is not always the case. Possibilistic decision theory offers a flexible and simple framework to represent qualitative uncertainty. We especially detailed main decision criteria based on possibility theory namely optimistic and pessimistic qualitative utilities. These two possibilistic criteria suffer from the drowning effect problem and fail to satisfy the principle of Pareto efficiency, in contrast to the classical numerical criterion— expected utility. [Fargier and Sabbadin, 2003, Fargier and Sabbadin, 2005] show that possibilistic utilities can be refined by an expected utility criterion. Choosing an EU-model is advantageous, since it both leads to an EU refinement of the original rule that overcomes the lack of decisiveness of the possibilistic criteria, and it satisfies the Sure thing principle and the principle of Pareto dominance. These refinements are equivalent to lexicographic procedures in the one stage procedures described in [Fargier and Sabbadin, 2003, Fargier and Sabbadin, 2005].

In next chapter, we will present the foundation of sequential decision problems in possibilistic decision trees and possibilistic Markov decision process where the decision maker should choose a sequence of decisions instead of one decision.

CHAPTRE 2

Possibilistic sequential decision-making models

Contents

2.1 Introduction

In the first chapter, we have been investigating non-sequential decision problems. In real world problems, the decision maker is often facing a succession of actions to be taken over time, i.e. sequential decision problems. In these problems a suitable policy is to be found, that associates a decision to each state of the world. Many graphical models have been proposed to represent such problems, such as influence diagrams [Howard and Matheson, 1984], Markov decision processes [Bellman, 1957b] and decision trees [Raiffa, 1968].

In this work, we are interested in the formalism of decision trees and Markov decision processes since they allow an explicit representation of a sequential decision problem.

In classical sequential decision-making models uncertainty is stochastic and the satisfaction of the decision maker is expressed by a numerical, additive utility function [Raiffa, 1968, Puterman, 1994]. In fact, considering ordinal preferences but remaining within a probabilistic quantification of uncertainty in sequential decision-making has led to quantile-based approaches [Gilbert et al., 2017], to the use of reference points [Weng, 2011] or to approaches by pairwise comparison [Yue

et al., 2012]. In this thesis, we are interested in the main purely ordinal decision graphical models: Possibilistic sequential decision models (e.g. see [Sabbadin et al., 1998, Sabbadin, 1999, Garcia and Sabbadin, 2006, Drougard et al., 2013, Ben Amor et al., 2014, Drougard et al., 2014, Bauters et al., 2016]).

This chapter is organized as follows: Section 2.2 formally defines possibilistic decision trees and reviews existing algorithms to find optimal policies with a reasonable complexity. Section 2.3 gives an overview on possibilistic Markov decision processes and then details the optimization of these graphical models w.r.t. possibilistic optimistic and pessimistic utilities i.e. how to find an optimal policy.

2.2 Possibilistic decision trees ($\Pi\mathcal{DT}$s)

Decision trees ($\mathcal{DT}$s) provide an explicit modeling of sequential decision problems by representing each possible scenario by a path from the root to the leaves of the tree [Raiffa, 1968]. In this Section, we study the optimization problem in possibilistic decision trees, denoted $\Pi\mathcal{DT}$s. where we aim to find an optimal policy w.r.t. a decision criterion: optimistic or pessimistic utilities shown in the previous Chapter.

2.2.1 Definition

A $\mathcal{DT}$ is composed of a graphical component and a numerical one as detailed below. The graphical component of a $\mathcal{DT}$ is a labelled graph $DT = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N} = \mathcal{N}_D \cup \mathcal{N}_C \cup \mathcal{N}_{LN}$ is the set of nodes composed of three three kinds of nodes (see Figure 2.1):

- $\mathcal{N}_D$ is the set of decision nodes (represented by squares);
- $\mathcal{N}_C$ is the set of chance nodes (represented by circles);
- $\mathcal{N}_{LN}$ is the set of leaves, also called utility nodes.

The set $\mathcal{E}$ contains the directed edges between nodes, forming a tree where each edge links a parent node to its child node. For any node $N \in \mathcal{N}$, $Out(N) \subseteq \mathcal{E}$ denotes its outgoing edges, $Succ(N)$ the set of its children nodes and $Succ(N, e)$ the child of N that is reached by edge $e \in Out(N)$.

A $\mathcal{DT}$ represents a sequential decision problem in the following way:

- Leaf nodes correspond to states of the world in which a utility is obtained (for the sake of simplicity we assume that utilities are attached to leaves only); the utility of a leaf node $LN_i \in \mathcal{N}_{LN}$ is denoted $u(LN_i)$.

- Decision nodes correspond to states of the world in which a decision is to be made: $D_i \in \mathcal{N}_D$ represents a decision variable Y_i. Its domain corresponds to the labels a of the edges starting from D_i.These edges lead to chance nodes, i.e. $Succ(D_i) \subseteq \mathcal{N}_C$.
- A state variable X_j is assigned to each chance node $C_j \in \mathcal{N}_C$. Its domain corresponds to the labels x of the edges starting from that node. Each edge starting from a chance node C_j represents an event $X_j = x$. For any $C_j \in \mathcal{N}_C$, $Succ(C_j) \subseteq \mathcal{N}_{LN} \cup \mathcal{N}_D$ i.e. after the execution of a decision, either a leaf node or a decision node is reached.

Given a decision tree $\mathcal{DT}$, $Start(\mathcal{DT})$ denotes the set of its first decision nodes (it is a singleton containing the root of the tree if this root is a decision node, or its successors if the root is a chance node). For the sake of simplicity, we suppose that all the paths from the root to a leaf in the tree have the same length. The horizon of the decision tree, denoted by h, is the number of decision nodes along these paths. The branching factor, denoted by b, is the number of children at each node. Given a node $N \in \mathcal{N}$, we shall also consider the subproblem DT_N defined by the tree rooted in N.

The joint knowledge on the state variables is not given in extenso, but through the labeling of the edges issued from chance nodes. In a possibilistic context, the uncertainty pertaining to the possible outcomes of each X_j is represented by a possibility distribution: each edge starting from a chance node C_j, representing an event $X_j = x$, is endowed with a number $\pi_{C_j}(x)$, the possibility $\pi(X_j = x|past(C_j))$[1]. A possibilistic ordered scale, V, is used to evaluate the utilities and possibilities.

Example 2.1. *Let us suppose that a "Rich and Unknown" person runs a startup company. In every state s/he must choose between Saving money (Sav) or Advertising (Adv) and s/he may then get Rich (R) or Poor (P) and Famous (F) or Unknown (U). Figure 2.1 shows the* $\Pi\mathcal{DT}$ *(with horizon* $h = 2$*) that represents this sequential decision problem. This* $\Pi\mathcal{DT}$ *contains 10 chance nodes* $\mathcal{N}_C = \{C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_9, C_{10}\}$*, 5 decision nodes* $\mathcal{N}_D = \{D_0, D_1, D_2, D_3, D_4\}$ *and 16 leaf nodes* $\mathcal{N}_{LN} = \{LN_1, \ldots, LN_{16}\}$.

Solving a decision tree amounts to building a *policy* (and thus, a succession of chance node) for each reachable decision node. Formally, we define a policy as a function $\delta : \mathcal{N}_D \mapsto A$, where A is the set of possible actions, including a special "undefined" action $\perp$, chosen for decision nodes which are left unexplored by a given policy. $\delta(D_i)$ is the action to be executed when a decision node D_i is reached.

Admissible policies assign a chance node to each reachable decision node, i.e. must be:

- *sound*: $\forall D_i \in \mathcal{N}_D, \delta(D_i) \in Out(D_i) \cup \{\perp\} \subseteq A$, and

[1]As in classical probabilistic decision trees, it is assumed that $\pi(X_j = x|past(C_j))$ only depends on the variables in $past(C_j)$ and often only on the decision made in the preceding node and on the state of the preceding chance node.

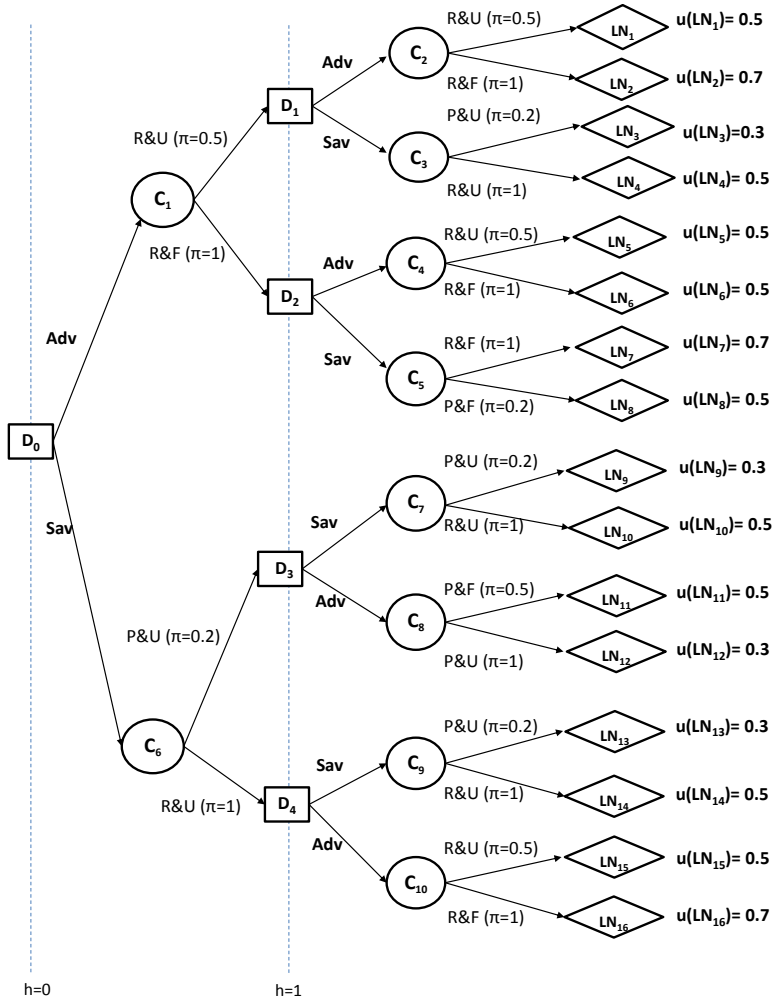


Figure 2.1: The $\Pi\mathcal{DT}$ of Example 2.1

- *complete*:
  (i) $\forall D_i \in Start(\mathcal{DT}), \delta(D_i) \neq \bot$ and
  (ii) $\forall D_i$ s.t. $\delta(D_i) \neq \bot, \forall N \in Succ(Succ(D_i, \delta(D_i)))$ either $\delta(N) \neq \bot$ or $N \in \mathcal{N}_{LN}$.

We denote by Δ_N (or simply Δ, when there is no ambiguity) the set of admissible policies built from a tree rooted in N. Each policy δ in Δ defines a connected subtree of $\mathcal{DT}$, the branches of which represent possible scenarios, or *trajectories*. Formally, a trajectory is a sequence of value assignments to decision and chance variables along a path from a starting decision node (a node in $Start(\mathcal{DT})$) to a leaf:

$$\tau = (a_{j_0}, x_{i_1}, a_{j_1}, \ldots, a_{j_{h-1}}, x_{i_h}),$$

where $Y_0 = a_{j_0}$ is the first decision in the trajectory, x_{i_1} the value taken by its first chance variable, X_{j_0} in this scenario, $Y_{i_1} = a_{j_1}$ is the second decision, etc.

We often identify a policy δ, the corresponding subtree and the set of its trajectories (hence the notation $\tau \in \delta$ to mean that τ is a trajectory of δ). We also consider subtrees of the original $\mathcal{DT}$, and thus sub-policies: let C_j be a chance node, $D_{i_1}, \ldots, D_{i_k}$ its successors and, for $l = 1, k$, the policies $\delta_{i_l} \in \Delta_{D_{i_l}}$ which solve the subproblem rooted in D_{i_l}.

$\delta_{i_1} + \cdots + \delta_{i_k}$ is the policy of Δ_{C_j} resulting from the composition of the δ_{i_l}: $(\delta_{i_1} + \cdots + \delta_{i_k})(N) = \delta_{i_l}(N)$ iff N belongs to the subtree rooted in D_{i_l}.

Example 2.2. *Let us consider the* $\Pi\mathcal{DT}$ *of example 2.1. This* $\Pi\mathcal{DT}$ *encodes 16 trajectories:*

- $\tau_1 = (Adv, R\&U, Adv, R\&U)$,
- $\tau_2 = (Adv, R\&U, Adv, R\&F)$,
- $\tau_3 = (Adv, R\&U, Sav, P\&U)$,
- $\tau_4 = (Adv, R\&U, Sav, R\&U)$,
- $\tau_5 = (Adv, R\&F, Adv, R\&U)$,
- $\tau_6 = (Adv, R\&F, Adv, R\&F)$ *etc.*

The possibilistic evaluation of a policy, as proposed by [Sabbadin et al., 1998], relies on the qualitative optimistic and pessimistic decision criteria axiomatized by [Dubois and Prade, 1995]. The utility of the policy is computed on the basis of the transition possibilities and the utilities of its trajectories. For each trajectory $\tau = (a_{j_0}, x_{i_1}, a_{j_1}, \ldots, x_{i_h})$

- Its utility, $u(\tau)$, is the utility $u(x_{i_h})$ of its leaf x_{i_h}.
- The possibility of τ given that a policy δ is applied from initial node D_0 is defined by:

$$\pi(\tau|\delta, D_0) = \begin{cases} \min\limits_{\pi_k \in \pi_\tau} \pi_k & \text{if } \tau \in \delta, \\ 0 & \text{otherwise.} \end{cases}$$

Following [Dubois and Prade, 1995], [Sabbadin et al., 1998] define as follows , the optimistic and pessimistic utility degrees of a policy $\delta \in \Delta$:

$$u_{opt}(\delta) = \max_{\tau \in \delta} \min(\pi(\tau|\delta, D_0), u(\tau)) \tag{2.1}$$

$$u_{pes}(\delta) = \min_{\tau \in \delta} \max(1 - \pi(\tau|\delta, D_0), u(\tau)) \tag{2.2}$$

This approach is purely ordinal (only min and max operations are used to aggregate the evaluations of the possibility of events and the utility of states). We can check that the preference

orderings $\succeq_O$ between policies, derived either from u_{opt} ($O = u_{opt}$) or from u_{pes} ($O = u_{pes}$), satisfy the principle of weak monotonicity:
$\forall C_j \in \mathcal{N}_C, \forall D_i \in Succ(C_j), \delta, \delta' \in \Delta_{D_i}, \delta" \in \Delta_{Succ(C_j)\setminus D_i}$:

$$\delta \succeq_O \delta' \implies \delta + \delta" \succeq_O \delta' + \delta''$$

where $\delta + \delta"$ (resp. $\delta' + \delta''$) is the policy resulting from the composition δ (resp. δ') with $\delta"$.

2.2.2 Optimization of $\Pi\mathcal{DT}$s: Backward induction

Since optimistic and pessimistic utilities satisfy the crucial property of *weak monotonicity*, an optimal policy can be computed in polytime with respect to the size of the tree (the total number of nodes) using a recursive method of Dynamic programming called backward search method or backward induction method. [Sabbadin et al., 1998, Sabbadin, 2001] have proposed qualitative counterparts of the stochastic dynamic programming algorithm *backwards induction*, denoted $BI\text{-}DT$, for $\Pi\mathcal{DT}$s (see Algorithm 2.1, written in a recursive style) that optimizes the decisions from the leaves of the tree to its root. Note that this algorithm does not generate all the best policies but returns only one among them.

This backwards reasoning procedure is depicted in a recursive manner:

- when a chance node $N \in \mathcal{N}_C$ is reached, optimistic (resp. pessimistic) utility is calculated for each of its children (i.e. each successor decision node or leaf node). Note that the optimistic utility of a leaf node is its utility. The optimistic utility obtained in N is then calculated using the possibility degrees on all successors and their optimistic utility.
- When a decision node $N \in \mathcal{N}_D$ is reached, we select a decision C_j among all the possible ones leading to an optimal sub-policy w.r.t. $\succeq_{u_{opt}}$. The choice is performed by comparing the optimistic utilities obtained in each successor chance node of N.

Decision trees represent sequential decision problems under the assumption of complete observation. However, they have serious limitations in their ability to model complex situations, especially when outcomes or events occur over a long time horizon. As a result, decision trees are often replaced with the use of Markov Decision Processes ($\mathcal{MDP}$s) [Puterman, 1994] which are compact representations of sequential decision problems. They explicitly account for timing of events, whereas time usually is less explicitly accounted in decision trees.

Algorithm 2.1: $BI\text{-}DT$: Backward-Induction-$\Pi\mathcal{DT}$-u_{opt}(N:Node)

Data: A $\Pi\mathcal{DT}$; the policy, δ, is memorized as global variable
Result: Set δ for the tree rooted in N and returns its optimistic utility

```
1  begin
2      // Leaves
3      if N ∈ N_LN then u_opt ← u(N);
4      // Chance nodes
5      if N ∈ N_C then
6          foreach N_i ∈ Succ(N) do
7              u^i_opt ← Backward-Induction-ΠDT − uopt(N_i);
8          u_opt ← max_{N_i ∈ Succ(N)} min(π_N(D_i), u^i_opt);
9      // Decision nodes
10     if N ∈ N_D then
11         u_opt ← 0;
12         foreach C_j ∈ Succ(N) do
13             u ← Backward-Induction-ΠDT-uopt(C_j);
14             if u > u_opt then
15                 u_opt ← u;
16                 δ(N) ← label((N, C_j));
17     return u_opt;
```

2.3 Possibilistic Markov Decision Processes ($\Pi\mathcal{MDP}$s)

Possibilistic Markov decision processes ($\Pi\mathcal{MDP}$s) represents a qualitative version of probabilistic $\mathcal{MDP}$s. In these models, uncertainty about the consequences of actions is represented by possibility distributions and rewards are qualitative [Sabbadin et al., 1998, Sabbadin, 1999, Sabbadin, 2001, Perny et al., 2005].

Solving a $\Pi\mathcal{MDP}$ amounts to finding an 'optimal' action for any state of the world encountered, with respect to optimization criteria: optimistic and pessimistic utilities. In finite-horizon $\Pi\mathcal{MDP}$s an optimal policy can be provided, using a possibilistic backwards induction algorithm, proposed by [Sabbadin, 1999] as an adaptation of the classical stochastic one [Bellman, 1957a,Puterman, 1994]. However, in infinite-horizon $\Pi\mathcal{MDP}$s the most often used algorithms are possibilistic policy iteration algorithm [Sabbadin, 2001] (the possibilistic counterpart of stochastic policy iteration [Howard, 1960]) and possibilistic value iteration algorithm [Sabbadin, 2001] (the possibilistic counterpart of stochastic value iteration [Bellman, 1957a]).

2.3.1 Finite-horizon possibilistic Markov decision processes (Finite-horizon $\Pi\mathcal{MDP}$s)

Finite-horizon $\Pi\mathcal{MDP}$ is a mathematical framework for representing complex multi-stage decision problems with finite time horizon.

2.3.1.1 Definition

A Finite-horizon $\mathcal{MDP}$ [Sabbadin et al., 1998, Sabbadin, 2001] is defined by:

- A finite set of **stages** $T = \{0, \ldots, h\}$. h is called the horizon of the problem.
- Finite **state spaces**, S_t, for each $t = 0 \ldots h$; $S = S_0 \cup ... \cup S_h$ denotes the set of all possible states at every time steps.
- Sets A_s of available **actions** in state $s \in S_t$; $A = A_{S_0} \cup ... \cup A_{S_h}$ denotes the full action space.
- The **rewards** $u(s)$ that are obtained in the final states $s \in S_h$. In this Section we do not consider intermediate satisfaction degrees.

In a possibilistic context, the uncertainty of the agent about the effect of an action a taken in state $s \in S_{t-1}$ is represented by a possibility distribution $\pi(.|s, a) : S_t \to V$. For $s' \in S_t$, $\pi(s'|s, a)$ measures to what extent s' is a plausible consequence of a in s. In the same way, consequences are ordered in terms of levels of satisfaction by a qualitative utility function $u : S_h \to V$.

Such a $\Pi\mathcal{MDP}$ can be represented by a labeled graph, where states are represented by circles, actions by squares and each final state has an attached utility. Each edge linking an action to a state denotes a transition, and is labeled by the possibility of that transition given the action is executed.

$\mathcal{DT}$s and finite-horizon $\mathcal{MDP}$ are two close frameworks. A decision tree is a particular finite-horizon Markov decision process, up to the notations. Decision nodes are states, and the chance nodes that follow a decision node are the actions available in this state.

Example 2.3. *Let us consider the problem introduced in Example 2.1 - it is possible to represent it as a Finite-horizon $\Pi\mathcal{MDP}$. Figure 2.2 represents the $\Pi\mathcal{MDP}$, in the form of an acyclic graph, when the horizon $h = 2$ (here, utilities are also shown).*
We have: $S_0 = \{R\&U_0\}$, $S_1 = \{R\&F_1, R\&U_1, P\&U_1\}$ and $S_2 = \{R\&F_2, R\&U_2, P\&F_2, P\&U_2\}$ also $\forall t = 0, 2, A_{s \in S_t} = \{Adv, Sav\}$.

A policy in a Finite-horizon $\mathcal{MDP}$ is a function that maps each state to an admissible action $\delta : S \to A$, s.t. $\delta(s) \in A_s$. When applied from a state $s_{i0} \in S_0$, such a policy defines a list of

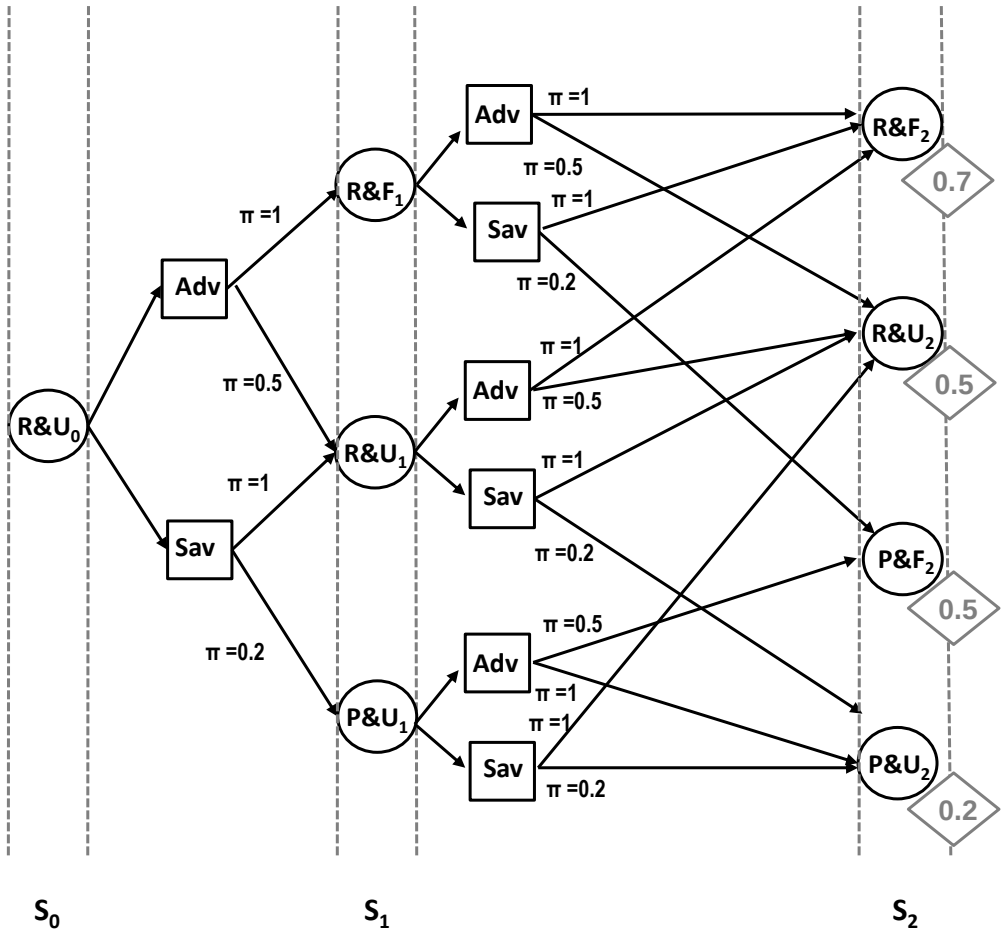


Figure 2.2: The finite-horizon $\Pi\mathcal{MDP}$ ($h = 2$) of Example 2.3

trajectories, as for the decision trees case. A trajectory τ is a sequence of actions and states along a path following (and excluding) a first state s_{i0} to a final state $s_{ih} \in S_h$. Formally,

$$\tau = (a_{j0}, \ldots, s_{ik}, a_{jk}, \ldots, s_{ih}),$$

where $s_{ik} \in S_k$ and $a_{jk} = \delta(s_{ik})$. We suppose, without loss of generality, that all trajectories have the same length h. If a trajectory has shorter length than h, neutral elements (0 or 1 as appropriate) are added at the end.

Note that s_{i0} is not part of the trajectory but given alongside the $\mathcal{MDP}$ model. a_{j0} is the first action in the trajectory - the one prescribed by δ for s_{i0}-etc. The reward associated to τ is the utility $u(s_{ih})$ obtained in the final state of the trajectory s_{ih}.

The evaluation of a policy δ in state s_0 using qualitative pessimistic utility is defined by the qualitative $minmax$ expectation of the degrees of satisfaction of the final states of the possible trajectories, and the optimistic utility as the $maxmin$ expectation of the same:

$$u_{pes}(s_0, \delta) = \min_{\tau \in \delta} \max\{1 - \pi(\tau|s_0, \delta), u(s_{ih})\} \tag{2.3}$$

$$u_{opt}(s_0, \delta) = \max_{\tau \in \delta} \min\{\pi(\tau|s_0, \delta), u(s_{ih})\} \tag{2.4}$$

The possibility $\pi(\tau|s_0, \delta)$ of τ given that policy δ is applied from initial state s_0 is defined by:

$$\pi(\tau|s_0, \delta) = \begin{cases} \min\limits_{\pi_k \in \pi_\tau} \pi_k & \text{if } \tau \in \delta, \\ 0 & \text{otherwise.} \end{cases}$$

These criteria can be optimized by choosing, for each state, an action that maximizes the following counterparts of the Bellman equations [Sabbadin, 2001]:

- In the pessimistic case

$$\begin{cases} u_{pes}(s) = \max_{a \in A_s} \min\{u(s), \min\limits_{s' \in S_{t+1}} \max\{1 - \pi(s'|s, a), u_{pes}^{t+1}(s')\} \forall t < h, s \in S_t \\ \\ u_{pes}(s) = u(s) \ \ \forall s \in S_h \end{cases} \tag{2.5}$$

- In the optimistic case:

$$\begin{cases} u_{opt}(s) = \max_{a \in A_s} \min\{u(s), \max\limits_{s' \in S_{t+1}} \min(\pi(s'|s, a), u_{opt}(s'))\} \forall t < h, s \in S_t \\ \\ u_{opt}(s) = u(s) \ \ \forall s \in S_h. \end{cases} \tag{2.6}$$

2.3.1.2 Optimization of finite-horizon $\Pi\mathcal{MDP}$s: Backward Induction

[Sabbadin et al., 1998, Sabbadin, 1999] have shown that any policy computed backwards by successive applications of Equation 2.5 (resp. 2.6) is optimal according to u_{pes} (resp. u_{opt}).

The principle of the optimistic version of this algorithm denoted $BI\text{-}MDP$ (Algorithm 2.2, the pessimistic version is similar) can be described as follows:

- Envision being in the last time stage, for all the possible states, it decides the best action for each state by calculating the maximal optimistic utility of that state,
- Then, suppose being in the next-to-last stage, for all the possible states, it decides the best action for each state, given we know the optimal optimistic utility of being in various states at the next time stage,
- This process is continued until reaching the present time stage.

Algorithm 2.2: $BI\text{-}MDP$: Backward-Induction-$\Pi\mathcal{MDP}$-u_{opt}

Data: A $\Pi\mathcal{MDP}$
Result: Computes and returns an optimal policy δ

begin
  $t \leftarrow h$;
  for $s \in S_h$ **do** $u_{opt}(s) \leftarrow u(s)$;
  while $t \geq 1$ **do**
    $t \leftarrow t - 1$;
    foreach $s \in S_t$ **do**
      $u_{opt}(s) \leftarrow \max_{a \in A_s} \max_{s' \in S_{t+1}} \min\{\pi(s'|s,a), u_{opt}(s')\}$;
      $\delta(s) \leftarrow \arg\max_{a \in A_s} \left\{ \max_{s' \in S_{t+1}} \min\{\pi(s'|s,a), u_{opt}(s') \right\}$;
  return δ;

2.3.2 Stationary Possibilistic Markov decision processes (stationary $\Pi\mathcal{MDP}$s)

The previous section was devoted to the finite-horizon version of the $\mathcal{MDP}$ framework. In some situations, we do not know when the process will end, or to control the system forever: the $\mathcal{MDP}$ problem has to be expressed whatever the horizon h, or, more generally, for an infinite horizon. In the present Section, we consider stationary problems, i.e. problems in which the set of states, the available actions and the transition functions do not depend on the stage of the problem.

2.3.2.1 Definition

A stationary possibilistic Markov Decision Process (stationary $\Pi\mathcal{MDP}$) [Sabbadin, 2001] is defined by:

- A finite set S of **states**.

- A finite set A of **actions**, A_s denotes the set of actions available in state s;
- A utility function u: $u(s)$ is the intermediate satisfaction degree obtained in state $s \in S$.

The uncertainty about the effect of an action a taken in state s, i.e. the transition function, is represented by possibility distribution $\pi(.|s,a)$: for any $s' \in S$, $\pi(s'|s,a)$ describing the uncertainty about the possible next state s' when the current state is s and the action taken is a (and then getting the associated reward $u(s') \in V$, the utility of being in state s').

Example 2.4. *Let us consider Figure 2.3 that shows a stationary* $\Pi\mathcal{MDP}$ *that captures the problem of Example 2.1, formally described as follows:*

- $S = \{R\&U,\ R\&F, P\&U\}$,
- $A_{R\&U} = \{Adv, Sav\},\ A_{R\&F} = \{Sav\},\ A_{P\&U} = \{Sav\}$,
- $\pi(P\&U|R\&U, Sav) = 0.2,\ \pi(R\&U|R\&U, Sav) = 1,\ \pi(R\&F|R\&U, Adv) = 1,$ $\pi(R\&F|R\&F, Sav) = 1,\ \pi(R\&U|R\&F, Sav) = 1$,
- $u(R\&U) = 0.5, u(R\&F) = 0.7,\ u(P\&U) = 0.3$.

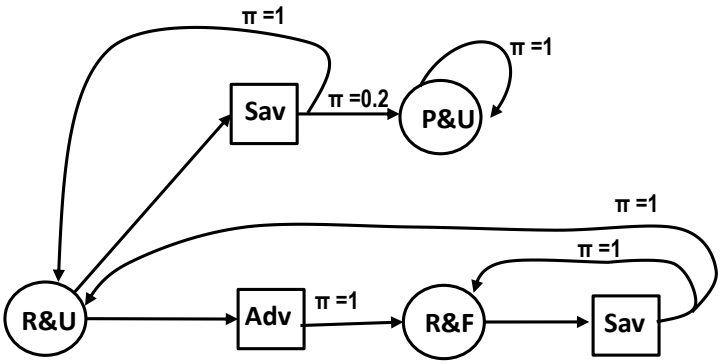


Figure 2.3: The stationary $\Pi\mathcal{MDP}$ of Example 2.4

Solving a stationary $\Pi\mathcal{MDP}$ consists in finding a (stationary) policy, i.e. a function $\delta : S \rightarrow A_s$, mapping states to actions, which is optimal with respect to a decision criterion. In the possibilistic case, as in the probabilistic case, the idea is to compute the value of a policy from the utility and the likelihood of its trajectories. Formally, let Δ be the set of all policies encoded by a possibilistic MDP. First, when the horizon is finite, each $\delta \in \Delta$ defines a list of scenarios called trajectories. Each trajectory τ is a sequence of states and actions that can be written as follows:

$$\tau = (s_0, a_0, s_1, \ldots, s_{h-1},\ a_{h-1}, s_h),$$

where s_i $(i = 0..h)$ is the i^{th} state in the trajectory (i.e. $s_i \in S$ at stage $t = i$), a_i $(i = 0..h-1)$ is the i^{th} action in the trajectory i.e. $a_i \in A_S$ at stage $t = i$ $(i = 0..h-1)$.

As in the case of finite-horizon $\Pi\mathcal{MDP}$, the possibility (resp. the utility) of τ given that policy δ is applied from s_0 is expressed by:

$$\pi(\tau|s_0, \delta) = \min_{i=1..h} \pi(s_i|s_{i-1}, \delta(s_{i-1})) \tag{2.7}$$

$$u(\tau) = \min_{i=0..h} u(s_i) \tag{2.8}$$

Optimistic and pessimistic utilities criteria [Sabbadin et al., 1998] can be optimized by choosing, for each state, an action that maximizes the following counterparts of the Bellman equations [Bellman, 1957a, Sabbadin, 2001]:

$$u_{opt}(s) = \max_{a \in A_s} \min\{u(s), \max_{s' \in S} \min(\pi(s'|s, a), u_{opt}(s'))\} \tag{2.9}$$

$$u_{pes}(s) = \max_{a \in A_s} \min\{u(s), \min_{s' \in S} \max(1 - \pi(s'|s, a), u_{pes}(s'))\} \tag{2.10}$$

This formulation is more general than the first one (finite-horizon $\Pi\mathcal{MDP}$) in the sense that it applies to both the finite and the infinite case. It has allowed the definition of a (possibilistic) *value iteration* algorithm and a *policy iteration* algorithm which converges to an optimal policy in polytime.

2.3.2.2 Optimization of stationary $\Pi\mathcal{MDP}$s: Value iteration

In [Sabbadin, 2001], Sabbadin assumes the existence of an additional action $stay$ that keeps the system in the same state (or equivalently, an action $do\text{-}nothing$ if the system does not evolve by itself without any action applied). Under this assumption, possibilistic counterpart of the value iteration algorithm is defined, denoted $VI\text{-}MDP$. It computes optimal policies from iterated modifications of possibilistic value function $\tilde{Q}_{opt}(s, a)$ (resp. $\tilde{Q}_{pes}(s, a)$) that evaluates the optimistic utility (resp. the pessimistic utility) of performing a in s.

The optimal possibilistic policy can be obtained from the solution of dynamic programming equations expressed by:

- In the optimistic case:

$$\tilde{Q}_{opt}(s, a) = \max_{s' \in S} \min\{(u(s'), \pi(s'|s, a)), u_{opt}(s')\} \tag{2.11}$$

where $u_{opt}(s) = \max_a Q_{opt}(s, a)$ and $Q_{opt}(s, stay) = u(s)$.

- In the pessimistic case:

$$\tilde{Q}_{pes}(s,a) = \min_{s' \in S} \min\{u(s'), \max\{(1 - \pi(s'|s,a)), u_{pes}(s')\}\} \tag{2.12}$$

where $u_{pes}(s) = \max_a Q_{pes}(s,a)$ and $Q_{pes}(s, stay) = u(s)$.

Therefore, a possibilistic version of the Value Iteration algorithm (Algorithm 2.3), that computes $\tilde{Q}_{opt}(s,a)$ or $\tilde{Q}_{pes}(s,a)$, has been defined [Sabbadin, 2001]. This algorithm converges to the actual value of $\tilde{Q}_{opt}$ (resp. $\tilde{Q}_{pes}$) in a finite number of steps.

Algorithm 2.3: $VI\text{-}MDP$: Possibilistic (Optimistic) Value iteration

```
Data: A stationary Π𝒟𝒫
Result: Computes and returns an optimal policy δ
1  begin
2      foreach s ∈ S do
3          u_opt(s) ← u(s)
4      repeat
5          foreach s ∈ S do
6              foreach a ∈ A do
7                  Q(s,a) ← max_{s'∈S} min{(π(s'|s,a), u_opt(s')};
8                  u_opt(s) ← max_a Q(s,a);
9                  δ(s) = arg max_a Q(s,a);
10     until Q converges to Q̃_opt;
11     // Q is epsilon close to Q̃_opt
12     return δ;
```

2.3.2.3 Optimization of stationary $\Pi\mathcal{MDP}$s: Policy iteration

R. Sabbadin has proposed a possibilistic policy iteration algorithm (Algorithm 2.4), denoted here $PI\text{-}MDP$, that alternates evaluation and improvement phases, as for its stochastic counter-part [Sabbadin, 2001] (a pessimistic counterpart of Algorithm 2.4 is obtained by the use of the pessimistic utility evaluations, instead of the optimistic ones). First of all, this algorithm chooses an initial policy arbitrary, then two steps follow:

- policy evaluation: it calculates the optimistic utility of each state given the current policy δ until convergence,
- policy improvement: it updates the policy if an improvement is possible.

The stopping criterion for the possibilistic policy iteration algorithm is the equality of the optimistic values of two successive policies δ and δ' (Line 14).

Algorithm 2.4: $PI\text{-}MDP$: Possibilistic (Optimistic) Policy iteration

Data: A stationary $\Pi\mathcal{MDP}$
Result: Computes and returns an optimal policy δ^*

```
begin
    // Arbitrary initialization of δ on S
    foreach s ∈ S do
        δ(s) ← "Stay"
    repeat
        // Evaluation of δ;
        repeat
            foreach s ∈ S do
                u^δ_opt(s) = max_{s'∈S} min{π(s'|s,a).u^δ_opt(s')};
        until u^δ_opt converges;
        // Improvement of δ;
        foreach s ∈ S do
            δ(s) = arg max_{a∈A} max_{s'∈S} min{π(s'|s,a).u^δ_opt(s')};
    until δ converges to δ*;
    // Stabilization of the optimistic value of δ
    return δ*;
```

2.4 Summary

In this chapter we have proposed a short review of the possibilistic sequential decision-making based on the possibilistic counterparts of decision trees and Markov decision processes. First, we have presented possibilistic decision trees that offer a natural and explicit model to handle possibilistic sequential decision problems. We have studied the optimization of policies for optimistic or pessimistic utilities, in such models, using backward induction algorithm. Besides, we have focused on the possibilistic Markov Decision Processes framework, and the three algorithms for optimizing possibilistic utilities criteria: Backward induction, value iteration and policy iteration.

In the next Chapter, we detail the problem of *drowning effect* when comparing policies in sequential problems. We define lexicographic refinements that compare full policies, and not simply their reductions. Chapter 4 provides a backward induction algorithm to compute a lexicographic optimal policy in possibilistic decision trees as well as a backward induction algorithm to optimize possibilistic (finite-horizon) Markov decision processes. We show also that these refinements can be represented by *infinitesimal* expected utilities (Chapter 5). The case of stationary $\Pi\mathcal{MDP}$ is handled in Chapter 6.

CHAPTRE 3

Extending Lexicographic refinements to possibilistic sequential decision-making

Contents

3.1 Introduction

Possibilistic decision criteria, especially pessimistic and optimistic utilities, are simple and realistic as presented in Chapter 1, but they have some shortcomings: the principle of Pareto efficiency is violated since these criteria suffer from the drowning effect [Fargier and Sabbadin, 2005].

In order to overcome the drowning effect, some refinements of possibilistic decision criteria have been proposed in the non-sequential case i.e. one-step decision case (see Chapter 1). In particular, [Fargier and Sabbadin, 2003, Fargier and Sabbadin, 2005] have proposed lexicographic refinements of possibilistic utilities which can be represented by a form of expected utility. But, these refinements are limited to one-step decision problems and do not apply to the sequential decision problems that interest us. The present Chapter provides an extension of lexicographic refinements to sequential decision-making, in order to apply them to decision trees and Markov decision processes.

This chapter is structured as follows: Section 3.2 exposes the problem of drowning effect

when optimizing possibilistic utilities criteria in sequential decision-making models. Section 3.3 proposes a way to compare policies using lexicographic procedures: first we detail the case of evaluating decision models with utilities on final stage i.e. $\Pi\mathcal{DT}$s and finite-horizon $\Pi\mathcal{MDP}$s, and then the case of models with intermediate utilities i.e. stationary $\Pi\mathcal{MDP}$s.

3.2 Drowning effect in possibilistic sequential decision-making

As mentioned in Chapter 1, the pessimistic and optimistic utilities present a severe drawback, known as the "drowning effect", due to the use of idempotent operations.

In this Section, we first illustrate the drowning effect of qualitative utilities, as well as its consequences, in possibilistic decision trees and finite-horizon possibilistic Markov decision processes. Then, we focus on stationary possibilistic Markov decision processes. When possibilistic qualitative utilities are used, two policies that give an identical and extreme utility (either good, for u_{opt} or bad, for u_{pes}) in some plausible trajectory, may be undistinguished although having given significantly different consequences in other possible trajectories. As shown by the following counter-example.

Counter-example 3.1. *Let us consider $\Pi\mathcal{DT}$ of Example 2.1 (we recall it in Figure 3.1).*

Let δ and δ' be the two policies defined by:

- $\delta(D_0) = Adv; \delta(D_1) = Adv; \delta(D_2) = Adv$.
- $\delta'(D_0) = Adv; \delta'(D_1) = Sav; \delta'(D_2) = Adv$,

We can check that δ has 4 trajectories ($\tau_1, \tau_2, \tau_5, \tau_6$):

- $\tau_1 = (Adv, R\&U, Adv, R\&U)$ *with* $\pi(\tau_1|\delta, D_0) = 0.5$ *and* $u(\tau_1) = 0.5$,
- $\tau_2 = (Adv, R\&U, Adv, R\&F)$ *with* $\pi(\tau_2|\delta, D_0) = 0.5$ *and* $u(\tau_2) = 0.7$,
- $\tau_5 = (Adv, R\&F, Adv, R\&U)$ $\pi(\tau_5|\delta, D_0) = 0.5$ *and* $u(\tau_5) = 0.5$,
- $\tau_6 = (Adv, R\&F, Adv, R\&F)$ $\pi(\tau_6|\delta, D_0) = 1$ *and* $u(\tau_6) = 0.5$,

Using Equation 2.1 and 2.2, we get:

- $u_{opt}(\delta) = max(min(0.5, 0.5), min(0.5, 0.7), min(0.5, 0.5), min(1, 0.5)) = 0.5$,
- $u_{pes}(\delta) = min(max(0.5, 0.5), max(0.5, 0.7), max(0.5, 0.5), max(0, 0.5)) = 0.5$.

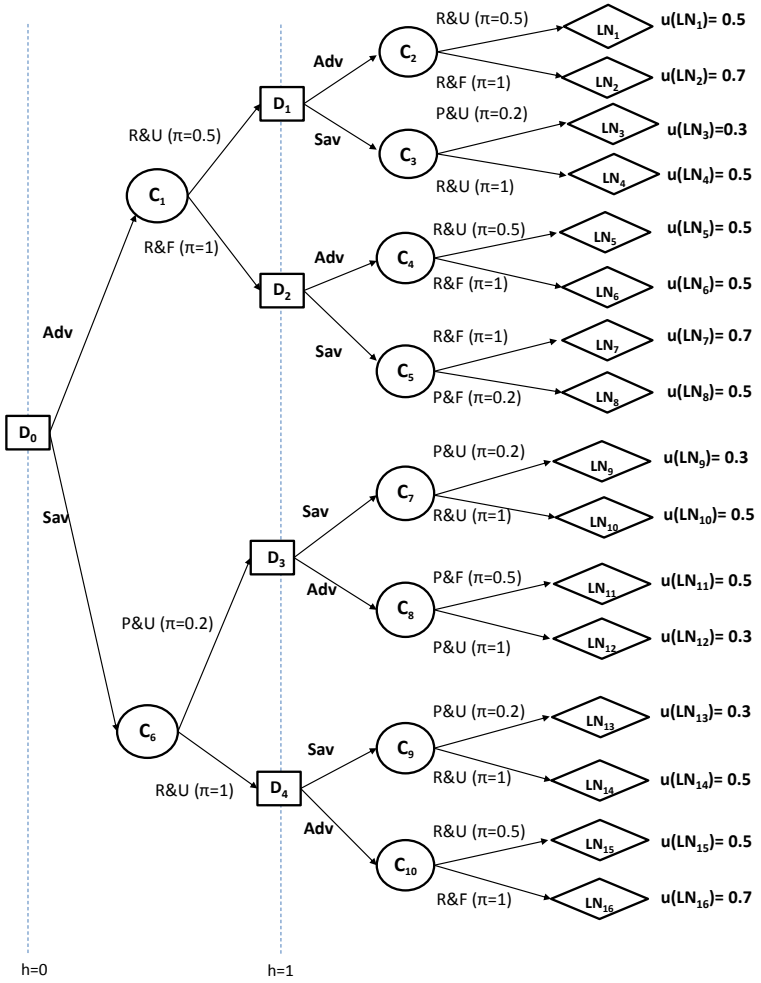


Figure 3.1: The $\Pi\mathcal{DT}$ of Counter-example 3.1

The policy δ' has also 4 trajectories ($\tau_3, \tau_4, \tau_5, \tau_6$):

- $\tau_3 = (Adv, R\&U, Sav, P\&U)$ *with* $\pi(\tau_3|\delta', D_0) = 0.2$ *and* $u(\tau_3) = 0.3$,
- $\tau_4 = (Adv, R\&U, Sav, R\&U)$ *with* $\pi(\tau_4|\delta', D_0) = 0.5$ *and* $u(\tau_2) = 0.5$,
- $\tau_5 = (Adv, R\&F, Adv, R\&U)$ *with* $\pi(\tau_5|\delta', D_0) = 0.5$ *and* $u(\tau_5) = 0.5$,
- $\tau_6 = (Adv, R\&F, Adv, R\&F)$ *with* $\pi(\tau_6|\delta', D_0) = 1$ *and* $u(\tau_6) = 0.5$,

Hence, we have:

- $u_{opt}(\delta') = max(min(0.2, 0.3), min(0.5, 0.5), min(0.5, 0.5), min(1, 0.5)) = 0.5$,

- $u_{pes}(\delta) = min(max(0.8, 0.3), max(0.5, 0.5), max(0.5, 0.5), max(0, 0.5)) = 0.5$.

Thus $u_{opt}(\delta) = u_{opt}(\delta')$ and $u_{pes}(\delta) = u_{pes}(\delta')$: δ, which provides at least utility 0.5 in all trajectories, is not preferred to δ' that provides clearly a bad utility (0.3) in some non-impossible trajectory (τ_3). τ_4, which is good and totally possible "drowns" the bad consequence of δ' in τ_3 in the optimistic comparison; in the pessimistic one, the bad utility of τ_3 is drowned by its low possibility, hence a global degree $u_{pes}(\delta')$ that is equal to the one of δ (which, once again, guarantees a utility degree of 0.5 at least).

The two possibilistic optimistic and pessimistic utilities thus may fail to satisfy the principle of *Pareto efficiency*, which may be written as follows:

Definition 3.1. *(Pareto efficiency)*
For any optimization criterion O (here u_{pes} or u_{opt}),$\forall \delta, \delta' \in \Delta$,

$\delta \succ_O \delta'$ if:
(i) $\forall N \in Common(\delta, \delta'), \delta_N \succeq_O \delta'_N$ and
(ii) $\exists N \in Common(\delta, \delta'), \delta_N \succ_O \delta'_N$.

where $Common(\delta, \delta')$ is the set of situations (decision nodes in $\mathcal{DT}$s, states in the $\mathcal{MDP}$ framework) for which both δ and δ' provide an action and δ_N (resp. δ'_N) is the restriction of δ (resp. δ') to the subtree rooted in N.

Moreover, neither u_{opt} nor u_{pes} fully satisfy the classical, *Strict monotonicity principle*, that can be written as follows:

Definition 3.2. *(Strict monotonicity)*
For any optimization criterion O (here u_{pes} or u_{opt}),
$\forall C_j \in \mathcal{N}_C, D_i \in Succ(C_j), \delta, \delta' \in \Delta_{D_i}, \delta" \in \Delta_{Succ(C_j) \setminus D_i}$,

$$\delta \succeq_O \delta' \iff \delta + \delta" \succeq_O \delta' + \delta".$$

It may, indeed, happen that $u_{pes}(\delta) > u_{pes}(\delta')$ while $u_{pes}(\delta + \delta") = u_{pes}(\delta' + \delta")$ (or that $u_{opt}(\delta) > u_{opt}(\delta')$ while $u_{opt}(\delta + \delta") = u_{opt}(\delta' + \delta")$), as shown in the following Counter-example.

Counter-example 3.2. *Let us consider three policies δ, δ' and δ'', represented with the following simple possibilistic lotteries, respectively:*

- $L_\delta = \langle 1/0.5, 1/0.4 \rangle$,
- $L_{\delta'} = \langle 1/0.4, 1/0.4 \rangle$,
- $L_{\delta''} = \langle 1/0.5, 1/0.1 \rangle$.

Let us consider $L_1 = \langle 1/L_\delta, 1/L_{\delta''} \rangle$ *and* $L_2 = \langle 1/L_{\delta'}, 1/L_{\delta''} \rangle$*. Using the reduction of compound lotteries (see Section 1.3.2), we get:*

- $Reduction(L_1) = \langle 1/0.5, 1/0.4, 1/0.1 \rangle$ *and*
- $Reduction(L_2) = \langle 1/0.4, 1/0.5, 1/0.1 \rangle$.

Then using Equation 1.8, we have $u_{opt}(L_\delta) = 0.5$ *and* $u_{opt}(L_{\delta'}) = 0.4$ *i.e.* $\delta \succ_{u_{opt}} \delta'$.

But $u_{opt}(Reduction(L_1)) = 0.5$ *and* $u_{opt}(Reduction(L_2)) = 0.5$*, i.e.* $\delta + \delta'' \equiv_{u_{opt}} \delta' + \delta''$*. This contradicts the strict monotonicity property.*

Now, let us present an example that shows the drowning effect in finite-horizon $\Pi\mathcal{MDP}$s.

Counter-example 3.3. *Figure 3.2 shows the finite-horizon* $\Pi\mathcal{MDP}$*s (with the horizon* $h = 2$*) that represents an adaptation of the decision problem of Example 2.1. Note that here being Poor and Unknown is an absorbing state (i.e. a state that, once entered, cannot be left).*

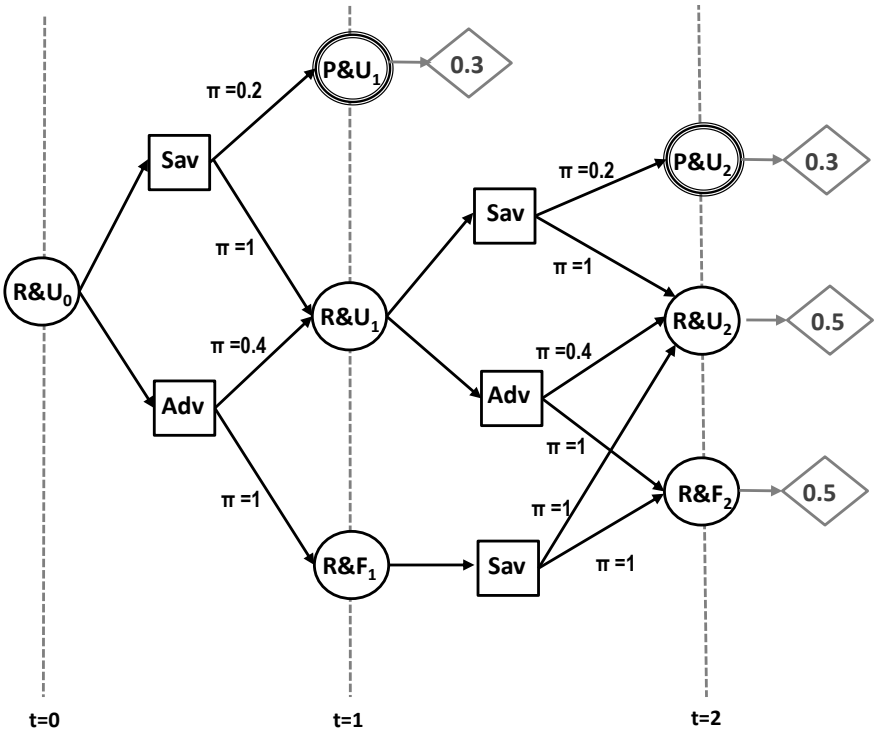


Figure 3.2: The finite-horizon $\pi\mathcal{MDP}$ of Counter-example 3.3

Let us consider the following two policies, δ *and* δ'*:*

- $\delta(R\&U_0) = Adv; \delta(R\&U_1) = Sav; \delta(R\&F_1) = Sav,$

- $\delta'(R\&U_0) = Adv;\ \delta'(R\&U_1) = Adv; \delta'(R\&F_1) = Sav.$

δ has 4 trajectories, $\tau_1, \tau_2, \tau_5, \tau_6$ with:

- $\tau_1 = (Adv, R\&U, Sav, P\&U_2)$ *with* $\pi(\tau_1|R\&U, \delta) = 0.2$ *and* $u(\tau_1) = 0.3$,
- $\tau_2 = (Adv, R\&U_1, Sav, R\&U_2)$ *with* $\pi(\tau_2|R\&U, \delta) = 0.4$ *and* $u(\tau_2) = 0.5$,
- $\tau_5 = (Adv, R\&F_1, Sav, R\&U_2)$ *with* $\pi(\tau_5|R\&U, \delta) = 1$ *and* $u(\tau_5) = 0.5$;
- $\tau_6 = (Adv, R\&F_1, Sav, R\&F_2)$ *with* $\pi(\tau_6|R\&U, \delta) = 1$ *and* $u(\tau_6) = 0.5$.

Hence $u_{opt}(\delta) = 0.5$.

δ' is also composed of 4 trajectories ($\tau_3, \tau_4, \tau_5, \tau_6$) each leading to utility 0.5.
Hence $u_{opt}(\delta') = 0.5$.

Thus $u_{opt}(\delta) = u_{opt}(\delta')$. *However δ' seems better than δ since it provides utility* 0.5 *for sure while δ provides a bad utility (*0.3*) in some non impossible trajectory (τ_1). τ_2, which is good and totally possible "drowns" the preference for δ' in the optimistic comparison.*

In the pessimistic case, δ and δ' are still equivalent ($u_{pes}(\delta) = u_{pes}(\delta') = 0.5$) and the pessimistic criterion is not able to pick up the best one i.e. that is δ'.

We finally provide a counter-example that exemplifies the drowning effect in stationary $\Pi\mathcal{MDP}$s.

Counter-example 3.4. *Consider the stationary $\Pi\mathcal{MDP}$s of Example 2.4 (we recall it in Figure 3.3); recall that $u(R\&U) = 0.5, u(R\&F) = 0.7, u(P\&U) = 0.3$. It admits two policies δ and δ':*

- $\delta(R\&U) = Sav; \delta(P\&U) = Stay; \delta(R\&F) = Sav;$
- $\delta'(R\&U) = Adv; \delta'(P\&U) = Stay; \delta'(R\&F) = Sav;$

For horizon $E = 2$, δ has 3 trajectories (τ_1, τ_2, τ_3) and δ' has 2 trajectories (τ_4, τ_5) such that:

- $\tau_1 = (R\&U, Sav, P\&U, Stay, P\&U)$ *with* $\pi(\tau_1|R\&U, \delta) = 0.2$ *and* $u(\tau_1) = 0.3$,
- $\tau_2 = (R\&U, Sav, R\&U, Sav, P\&U)$ *with* $\pi(\tau_2|R\&U, \delta) = 0.2$ *and* $u(\tau_2) = 0.3$,
- $\tau_3 = (R\&U, Sav, R\&U, Sav, R\&U)$ *with* $\pi(\tau_3|R\&U, \delta) = 1$ *and* $u(\tau_3) = 0.5$,
- $\tau_4 = (R\&U, Adv, R\&F, Sav, R\&F)$ *with* $\pi(\tau_4|R\&U, \delta') = 1$ *and* $u(\tau_4) = 0.5$,
- $\tau_5 = (R\&U, Adv, R\&F, Sav, R\&U)$ *with* $\pi(\tau_5|R\&U, \delta') = 1$ *and* $u(\tau_5) = 0.5$.

Thus $u_{opt}(\delta) = u_{opt}(\delta') = 0.5$ *although δ' provides a good utility* 0.5 *for sure while δ provides a bad utility (*0.3*) in some non impossible trajectories (τ_1 and τ_2). τ_3 which is good and totally possible "drowns" τ_1 and τ_2, thus δ is considered as good as δ'.*

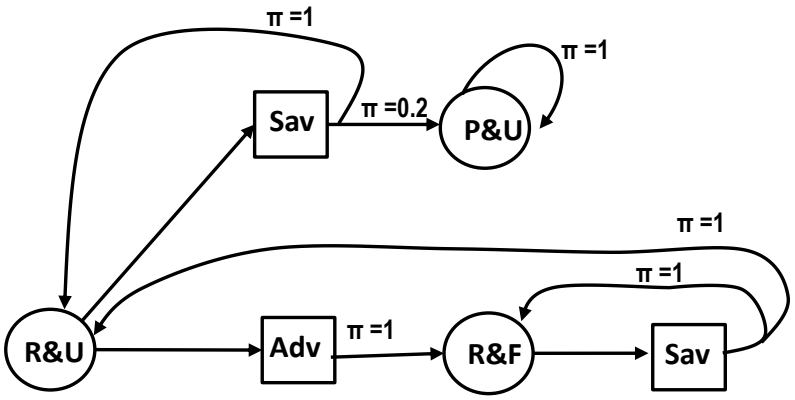


Figure 3.3: The stationary $\pi\mathcal{MDP}$ of Counter-example 3.4

3.3 Lexicographic refinements in sequential decision-making problems

As we have seen in Chapter 1, [Fargier and Sabbadin, 2003,Fargier and Sabbadin, 2005] have proposed lexicographic refinements in order to overcome the drowning effect of possibilistic criteria in one-stage decision problems. However, these refined criteria cannot be used in *sequential* decision problems, where the drowning effect is also due to the reduction of compound possibilistic policies into simple possibility distributions or into numbers i.e. optimistic or pessimistic utilities.

The purpose of the present work is to build efficient preference relations on policies, that agree with the qualitative utilities when the latter can make a decision, and break ties when not - to build refinements that satisfy the principle of Pareto efficiency.

Formally, in sequential decision framework, a preference relation $\succeq'$ refines a preference relation $\succeq$ if and only if whatever δ, δ', if $\delta \succ \delta'$ then $\delta \succ' \delta'$.

In this section, we propose an extension of the lexicographic refinements to policies in sequential decision models, i.e. $\Pi\mathcal{DT}$s and $\Pi\mathcal{MDP}$s. We will first consider the case where we do not have intermediate utilities, then the case of policies with intermediate utilities.

3.3.1 Problems without intermediate utilities

A straightforward way of applying lexicographic comparisons to sequential decision problem is to associate to any policy δ the possibility distribution that it induces on the utility rewards (i.e. the reduction of δ), as usually done in possibilistic (and probabilistic) $\mathcal{DT}$s.

First, for any policy δ (or a sub policy) and any of its trajectories, $\tau = (a_{j_0}, x_{i_1}, a_{j_1}, \ldots, x_{i_h})$

in a $\Pi\mathcal{DT}$ or $\tau = (a_{j0}, ..., s_{ik}, a_{jk}, s_{ih})$ in a finite-horizon $\Pi\mathcal{MDP}$s, we associate a vector defined by:

$$\pi_\tau = (\pi_1, \ldots, \pi_h, u_h).$$

This vector gathers the possibility and utility degrees encountered on the trajectory, formally:

- For the case of $\Pi\mathcal{DT}$s, u_h is the utility $u(x_{i_h})$ of the leaf of τ and $\pi_k = \pi_{C_{j_{k-1}}}(x_{i_k})$ (where $\pi_{C_{j_{k-1}}}$ is the possibility distribution at chance node $C_{j_{k-1}}$) is the possibility of x_{i_k} given that action $a_{j_{h-1}}$ is executed.
- For the case of $\Pi\mathcal{MDP}$s, u_h is the utility $u(s_{ih})$ obtained in the final state of the trajectory s_{ih}. $\pi_k = \pi(s_{ik}|s_{ik-1}, a_{jk-1})$ denotes the possibility degree to reach s_k applying action a_{jk-1} from state s_{ik-1}.

Since a policy δ can be seen as a compound lottery, following the reduction of compound lotteries procedure (see Equation 1.5), it is possible to reduce δ to a distribution π_δ on the utility degrees (i.e a simple lottery) defined by:

$$\pi_\delta(u) = \max_{\pi_\tau, \tau \in \delta \text{ and } u_h = u} \; \min_{\pi_k \in \pi_\tau} \pi_k.$$

This principle of reduction is used, when qualitative decision theory is considered, by [Sabbadin et al., 1998, Sabbadin, 2001] to compare policies: the pessimistic (resp. optimistic) utility of a policy is simply the one of its reduction. Because the π_δ are single stepped, one can think on applying lexicographic comparisons as such, and can write:

$$\delta \trianglerighteq_{lmax(lmin)} \delta' \quad \text{iff} \quad \pi_\delta \trianglerighteq_{lmax(lmin)} \pi_{\delta'},$$
$$\delta \trianglerighteq_{lmin(lmax)} \delta' \quad \text{iff} \quad \pi_\delta \trianglerighteq_{lmin(lmax)} \pi_{\delta'}.$$

$\trianglerighteq_{lmax(lmin)}$ (resp. $\trianglerighteq_{lmin(lmax)}$) refines $\succeq_{u_{opt}}$ (resp. $\succeq_{u_{pes}}$), but neither $\trianglerighteq_{lmax(lmin)}$ nor $\trianglerighteq_{lmin(lmax)}$ do satisfy the principle of Pareto efficiency, as shown by the following counter-example.

Counter-example 3.5. *Consider the $\Pi\mathcal{DT}$ in Figure 3.4 that is a modified version of the problem of Example 2.1.*

Let us consider the two policies δ and δ' defined by:

- $\delta(D_0) = Adv, \delta(D_1) = Sav, \delta(D_2) = Adv,$
- $\delta'(D_0) = Adv, \delta' = (D_1) = Adv, \delta'_{D_2} = Adv.$

We have: $Common(\delta, \delta') = \{D_0, D_1, D_2\}$, $\delta_{D_0} = \delta'_{D_0}$, $\delta_{D_2} = \delta'_{D_2}$ and δ_{D_1} dominates δ'_{D_1} w.r.t. $lmax(lmin)$, since $((1, 0.1), (1, 0.9)) \triangleright_{lmax(lmin)} ((1, 0.1)(0.5, 0.9))$.

So, δ should be strictly preferred to δ'.

Let us compute the reduction of δ:

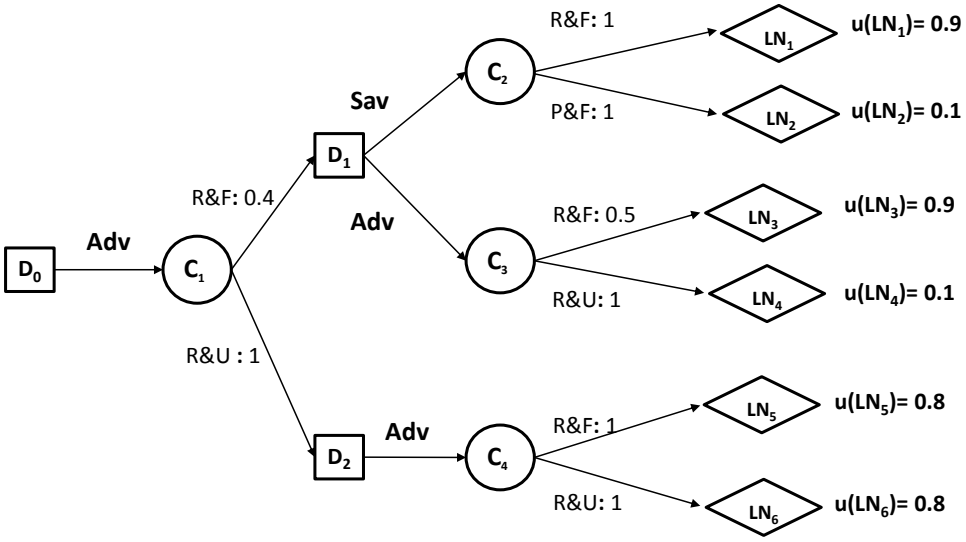


Figure 3.4: A counter-example showing the non-efficiency of $\trianglerighteq_{lmax(lmin)}$

- $\pi_\delta(0.9) = \pi_\delta(0.1) = \min(0.4, 1) = 0.4$ *and*
- $\pi_\delta(0.8) = \min(1, 1) = 1$,

and for δ' *we have:*

- $\pi_{\delta'}(0.9) = \min(0.4, 0.5) = 0.4$,
- $\pi_{\delta'}(0.1) = \min(0.4, 1) = 0.4$ *and*
- $\pi_{\delta'}(0.8) = \min(1, 1) = 1$.

Thus, δ *and* δ' *are indifferent for* $\trianglerighteq_{lmax(lmin)}$*, since both of them have the same reduction. This contradicts Pareto efficiency.*

The drowning effect here is due to the reduction of the policies, namely to the fact that the possibility of a trajectory is drowned by the one of the least possible of its transitions. That is why we propose to give up the principle of reduction and to build lexicographic comparisons on policies considered *in extenso*.

Definition 3.3. *(Leximax, Leximin relations on trajectories)*
The comparison of trajectories can be seen as a comparison of vectors associated to trajectories. For any $\pi_\tau = (\pi_1, \ldots, \pi_h, u_h)$ *and* $\pi_{\tau'} = (\pi'_1, \ldots, \pi'_h, u'_h)$*, we define* $\succeq_{lmin}$ *and* $\succeq_{lmax}$ *orders by:*

$$\tau \succeq_{lmin} \tau' \text{ iff } (\pi_1, \ldots, \pi_h, u_h) \succeq_{lmin} (\pi'_1, \ldots, \pi'_h, u'_h) \tag{3.1}$$

$$\tau \succeq_{lmax} \tau \ \textit{iff} \ (1-\pi_1,\ldots,1-\pi_h,u_h) \succeq_{lmax} (1-\pi'_1,\ldots,1-\pi'_h,u'_h) \tag{3.2}$$

Hence the proposition of the following preference relations[1]:

Definition 3.4. *(Leximax(leximin), Leximin(leximax) relations on policies)*
Let $\delta, \delta' \in \Delta$. Then:

$$\delta \succeq_{lmax(lmin)} \delta' \ \textit{iff} \ \forall i,\ \tau_{\lambda(i)} \sim_{lmin} \tau'_{\lambda(i)}$$
$$\textit{or} \ \exists i^*,\ \forall i \leq i*, \tau_{\lambda(i)} \sim_{lmin} \tau'_{\lambda(i)} \ \textit{and} \ \tau_{\lambda(i^*)} \succ_{lmin} \tau'_{\lambda(i^*)}, \tag{3.3}$$

$$\delta \succeq_{lmin(lmax)} \delta' \ \textit{iff} \ \forall i,\ \tau_{\sigma(i)} \sim_{lmax} \tau'_{\sigma(i)}$$
$$\textit{or} \ \forall i,\ \tau_{\sigma(i)} \sim_{lmax} \tau'_{\sigma(i)} \ \textit{or} \ \exists i^*,\ \forall i \leq i*,\ \tau_{\sigma(i)} \sim_{lmax} \tau'_{\sigma(i)} \ \textit{and} \ \tau_{\sigma(i^*)} \succ_{lmax} \tau'_{\sigma(i^*)}, \tag{3.4}$$

where $\tau_{\lambda(i)}$ (resp. $\tau'_{\lambda(i)}$) is the i^{th} best trajectory of δ (resp.δ') according to $\succeq_{lmin}$ and $\tau_{\sigma(i)}$ (resp. $\tau'_{\sigma(i)}$) is the i^{th} worst trajectory of δ (resp.δ') according to $\succeq_{lmax}$.

Hence, a policy can be represented by a matrix with N lines, s.t. N is the number of trajectories, and $M = h + 1$ columns (h being the horizon of the decision model). Indeed, comparing two policies w.r.t. $\succeq_{lmax(lmin)}$ (resp. $\succeq_{lmin(lmax)}$) consists in first ordering the two corresponding matrices of trajectories as follows:

- the elements of each trajectory in increasing order w.r.t $\succeq_{lmin}$ (resp. in decreasing order $\succeq_{lmax}$),
- then all the trajectories of each policy are arranged lexicographically top-down in decreasing order (resp. top-down in increasing order).

Then, it is enough to lexicographically compare the two new matrices of trajectories, denoted ρ_δ (resp. $\rho_{\delta'}$), element by element. The first pair of different elements determines the best matrix/policy. Note that the ordered matrix ρ_δ (resp. $\rho_{\delta'}$) can be seen as the utility of applying the policy δ (resp. δ').

Formally, let ρ denotes a $N \times M$ matrix of elements of V. $\forall\ i \in 1..M; \forall\ j \in 1..N$, ρ_{ij} denotes an element of ρ in line i and column j. Given two ordered matrices ρ and ρ', we say that $\rho \succ_{lmaxlmin} \rho'$ iff $\exists\ i, j$ such that $\forall\ i' < i, \forall\ j',\ \rho_{i',j'} = \rho'_{i',j'}$ and $\forall\ j' < j,\ \rho_{i,j'} = \rho'_{i,j'}$ and $\rho_{i,j} > \rho'_{i,j}$. $\rho \sim \rho'$ iff they are identical.

As a matter of fact, once the matrix of trajectories ρ_δ is reordered, the first element is always equal to $u_{opt}(\delta)$ (resp. to $u_{pes}(\delta)$ when applying $lmin(lmax)$):

[1] If the policies have different numbers of trajectories, neutral trajectories (vectors) are added at the bottom of the shortest list of trajectories.

Proposition 3.1.
Let δ be a policy and ρ be its ordered matrix w.r.t. $\succeq_{lmaxlmin}$. Then:

$$u_{opt}(\delta) = \rho_{1,1}$$

Proof of Proposition 3.1.
Let ρ be the ordered matrix of δ. Since $\succeq_{lmax(lmin)}$ refines $\succeq_{maxmin}$ [Fargier and Sabbadin, 2005], we have:
$\rho_{1,1} = \max_i \ \min_j \rho_{i,j}$
$= \max_{i=1,n} \ \min(\pi_1, \ldots, \pi_h, u_h)$
$= \max_{i=1,n} \ \min(\pi_1, \min(\pi_2, \min(\pi_{h-1}, \min(\pi_h, u_h))))$
$= u_{opt}(\delta)$.

□

Example 3.1. *Let us consider the Counter-example 3.1 with the same $\Pi\mathcal{DT}$ (we recall it in Figure 3.5) and let us optimize it using the proposed lexicographic comparisons. We consider, once again, the policies δ and δ' defined by:*

- $\delta(D_0) = Adv; \delta(D_1) = Adv; \delta(D_2) = Adv.$
- $\delta'(D_0) = Adv; \delta'(D_1) = Sav; \delta'(D_2) = Adv,$

As we have seen, δ has 4 trajectories ($\tau_1, \tau_2, \tau_5, \tau_6$):

- $\tau_1 = (Adv, R\&U, Adv, R\&U)$ *with* $\pi_{\tau_1} = (0.5, 0.5, 0.5)$,
- $\tau_2 = (Adv, R\&U, Adv, R\&F)$ *with* $\pi_{\tau_2} = (0.5, 1, 0.7)$,
- $\tau_5 = (Adv, R\&F, Adv, R\&U)$ *with* $\pi_{\tau_5} = (1, 0.5, 0.5)$,
- $\tau_6 = (Adv, R\&F, Adv, R\&F)$ *with* $\pi_{\tau_6} = (1, 1, 0.5)$.

So, the ordered matrix of trajectories is: $\rho_\delta = \begin{bmatrix} 0.5 & 1 & 1 \\ 0.5 & 0.7 & 1 \\ 0.5 & 0.5 & 1 \\ 0.5 & 0.5 & 0.5 \end{bmatrix}$.

δ' has also 4 trajectories ($\tau_3, \tau_4, \tau_5, \tau_6$):

- $\tau_3 = (Adv, R\&U, Sav, P\&U)$ *with* $\pi_{\tau_3} = (0.5, 0.2, 0.3)$,
- $\tau_4 = (Adv, R\&U, Sav, R\&U)$ *with* $\pi_{\tau_4} = (0.5, 1, 0.5)$,
- $\tau_5 = (Adv, R\&F, Adv, R\&U)$ *with* $\pi_{\tau_5} = (1, 0.5, 0.5)$,

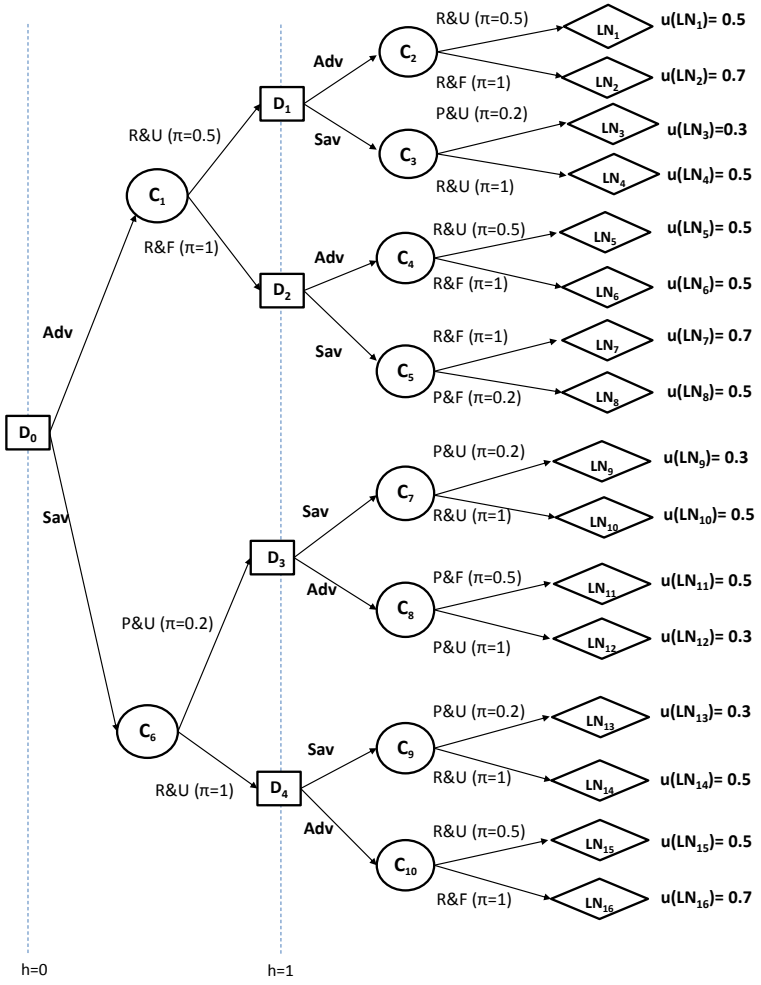


Figure 3.5: The $\Pi\mathcal{DT}$ of Example 3.1

- $\tau_6 = (Adv, R\&F, Adv, R\&F)$ *with* $\pi_{\tau_6} = (1, 1, 0.5)$,

The ordered matrix of trajectories is: $\rho_{\delta'} = \begin{bmatrix} 0.5 & 1 & 1 \\ 0.5 & 0.5 & 1 \\ 0.5 & 0.5 & 1 \\ 0.2 & 0.3 & 0.5 \end{bmatrix}$.

Given the two ordered matrices ρ_δ *and* $\rho_{\delta'}$, δ *and* δ' *are indifferent for optimistic utility since the two first elements of the matrices are equal i.e.* $u_{opt}(\delta) = u_{opt}(\delta') = 0.5$. *For* $lmax(lmin)$ *we compare the successive next elements until we find a pair of different values. In particular, we have the second element of the second best trajectory of* δ *is strictly greater than the second element of the second best trajectory of* δ' *i.e.* $0.7 > 0.5$, *while all the former elements are equal. So, the second best trajectory of* δ *is strictly preferred to the second best trajectory of* δ' *according to* $\succeq_{lmin}$. *We deduce that* δ *is strictly preferred to* δ'*:*

$$\delta \succ_{lmax(lmin)} \delta' \textit{ since } (0.5, 0.7, 1) \succ_{lmin} (0.5, 0.5, 1).$$

We show now that, the proposed lexicographic criteria are relevant refinements and escape the drowning effect:

Proposition 3.2. *($\succeq_{lmax(lmin)}$ and $\succeq_{lmin(lmax)}$ orders)*

- $\succeq_{lmax(lmin)}$ *is complete, transitive and refines* $\succeq_{u_{opt}}$*;*
- $\succeq_{lmin(lmax)}$ *is complete, transitive and refines* $\succeq_{u_{pes}}$.

Proof of Proposition 3.2.

- **Completeness**. It is a consequence of the completeness of $\succeq_{lmax}$ and $\succeq_{lmin}$.

- **Transitivity**.
  - We prove that $\succeq_{lmax(lmin)}$ is transitive. The proof relies on the transitivity of $\succeq_{lmin}$. Let us consider three policies, δ, δ' and δ'' and assume $\delta \succeq_{lmax(lmin)} \delta'$ and $\delta' \succeq_{lmax(lmin)} \delta''$. Since $\delta \succeq_{lmax(lmin)} \delta'$ and $\delta' \succeq_{lmax(lmin)} \delta''$, then we are in either following cases:
    1. $\forall i, \tau_{\lambda(i)} \sim_{lmin} \tau'_{\lambda(i)} \sim_{lmin} \tau''_{\lambda(i)}$. This happens when $\delta \sim_{lmax(lmin)} \delta' \sim_{lmax(lmin)} \delta''$. And then, by transitivity of $\succeq_{lmin}$, we have $\forall i, \tau_{\lambda(i)} \sim_{lmin} \tau''_{\lambda(i)} \Leftrightarrow \delta \sim_{lmax(lmin)} \delta''$.
    2. When either $\delta \succ_{lmax(lmin)} \delta'$ or $\delta' \succ_{lmax(lmin)} \delta''$, then, by definition of $\succeq_{lmax(lmin)}$, there exists i^*, such that:
       (a) $\forall i < i^*, \tau_{\lambda(i)} \sim_{lmin} \tau'_{\lambda(i)} \sim_{lmin} \tau''_{\lambda(i)}$,
       (b) $\tau_{\lambda(i^*)} \succeq_{lmin} \tau'_{\lambda(i^*)} \succeq_{lmin} \tau''_{\lambda(i^*)}$ and
       (c) either $\tau_{\lambda(i^*)} \succ_{lmin} \tau'_{\lambda(i^*)}$ or $\tau'_{\lambda(i^*)} \succ_{lmin} \tau''_{\lambda(i^*)}$, or both.
       Then, once again by transitivity of $\succeq_{lmin}$, $\tau_{\lambda(i^*)} \succ_{lmin} \tau''_{\lambda(i^*)}$.
       So, $\delta \succ_{lmax(lmin)} \delta''$.

    So, points 1 and 2 imply, together, that $\delta \succeq_{lmax(lmin)} \delta'$ and $\delta' \succeq_{lmax(lmin)} \delta''$ imply $\delta \succeq_{lmax(lmin)} \delta''$.
  - Similarly, it can be checked that $\succeq_{lmin(lmax)}$ is transitive. Let us consider three policies, δ, δ' and δ'' and assume $\delta \succeq_{lmin(lmax)} \delta'$ and $\delta' \succeq_{lmin(lmax)} \delta''$. Since $\delta \succeq_{lmin(lmax)} \delta'$ and $\delta' \succeq_{lmin(lmax)} \delta''$, then we are in either following cases:
    1. $\forall i, \tau_{\sigma(i)} \sim_{lmax} \tau'_{\sigma(i)} \sim_{lmax} \tau''_{\sigma(i)}$.
       This happens when $\delta \sim_{lmin(lmax)} \delta' \sim_{lmin(lmax)} \delta''$. And then, by transitivity of $\succeq_{lmax}$, we have $\forall i, \tau_{\sigma(i)} \sim_{lmax} \tau''_{\sigma(i)} \Leftrightarrow \delta \sim_{lmin(lmax)} \delta''$.

2. When either $\delta \succ_{lmin(lmax)} \delta'$ or $\delta' \succ_{lmin(lmax)} \delta''$, then, by definition of $\succeq_{lmin(lmax)}$, there exists i^*, such that:
(a) $\forall i < i^*, \tau_{\sigma(i)} \sim_{lmax} \tau'_{\sigma(i)} \sim_{lmax} \tau''_{\sigma(i)}$,
(b) $\tau_{\sigma(i^*)} \succeq_{lmax} \tau'_{\sigma(i^*)} \succeq_{lmax} \tau''_{\sigma(i^*)}$ and
(c) either $\tau_{\sigma(i^*)} \succ_{lmax} \tau'_{\sigma(i^*)}$ or $\tau'_{\sigma(i^*)} \succ_{lmax} \tau''_{\sigma(i^*)}$, or both.
Then, once again by transitivity of $\succeq_{lmax}$, $\tau_{\sigma(i^*)} \succ_{lmax} \tau''_{\sigma(i^*)}$.
So, $\delta \succ_{lmin(lmax)} \delta''$.

So, points 1 and 2 imply, together, that $\delta \succeq_{lmin(lmax)} \delta'$ and $\delta' \succeq_{lmin(lmax)} \delta''$ imply $\delta \succeq_{lmin(lmax)} \delta''$.

- **Refinement**.
  - We prove that $\succeq_{lmax(lmin)}$ refines $\succeq_{u_{opt}}$. Let us consider two policies δ and δ'. If $u_{opt}(\delta) > u_{opt}(\delta')$
    $\Leftrightarrow \max_{\tau\in\delta} \min\{\min_{\pi_k\in\pi_\tau} \pi_k, u_h\} > \max_{\tau'\in\delta'} \min\{\min_{\pi'_k\in\pi_{\tau'}} \pi'_k, u'_h\}$
    $\Rightarrow \max_{\tau\in\delta} \min(\pi_1, \ldots, \pi_h, u_h) > \max_{\tau'\in\delta'} \min(\pi'_1, \ldots, \pi'_h, u'_h)$.
    Since $\min(\pi_1, \ldots, \pi_h, u_h) > \min(\pi'_1, \ldots, \pi'_h, u'_h)$
    $\Rightarrow (\pi_1, \ldots, \pi_h, u_h) \succ_{lmin} (\pi'_1, \ldots, \pi'_h, u'_h)$ (leximin ordering refines min ordering), then $\tau_{\lambda(1)} \succ_{lmin} \tau'_{\lambda(1)} \Rightarrow \delta \succ_{lmax(lmin)} \delta'$ where $\tau_{\lambda(1)}$ (resp. $\tau'_{\lambda(1)}$) is the best trajectory of δ (resp.δ') according to $\succeq_{lmin}$.
    So by the definition of $\succeq_{lmax(lmin)}$ we have $\delta \succ_{lmax(lmin)} \delta'$. And we deduce that $\succeq_{lmax(lmin)}$ refines $\succeq_{u_{opt}}$.
  - We show in the same way that $\succeq_{lmin(lmax)}$ refines $\succeq_{u_{pes}}$. Let us consider two policies δ and δ'. If $u_{pes}(\delta) > u_{pes}(\delta')$
    $\Leftrightarrow \min_{\tau\in\delta} \max\{\min_{\pi_k\in\pi_\tau} (1-\pi_k), u_h\} > \min_{\tau'\in\delta'} \max\{\min_{\pi'_k\in\pi_{\tau'}} (1-\pi'_k), u'_h\}$
    $\Leftrightarrow \min_{\tau\in\delta} \max(1-\pi_1, \ldots, 1-\pi_h, u_h) > \min_{\tau'\in\delta'} \max(1-\pi'_1, \ldots, 1-\pi'_h, u'_h)$.
    Since $\max(1-\pi_1, \ldots, 1-\pi_h, u_h) > \max(1-\pi'_1, \ldots, 1-\pi'_h, u'_h)$
    $\Rightarrow (1-\pi_1, \ldots, 1-\pi_h, u_h) \succ_{lmax} (1-\pi'_1, \ldots, 1-\pi'_h, u'_h)$ (leximax ordering refines max ordering). Then $\tau_{\sigma(1)} \succ_{lmax} \tau'_{\sigma(1)} \Rightarrow \delta \succ_{lmin(lmax)} \delta'$ where $\tau_{\sigma(1)}$ (resp. $\tau'_{\sigma(1)}$) is the worst trajectory of δ (resp.δ') according to $\succeq_{lmax}$. So, by definition of $\succeq_{lmin(lmax)}$ we have $\delta \succ_{lmin(lmax)} \delta'$.

□

Proposition 3.3.
$\succeq_{lmax(lmin)}$ and $\succeq_{lmin(lmax)}$ both satisfy the principle of Pareto efficiency as well as strict monotonicity.

Proof of Proposition 3.3.
(i) We first prove that $\succeq_{lmax(lmin)}$ and $\succeq_{lmin(lmax)}$ are strictly monotonic.

Note that $\forall C_j \in \mathcal{N}_C, \forall D_i \in Succ(C_j), \delta, \delta' \in \Delta_{D_i}, \delta'' \in \Delta_{Succ(C_j)\setminus D_i}$. The trajectories of $\delta+\delta''$ are composed of two disjoint sets of trajectories : One for δ and one for δ''. The same holds for $\delta' + \delta''$. Then, note that adding or removing identical trajectories to two sets of trajectories does not change the $\succeq_{lmax(lmin)}$ or the $\succeq_{lmin(lmax)}$ ordering between these two sets.

To be more precise, assume, for example, that $\delta \succ_{lmax(lmin)} \delta'$.
Then, $\exists i^*, \forall i < i^*, \tau_{\lambda(i)} \sim_{lmin} \tau'_{\lambda(i)}$ and $\tau_{\lambda(i^*)} \succ_{lmin} \tau'_{\lambda(i^*)}$. The trajectories corresponding to δ'' are composed of trajectories which rank before $\tau_{\lambda(i^*)}$, and after $\tau_{\lambda(i^*)}$. Obviously, the ones that rank before $\tau_{\lambda(i^*)}$ are added to both lists of trajectories, and thus simply delay i^* while not inducing a new preference. And the ones that rank after $\tau_{\lambda(i^*)}$ are not taken into consideration in the comparison of $\delta + \delta''$ and $\delta' + \delta''$. In the same way, by definition of $\succeq_{lmin(lmax)}$ we get $\delta \succ_{lmin(lmax)} \delta'$ i.e. $\exists i^*, \forall i < i^*, \tau_{\sigma(i)} \sim_{lmax} \tau'_{\sigma(i)}$ and $\tau_{\sigma(i^*)} \succ_{lmax} \tau'_{\sigma(i^*)}$. The same result as for $\tau_{\lambda(i^*)}$ handles for $\tau_{\sigma(i^*)}$. Thus, $\succeq_{lmax(lmin)}$ and $\succeq_{lmin(lmax)}$ are strictly monotonic.
(ii) Now we prove that $\succeq_{lmax(lmin)}$ satisfy The principle of Pareto efficiency. So, suppose that $\delta \succeq_{lmax(lmin)} \delta'$. Two cases arise:

- if $\forall i,\ \tau_{\lambda(i)} \sim_{lmin} \tau'_{\lambda(i)}$ and then $\delta \sim_{lmax(lmin)} \delta'$,
- if $\exists i^*$, s.t. $\forall i < i^*,\ \tau_{\lambda(i)} \sim_{lmin} \tau'_{\lambda(i)}$ and $\tau_{\lambda(i^*)} \succ_{lmin} \tau'_{\lambda(i^*)}$. Then, $\tau_{\lambda(i^*)} \succ_{lmin} \tau'_{\lambda(i^*)}$ implies that there exist a pair of different $(\beta_{i^*,k}, \beta'_{i^*,k})$, where $\beta_{i^*,k}$ (resp. $\beta'_{i^*,k}$) is an element of $\tau_{\lambda(i^*)}$ (resp. $\tau'_{\lambda(i^*)}$), that determines the best policy. Here we get $\beta_{i^*,k} > \beta'_{i^*,k}$ i.e. $\tau_{\lambda(i^*)} \succ_{lmin} \tau'_{\lambda(i^*)}$ and thus $\delta \succ_{lmax(lmin)} \delta'$.

In summary, if we have $\delta \succeq_{lmax(lmin)} \delta'$ and $\exists i^*$, s.t. $\tau_{\lambda(i^*)} \succ_{lmin} \tau'_{\lambda(i^*)}$ we get $\delta \succ_{lmax(lmin)} \delta'$ which expresses exactly the principle of Pareto efficiency in the case $\succeq_{lmax(lmin)}$.
(iii) Let us prove $\succeq_{lmin(lmax)}$ satisfy The principle of Pareto efficiency. When considering the $\succeq_{lmin(lmax)}$ order, the same kind of result can be obtained.

So, suppose that $\delta \succeq_{lmin(lmax)} \delta'$. Two cases arise:

- if $\forall i,\ \tau_{\sigma(i)} \sim_{lmax} \tau'_{\sigma(i)}$ and then $\delta \sim_{lmin(lmax)} \delta'$,
- if $\exists i^*$, s.t. $\forall i < i^*,\ \tau_{\sigma(i)} \sim_{lmax} \tau'_{\sigma(i)}$ and $\tau_{\sigma(i^*)} \succ_{lmax} \tau'_{\sigma(i^*)}$. Then, $\tau_{\sigma(i^*)} \succ_{lmax} \tau'_{\sigma(i^*)}$ implies that there exist a pair of different $(\beta_{i^*,k}, \beta'_{i^*,k})$, where $\beta_{i^*,k}$ (resp. $\beta'_{i^*,k}$) is an element of $\tau_{\sigma(i^*)}$ (resp. $\tau'_{\sigma(i^*)}$), determining the best policy. We get $\beta_{i^*,k} > \beta'_{i^*,k}$ i.e. $\tau_{\sigma(i^*)} \succ_{lmax} \tau'_{\sigma(i^*)}$ and thus $\delta \succ_{lmin(lmax)} \delta'$.

In summary, if we have $\delta \succeq_{lmin(lmax)} \delta'$ and $\exists i^*$, s.t. $\tau_{\sigma(i^*)} \succ_{lmax} \tau'_{\sigma(i^*)}$ we get $\delta \succ_{lmin(lmax)} \delta'$ which expresses exactly the principle of Pareto efficiency in the case of $\succeq_{lmin(lmax)}$. □

3.3.2 Problems with intermediate utilities

Let h be the horizon of the stationary $\Pi\mathcal{MDP}$, as in the previous Section a trajectory is a sequence of states and actions. A policy can thus be viewed as a matrix where each line corresponds to a distinct trajectory $\tau = (s_0, a_0, s_1, \ldots, s_{h-1}, a_{h-1}, s_h)$ i.e. to a vector $v_\tau = (u_0, \pi_1, u_1, \pi_2, \ldots, \pi_{h-1}, u_h)$. This allow us to define the comparison of trajectories and policies by [2]:

$$\tau \succeq_{lmin} \tau' \text{ iff } (u_0, \pi_1, \ldots, \pi_h, u_h) \succeq_{lmin} (u'_0, \pi'_2, \ldots, \pi'_h, u'_h) \tag{3.5}$$

$$\tau \succeq_{lmax} \tau' \text{ iff } (u_0, 1-\pi_1, \ldots, 1-\pi_h, u_h) \succeq_{lmax} (u'_0, 1-\pi'_1, \ldots 1-\pi'_h, u'_h) \tag{3.6}$$

$\delta \succeq_{lmax(lmin)} \delta'$iff $\forall i,\ \tau_{\lambda(i)} \sim_{lmin} \tau'_{u(i)}$

$$or\ \exists i^*,\ \forall i < i^*, \tau_{\lambda(i)} \sim_{lmin} \tau'_{\lambda(i)}\ \ and\ \ \tau_{\lambda(i^*)} \succ_{lmin} \tau'_{\lambda(i^*)} \tag{3.7}$$

$\delta \succeq_{lmin(lmax)} \delta'$ iff $\forall i,\ \tau_{\sigma(i)} \sim_{lmax} \tau'_{\sigma(i)}$

$$or\ \exists i^*,\ \forall i < i^*, \tau_{\sigma(i)} \sim_{lmax} \tau'_{\sigma(i)}\ \ and\ \ \tau_{\sigma(i^*)} \succ_{lmax} \tau'_{\sigma(i^*)} \tag{3.8}$$

where $\tau_{\lambda(i)}$ (resp. $\tau'_{\lambda(i)}$) is the i^{th} best trajectory of δ (resp. δ') according to $\succeq_{lmin}$ and $\tau_{\sigma(i)}$ (resp. $\tau'_{\sigma(i)}$) is the i^{th} worst trajectory of δ (resp. δ') according to $\succeq_{lmax}$.

It is easy to show that using these definitions, we also get efficient refinements of u_{opt} and u_{pes}:

Proposition 3.4.

- *If* $u_{opt}(\delta) > u_{opt}(\delta')$ *then* $\delta \succ_{lmax(lmin)} \delta'$,
- *If* $u_{pes}(\delta) > u_{pes}(\delta')$ *then* $\delta \succ_{lmin(lmax)} \delta'$.

Proof of Proposition 3.4.

- We prove that $\succeq_{lmax(lmin)}$ refines $\succeq_{u_{opt}}$ in sequential decision models with intermediate utilities (stationary $\Pi\mathcal{MDP}$). Following Proof of Proposition 3.2, we consider two policies

[2] If a trajectory is shorter than h, neutral elements (1 for the optimistic case and 0 for the pessimistic one) are added at the end. If the policies have different numbers of trajectories, neutral trajectories (vectors) are added to the shortest one.

δ and δ'.
If $u_{opt}(\delta) > u_{opt}(\delta')$
$\Leftrightarrow \max_{\tau\in\delta} \min\{\pi(\tau|s_0,\delta), u(\tau)\} > \max_{\tau'\in\delta'} \min\{\pi(\tau'|s_0,\delta'), u(\tau')\}$
$\Rightarrow \max_{\tau\in\delta} \min\{\min_{i=1..h} \pi(s_i|s_{i-1}, \delta(s_{i-1})), \min_{i=1..h} u(s_i)\} >$
$\max_{\tau'\in\delta'} \min\{\min_{i=1..h} \pi'(s_i|s_{i-1}, \delta'(s_{i-1})), \min_{i=1..h} u'(s_i)\}$
$\Rightarrow \max_{\tau\in\delta} \min(u_0, \pi_1, u_1, \pi_2, \ldots, \pi_{h-1}, u_h) > \max_{\tau'\in\delta'} \min(u'_0, \pi'_1, u'_1, \pi'_2, \ldots, \pi'_{h-1}, u'_h)$.

Since $\min(u_0, \pi_1, u_1, \pi_2, \ldots, \pi_{h-1}, u_h) > \min(u'_0, \pi'_1, u'_1, \pi'_2, \ldots, \pi'_{h-1}, u'_h)$
$\Rightarrow (u_0, \pi_1, u_1, \pi_2, \ldots, \pi_{h-1}, u_h) \succ_{lmin} (u'_0, \pi'_1, u'_1, \pi'_2, \ldots, \pi'_{h-1}, u'_h)$ (as leximin ordering refines min ordering), then $\tau_{\lambda(1)} \succ_{lmin} \tau'_{\lambda(1)} \Rightarrow \delta \succ_{lmax(lmin)} \delta'$ where $\tau_{\lambda(1)}$ (resp. $\tau'_{\lambda(1)}$) is the best trajectory of δ (resp.δ') according to $\succeq_{lmin}$.

Using the definition of $\succeq_{lmax(lmin)}$ (Equation 3.3) we have $\delta \succ_{lmax(lmin)} \delta'$. Thus, $\succeq_{lmax(lmin)}$ refines $\succeq_{u_{opt}}$.

- We prove in the same way that $\succeq_{lmin(lmax)}$ refines $\succeq_{u_{pes}}$. Considering two policies δ and δ' s.t. $u_{pes}(\delta) > u_{pes}(\delta')$

$\Leftrightarrow \min_{\tau\in\delta} \max\{1 - \pi(\tau|s_0,\delta), u(\tau)\} > \min_{\tau'\in\delta'} \max\{1 - \pi(\tau'|s_0,\delta'), u(\tau')\}$
$\Leftrightarrow \min_{\tau\in\delta} \max\{\min_{i=1..h} (1 - \pi(s_i|s_{i-1}, \delta(s_{i-1}))), \min_{i=1..h} u(s_i)\} >$
$\min_{\tau'\in\delta'} \max\{\min_{i=1..h} (1 - \pi'(s_i|s_{i-1}, \delta'(s_{i-1}))), \min_{i=1..h} u'(s_i)\}$
$\Leftrightarrow \min_{\tau\in\delta} \max\{\min(u_0, 1-\pi_1, u_1, 1-\pi_2, \ldots, 1-\pi_{h-1}, u_h)\} >$
$\min_{\tau'\in\delta'} \max\{(u'_0, 1-\pi'_1, u'_1, 1-\pi'_2, \ldots, 1-\pi'_{h-1}, u'_h)\}$.

Since $\max(u_0, 1-\pi_1, u_1, 1-\pi_2, \ldots, 1-\pi_{h-1}, u_h) >$
$\max(u'_0, 1-\pi'_1, u'_1, 1-\pi'_2, \ldots, 1-\pi'_{h-1}, u'_h)$
$\Rightarrow (u_0, 1-\pi_1, u_1, 1-\pi_2, \ldots, 1-\pi_{h-1}, u_h) \succ_{lmax}$
$(u'_0, 1-\pi'_1, u'_1, 1-\pi'_2, \ldots, 1-\pi'_{h-1}, u'_h)$ (as leximax ordering refines max ordering).

Then $\tau_{\sigma(1)} \succ_{lmax} \tau'_{\sigma(1)1} \Rightarrow \delta \succ_{lmin(lmax)} \delta'$ where $\tau_{\sigma(1)}$ (resp. $\tau'_{\sigma(1)}$) is the worst trajectory of δ (resp.δ') according to $\succeq_{lmax}$. So, by definition of $\succeq_{lmin(lmax)}$ (Equation 3.4) we have $\delta \succ_{lmin(lmax)} \delta'$. We deduce that $\succeq_{lmin(lmax)}$ refines $\succeq_{u_{pes}}$.

□

Proposition 3.5. *Relations $\succeq_{lmin(lmax)}$ and $\succeq_{lmax(lmin)}$ are complete, transitive and satisfy the principle of strict monotonicity.*

Proof of Proposition 3.5.

- **Completeness**. It is a consequence of the completeness of $\succeq_{lmax}$ and $\succeq_{lmin}$.

- **Transitivity**. The proof can be deduced from Proof of Proposition 3.2 (second point) by replacing $\pi_\tau = (\pi_1, \ldots, \pi_h, u_h)$ by $v_\tau = (u_0, \pi_1, u_1, \pi_2, \ldots, \pi_{h-1}, u_h)$. Since π_τ and v_τ are obviously two vectors of numbers in $[0, 1]$, then by transitivity of $\succeq_{lmax}$ and $\succeq_{lmin}$ we can conclude that $\succeq_{lmin(lmax)}$ and $\succeq_{lmax(lmin)}$ are transitive.

- **Monotonicity**. Using Proof of Proposition 3.2 (third point): when considering intermediate utilities v_τ are vectors of numbers that will be ordered just like π_τ. So, $\delta + \delta''$ contains two disjoint sets of trajectories (i.e. vectors): the ones of δ and the ones of δ'' (and similarly for $\delta' + \delta''$). Then, adding or removing identical trajectories (i.e. vectors) to two sets of trajectories does not change their comparison by $\succeq_{lmax(lmin)}$ (resp. $\succeq_{lmin(lmax)}$) - while it may transform a strict preference into an indifference if u_{opt} (resp. u_{pes}) were used.

□

3.4 Summary

In this Chapter, we have discussed the limitations of possibilistic qualitative utilities, i.e. the drowning effect in sequential decision-making. It appears that these criteria do not satisfy the principle of Pareto efficiency nor the strict monotonicity. However, we have shown that it is possible to define an extended version of lexicographic comparisons, (initially proposed for the one-step decision problem), to improve discrimination in sequential decision-making. Properties of the proposed lexicographic comparisons have important consequences; from a prescriptive point of view, they highlight the rationality of $lmax(lmin)$ and $lmin(lmax)$ and suggest a probabilistic interpretation presented Chapter 5. These properties allow us to define Dynamic Programming algorithms for calculating lexicographic optimal policies- this is the topic of the next Chapter.

CHAPTRE 4

Optimizing Lexicographic criteria in $\Pi\mathcal{DT}$s and finite-horizon $\Pi\mathcal{MDP}$s

Contents

4.1 Introduction

In the previous Chapter, we have proposed lexicographic criteria that satisfy the crucial properties of strict monotonicity and transitivity which allow us to define solving algorithms based on Dynamic Programming to get lexicographic optimal policies. In the present Chapter, we propose backward induction algorithms to compute lexicographic optimal strategies policies in $\Pi\mathcal{DT}$s and finite-horizon $\Pi\mathcal{MDP}$s.

This Chapter is organized as follows: the next Section is devoted to the adaptation of Dynamic Programming algorithm, namely backward induction, to the lexicographic criteria in $\Pi\mathcal{DT}$s. Section 4.3 details the lexicographic backward induction algorithm to optimize policies in possibilistic finite-horizon $\Pi\mathcal{MDP}$s. Finally, Section 4.4 presents an experimental study of these algorithms.

The main results of this chapter are published in [Ben Amor et al., 2016a, Ben Amor et al., 2016b].

4.2 Optimizing lexicographic criteria in $\Pi\mathcal{DT}$s

This Section rises the question of the policy optimization of lexicographic criteria (Definition 3.4) in $\Pi\mathcal{DT}$s .

4.2.1 Lexicographic backward induction algorithm in $\Pi\mathcal{DT}$s

Our aim here is to adapt the backward induction for optimizing possibilistic utilities in $\Pi\mathcal{DT}$s (see section 2.2.2) to lexicographic criteria.

The proposed lexicographic backward induction algorithm, so-called $(LexBI\text{-}DT)$, (Algorithm 4.1 for the $lmax(lmin)$ variant; the $lmin(lmax)$ variant is similar) proceeds in the classical way, by backward induction:

- When a chance node is reached, an optimal sub-policy is recursively built for each of its children; these subpolicies are combined but the resulting policy is NOT reduced, contrarily to what is classically done.
- When a decision node is reached, the program is called for each child and the best of them is selected.

The difference between the lexicographic backward induction algorithm and the classical version of Sabaddin (Algorithm 2.1) is that for the first one the lexicographic comparison of policies is done on the basis of their trajectories, rather than optimistic or pessimistic utilities. To this extent, the algorithm needs for each possible policy the matrix ρ that contains the following vectors:

- $\pi_\tau = (\pi_1, \ldots, \pi_h, u_h)$, in the optimistic case, or
- $\pi_\tau = (1-\pi_1, \ldots, 1-\pi_h, u_h)$, in the pessimistic case.

Hence, for line z corresponding to the z-th trajectory and stage t s.t. $t = 0..h$ we define an element of the matrix denoted by $\rho_{z,t}$ as:

$$\rho_{z,t} = \begin{cases} \pi_t & \text{if } t \leq h \text{ and } O = lmax\Big(lmin\Big) \\ 1-\pi_t & \text{if } t \leq h \text{ and } O = lmin\Big(lmax\Big) \\ u_h & \text{if } t = h+1. \end{cases} \tag{4.1}$$

Indeed, this algorithm is written in a recursive manner, and proceeds as follows:

- When node N reached is a chance node, an optimal sub-policy is recursively built for each of its children N_i: successors of a chance node may be decision nodes D_i or leaf nodes LN_i. These recursive calls return for each N_i a matrix ρ^{N_i} that contains the π_τ vectors of the trajectories τ of this sub-policy. To make the lexicographic comparison of trajectories, and thus of policies, we only need to compare their π_τ vectors - hence we memorize the matrices of numbers rather than explicit trajectories. The matrix corresponding to the trajectories beginning at N, ρ is obtained by combining the ρ^{N_i} according to π_N, the possibility distribution associated to N; (this matrix is not reduced).
- When node N reached is a decision node, an optimal sub-policy is computed for every child C_j. The best of them is selected, $\delta(N)$ receives the action corresponding to this chance node and the corresponding ρ matrix is returned.

Algorithm 4.1: $LexBI\text{-}DT$: Backward-Induction-$\Pi\mathcal{DT}$-lmax(lmin)(N:Node)

Data: A $\Pi\mathcal{DT}$; the policy, δ, is memorized as global variable
Result: Set δ for the tree rooted in N and returns the matrix ρ of the π_τ vectors corresponding its trajectories

begin
  // Leaves
  if $N \in \mathcal{N}_\mathcal{U}$ **then** $\rho = [u(N)]$;
  // Chance nodes
  if $N \in \mathcal{N}_C$ **then**
    $k = |Succ(N)|$;
    foreach $N_i \in Succ(N)$ **do**
      $\rho^{N_i} \leftarrow Backward\text{-}Induction\text{-}\Pi\mathcal{DT}\text{-}lmax(lmin)(N_i)$
    $\rho \leftarrow ConcatAndOrder(\pi_N, \rho^{N_1}, ..., \rho^{n_k})$;
  // Decision nodes
  if $N \in \mathcal{N}_D$ **then**
    $\rho \leftarrow [0]$;
    foreach $C_j \in Succ(N)$ **do**
      $\rho^{C_j} \leftarrow Backward\text{-}Induction\text{-}\Pi\mathcal{DT}\text{-}lmax(lmin)(C_j)$;
      if $\rho^{C_j} \succeq_{lmax(lmin)} \rho$ **then**
        $\rho \leftarrow \rho^{C_j}$;
        $\delta(N) \leftarrow label(N, C_j)$;
  return ρ;

When N is a chance node, its matrix depends on $N_1, \ldots, N_k$ (on the ρ^{N_i}, recursively computed) the successors of N, and on the possibility distribution on them. It is built by a call to the function $ConcatAndOrder(\pi_N, \rho^{N_1}, \ldots, \rho^{N_k})$ (outlined by Algorithm 4.2). This function adds a column to each ρ^{N_i}, filled with $\pi_N(N_i)$ the possibility degrees of getting N_i; the matrices are vertically concatenated. In order to get faster lexicographic comparisons, the elements in the lines are then ordered in decreasing (resp. increasing) order, and the lines are reordered by decreasing (resp. increasing) order w.r.t. to $lmax$ (resp. $lmin$). Once ρ has been reordered, $\rho_{1,1}$, the first element of ρ is equal to the optimistic utility (resp. the pessimistic utility) of the sub-policy represented by ρ.

Algorithm 4.2: ConcatAndOrder(π, $\rho^1, \ldots, \rho^k$)

Data: k matrices $\rho^1, \ldots, \rho^k$ and a distribution π on $\{1, \ldots, k\}$
Result: ρ, the combination of $\rho^1, \ldots, \rho^k$ according to π

```
1  // Notations:
2  // L_ρ:  number of lines of ρ,
3  // C_ρ:  number of columns of ρ,
4  // ρ_(z):  the line z in ρ,
5  // ρ_{z,t}:  the element in line z and column t in ρ
6  begin
7      NbLines ← Σ_{m=1}^{k} L_{ρ^m};
8      max_C ← max_{m=1,k}(C_{ρ^m});
9      Creates a matrix ρ with NbLines lines and max_C + 1 collumns
10     // Concatenation
11     z ← 0;
12     for m = 1, k do
13         for z' = 1, L_{ρ^m} do
14             z ← z + 1;
15             for t = 1, C_{ρ^m} do ρ_{z,t} ← ρ^m_{z',t};
16             for t = C_{ρ^m} + 1, max_C do ρ_{z,t} ← 0;
17             ρ_{z,max_C+1} ← π(m);
18     // Ordering the elements of each line by increasing
          order
19     for z = 1, NbLines do
20         sortIncreasing(ρ_(z), ≥);
21     // Ordering the lines by decreasing order according
          to lmax
22     sortDecreasing(ρ, ≥_lmax);
23     return ρ;
```

Because the ρ matrices are ordered, the lexicographic comparison of two decisions (line 15 of

Algorithm 4.1) is performed by scanning the elements of their ρ matrices, line by line from the first one. The first pair of different values determines the best matrix/chance node. If the matrices have different numbers of lines, dummy lines are added at the bottom of the shortest one (filled with 1 for the optimistic case, with 0 for the pessimistic one).

Example 4.1. *Let us consider the counter-example 3.5 with the same $\Pi\mathcal{DT}$ (we recall it in Figure 4.1). Main steps for the evaluation of this $\Pi\mathcal{DT}$ using lexicographic backward induction (Algorithm 4.1) w.r.t. $lmax(lmin)$ criterion are as follows:*

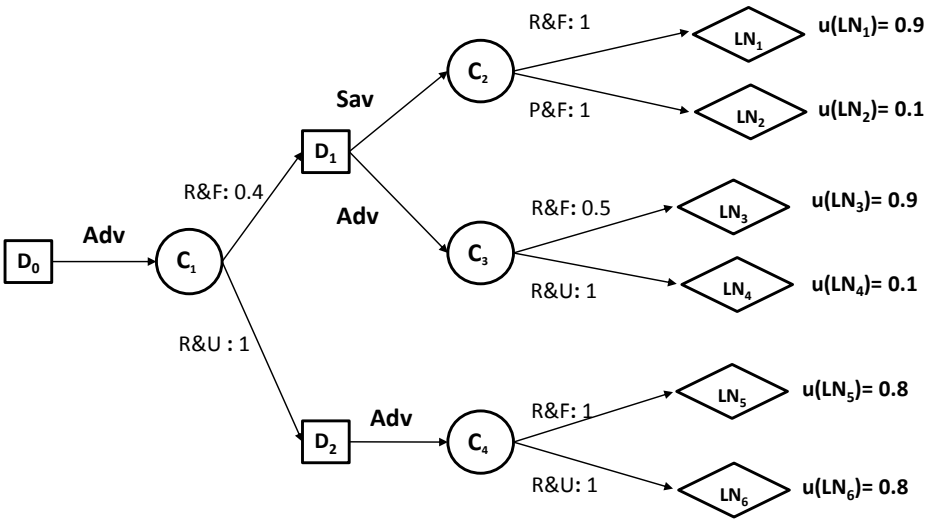


Figure 4.1: The $\Pi\mathcal{DT}$ of Example 4.1

- *Initially, we have $N = D_0$ with $Succ(D_0) = \{C_1\}$.*
- *For C_1, $\rho^{C_1} = Backward\text{-}Induction\text{-}\Pi\mathcal{DT}\text{-}lmax(lmin)(C_1)$ and $Succ(C_1) = \{D_1, D_2\}$*
- *For $N_1 = D_1$, we have $\rho^{D_1} = Backward\text{-}Induction\text{-}\Pi\mathcal{DT}\text{-}lmax(lmin)(D_1)$ and $Succ(D_1) = \{C_2, C_3\}$:*
  - *For C_2, we have $\rho^{C_2} = ConcatAndOrder(\pi_{C_2}, \rho^{LN_1}, \rho^{LN_2}) = ((1,1), \left[0.9\right], \left[0.1\right])$*
    $\rho^{C_2} = \begin{bmatrix} 0.9 & 1 \\ 0.1 & 1 \end{bmatrix}$
  - *For C_3, we have $\rho^{C_3} = ConcatAndOrder(\pi_{C_3}, \rho^{LN_3}, \rho^{LN_4}) = ((0.5,1), \left[0.9\right], \left[0.1\right])$*
    $\rho^{C_3} = \begin{bmatrix} 0.5 & 0.9 \\ 0.1 & 1 \end{bmatrix}$.

$\Rightarrow$ *Since $\rho^{C_2} \succ_{lmax(lmin)} \rho^{C_3}$, So $\rho^{D_1} = \begin{bmatrix} 0.9 & 1 \\ 0.1 & 1 \end{bmatrix}$ and $\delta(D_1) = \{D_1, Sav, C_2\}$.*

- *For $N_2 = D_2$, we have $\rho^{D_2} = Backward\text{-}Induction\text{-}\Pi\mathcal{DT}\text{-}lmax(lmin)(D_2)$ and $Succ(D_2) = \{C_4\}$:*
  - *For C_4, we have $\rho^{C_4} = ConcatAndOrder(\pi_{C_4}, \rho^{LN_5}, \rho^{LN_6}) = ((1,1), \left[0.8\right], \left[0.8\right])$*
    $\rho^{C_4} = \begin{bmatrix} 0.8 & 1 \\ 0.8 & 1 \end{bmatrix}$

  $\Rightarrow$ Since $\rho^{C_4} \succ_{lmax(lmin)} \left[0\right]$, So $\rho^{D_2} = \begin{bmatrix} 0.8 & 1 \\ 0.8 & 1 \end{bmatrix}$ and $\delta(D_2) = \{D_2, Adv, C_4\}$.

- *return to C_1, we have $\rho^{C_1} = ConcatAndOrder(\pi_{C_1}, \rho^{D_1}, \rho^{D_2})$*
  $= ((0.4,1), \begin{bmatrix} 0.9 & 1 \\ 0.1 & 1 \end{bmatrix}, \begin{bmatrix} 0.8 & 1 \\ 0.8 & 1 \end{bmatrix}) = (\begin{bmatrix} 0.4 & 0.9 & 1 \\ 0.4 & 0.1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0.8 & 1 \\ 1 & 0.8 & 1 \end{bmatrix}) = \begin{bmatrix} 0.8 & 1 & 1 \\ 0.8 & 1 & 1 \\ 0.4 & 0.9 & 1 \\ 0.1 & 0.4 & 1 \end{bmatrix}$

  Since $\rho^{C_1} \succ_{lmax(lmin)} \left[0\right]$, So $\rho^{D_0} = \begin{bmatrix} 0.8 & 1 & 1 \\ 0.8 & 1 & 1 \\ 0.4 & 0.9 & 1 \\ 0.1 & 0.4 & 1 \end{bmatrix}$ and $\delta(D_0) = \{D_0, Adv, C_1\}$.

$\Rightarrow$ The optimal strategy returned is: $\delta = (\{D_0, Adv, C_1\}, \{D_1, Sav, C_2\}, \{D_2, Adv, C_4\})$.

4.2.2 Complexity analysis

Even if working with matrices rather than numerical values, Algorithm 4.1 is polynomial w.r.t. the size of the original tree. However, it is computationally more expensive than the algorithm proposed by [Sabbadin et al., 1998] for the optimization of u_{opt}/u_{pes} (Algorithm 2.1), since it requires to memorize the trajectories that follow from the current policy (i.e. the ρ matrices). Consider the branching factor of the tree b; the size of the tree is thus equal to b^{2h}. Now, consider the size of a matrix ρ: it is in the order of $b^h \times (h+1)$ in the worst case (i.e. at the end of the backward induction) - the same order of magnitude as the one of the size of the tree.

Let us now study the time complexity. The complexity of ConcatAndOrder is based on operations: the concatenation of b matrices which is linear and ordering matrices which depends on the sorting algorithm: for instance, if we use QuickSort on an $n \times m$ matrix, then ordering the elements within a line is performed in $O(m \cdot log(m))$, and the inter-ranking of the lines is done in $O(n \cdot log(n) \cdot m)$ operations. Hence, the overall complexity of ordering matrices is in $O(n \cdot m \cdot \log(n \cdot m))$.

At each step t, from $t = h-1$ to $t = 1$, there is a chance phase and a decision phase - b decision nodes, each followed by b chance node. The chance phase is more expensive than the decision one - it makes the same number of recursive calls than the decision phase, but the decision phase does

not increase the size of the matrices it receives while the chance phase builds and orders, for each of its b^2 chance nodes a matrix that is bigger than the one it receives: it receives (for a given chance node, again) b matrices with b^{h-t-1} lines and $h-t-1$ columns and concat them in a matrix with b^{h-t} lines and $h-t$ columns (concat). The ordering then costs $b^{h-t} \cdot (h-t) \cdot \log(b^{h-t} \cdot (h-t)))$. The comparison is cheaper, since it costs $b^{h-t} \cdot (h-t)$ in the worst case.

So the worst time complexity at step t is a function $T_t = b \cdot b^{h-t} \cdot (h-t) \cdot log(b^{h-t} \cdot (h-t))$. The order of magnitude of worst case complexity of the algorithm is thus:
$M = \Sigma_{t=h-1}^{1} T_t = \Sigma_{t=h-1}^{1} b \cdot b^{h-t} \cdot (h-t) \cdot log(b^{h-t} \cdot (h-t))$. Letting $y = b^{h-t} \cdot (h-t)$, we get: $y = \begin{cases} b^{h-1} \cdot (h-1) & \text{at } t = 1 \\ b & \text{at } t = h-1 \end{cases}$

Thus $M = \Sigma_{y=b}^{b^{h-1} \cdot (h-1)} b \cdot y \cdot log(y)$ then we can calculate this upper bound approximation as follows: $M = \Sigma_{y=b}^{b^{h-1} \cdot (h-1)} b \cdot y \cdot log(y) \leq \int_{y=b}^{b^{h-1} \cdot (h-1)} b \cdot y \cdot log(y)$,

since the primitive of the function $f = y \cdot log(y)$ is $F = 0.5 \times y^2(log(y) - 0.5)$, we get:
$\Rightarrow M \leq [b \cdot y^2 \cdot log(y)]_{y=b}^{b^{h-1} \cdot (h-1)}$

$\Rightarrow M \leq b \cdot (b^{h-1} \cdot (h-1))^2 \cdot log(b^{h-1} \cdot (h-1)) - b \cdot b^2 \cdot log(b)$,

$\Rightarrow M \leq b \cdot (b^{h-1} \cdot (h-1))^2 \cdot log(b^{h-1} \cdot (h-1))$

Hence, the time complexity is in $O(b^{2h} \cdot (h-1)^2 log((h-1) \cdot b^{h-1}))$ - it is polynomial with respect to the horizon and the size of the tree (which, again, is in b^{2h}).

4.3 Optimizing lexicographic criteria in finite-horizon $\Pi\mathcal{MDP}$s

A first way to solve a finite-horizon $\Pi\mathcal{MDP}$ would be to compute a $\mathcal{DT}$ that is equivalent to the finite-horizon $\mathcal{MDP}$ (this is always possible, through the duplication of the nodes with several predecessors) and to apply the algorithm presented in the previous Section. However, this approach may lead to algorithms which are exponential in time and space (w.r.t. the finite-horizon $\mathcal{MDP}$ description), since the size of the $\mathcal{DT}$s associated to a finite-horizon $\mathcal{MDP}$ may be exponential in the size of the $\mathcal{MDP}$. So there can be a combinatorial explosion. In this Section, we propose an algorithm that calculates a lexicographic optimal policy on the finite-horizon $\Pi\mathcal{MDP}$ itself.

4.3.1 Lexicographic backward induction algorithm in finite-horizon $\Pi\mathcal{MDP}$s

The lexicographic variant of the backward induction algorithm, so-called $(LexBI\text{-}MDP)$, (see Algorithm 4.3) performs the optimization of $lmax(lmin)$ (the $lmin(lmax)$ variant is similar) in finite-horizon $\Pi\mathcal{MDP}$s.

As in the case of $\Pi\mathcal{DT}$s, the comparison of decisions (here, of actions a) is done on the basis of the trajectories they induce, given the decisions made for the future state. To this extent, one memorizes, for each state s for which a decision has been made, the matrix $\rho(s)$ corresponding to the trajectories obtained when the current policy is applied from s. For $s \in S_t, \rho(s)$ is defined as follows: for $lmax(lmin)$ each line gathers the possibility degrees $\pi(s'|a, s)$ of reaching the following state $s' \in S_{t+1}$, given that $\delta(s) = a$ is executed (resp. $1 - \pi(s'|s, a)$ for $lmin(lmax)$), combined with a trajectory in the matrix of the next state s'. Of course the matrices corresponding to the final states simply contain their utilities.

Algorithm 4.3: $LexBI\text{-}MDP$: Backward-induction-$\Pi\mathcal{MDP}$-lmax(lmin)

Data: A possibilistic finite horizon $\mathcal{MDP}$
Result: Computes and returns δ for $\mathcal{MDP}$

```
begin
    // Initialization
    ∀s ∈ S_h, ρ(s) ← [u(s)];
    t ← h;
    // Backward induction
    while t ≥ 1 do
        Future ← (ρ(s'), s' ∈ S_t);
        t ← t − 1;
        foreach s ∈ S_t do
            ρ(s) ← [0];
            foreach a ∈ A_s do
                M ← ConcatAndOrder(π(.|a, s), Future);
                if M ≥_{lmax(lmin)} ρ(s) then
                    ρ(s) ← M;
                    δ(s) ← a;
    return δ;
```

The principle of the backward induction algorithm can be summarized as follows:

- suppose being at period t (e.g. $t = h - 1$). Since the algorithm proceeds backwards, a decision $\delta(s')$ has been made for all future states (the s' in $S_{t'}, t' > t$). We have to decide

the best action for the states of S_t.

- for each state $s \in S_t$ and each action $a \in A_s$ we build the matrix M corresponding to trajectories that would be obtained if a was chosen for s, using the *ConcatAndOrder* procedure described in the previous Section - M is built from $\pi(.|s,a)$ and from the matrices already computed for the s'.
- the matrix M is then compared with the best matrix $\rho(s)$ found so far : if better, a become the current best decision for s ($\delta(s) \leftarrow a$) and M becomes the new $\rho(s)$. The lexicographic comparison of matrices is the one described in Section 3.3.1 and is made easier by the fact that the matrices have been ordered on the fly.
- the process is continued for each $S_{t'}, t' = t-1, \ldots, 0$ (by moving backward in time) until we reach the present time period (t=0) and get an optimal policy.

Example 4.2. *Main steps for the evaluation of the finite-horizon $\Pi\mathcal{MDP}$ (see Figure 4.2 to recall it) of Counter-exmple 3.3 using backward induction (Algorithm 4.3) w.r.t. $lmax(lmin)$ criterion are as follows:*

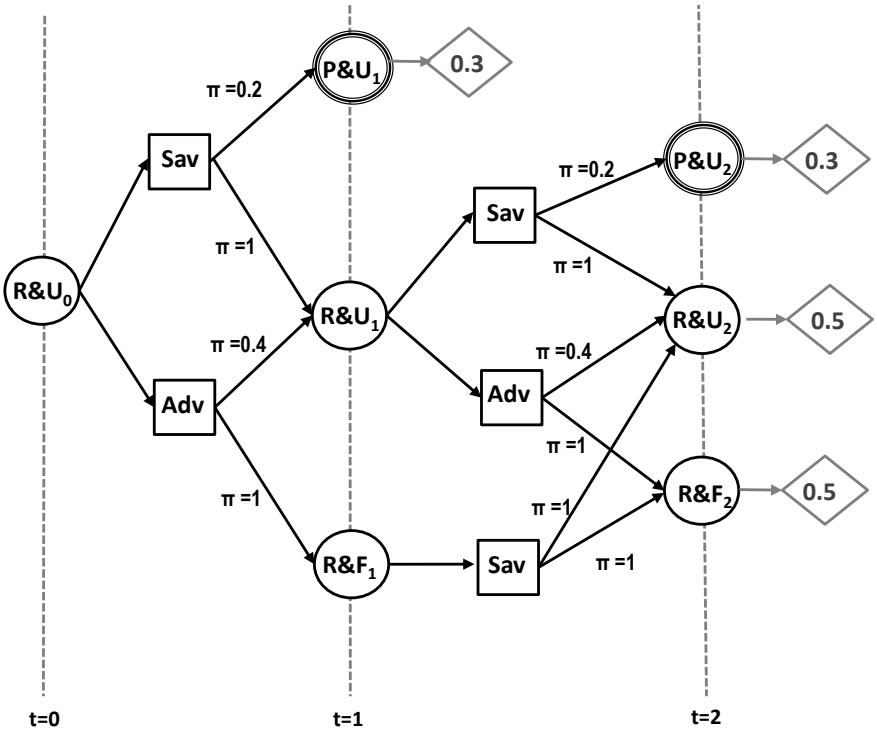


Figure 4.2: The finite-horizon $\pi\mathcal{MDP}$ of Example 4.2

• ***t=2*** *(Initialization):*
$\rho(P\&U_2) = [0.3]$, $\rho(R\&U_2) = [0.5]$, $\rho(R\&F_2) = [0.5]$.

• ***t=1***: $S_1 = \{R\&U_1, R\&F_1\}$
$*$ $s = R\&U_1$: $A_{R\&U_1,1} = \{Sav, Adv\}$ *and* $\rho(R\&U_1) \leftarrow [0]$,

- *for* $a = Sav$:
  $M \leftarrow ConcatAndOrder((0.2, 1), \rho(P\&U_2), \rho(R\&U_2))$
  $= ConcatAndOrder((0.2, 1), [0.3], [0.5]) = \begin{bmatrix} 0.5 & 1 \\ 0.2 & 0.3 \end{bmatrix}$.
  Since $M >_{lmax(lmin)} \rho(R\&U_1)$, $\rho(R\&U_1) \leftarrow M$ *and* $\delta_1(R\&U_1) \leftarrow Sav$.
- *for* $a = Adv$:
  $M \leftarrow ConcatAndOrder((0.4, 1), \rho(R\&U_2), \rho(R\&F_2))$
  $= ConcatAndOrder((0.4, 1), [0.5], [0.5]) = \begin{bmatrix} 0.5 & 1 \\ 0.4 & 0.5 \end{bmatrix}$.
  Thus $M >_{lmax(lmin)} \rho(R\&U_1)$, $\rho(R\&U_1) \leftarrow M$ *and* $\delta_1(R\&U_1) \leftarrow Adv$.

$*$ $s = R\&F_1$, $A_{S_{R\&F_1},1} = \{Sav\}$ *and* $\rho(R\&F_1) \leftarrow [0]$,

- *for* $a = Sav$:
  $M \leftarrow ConcatAndOrder((1, 1), \rho(R\&U_2), \rho(R\&F_2))$
  $= ConcatAndOrder((1, 1), [0.5], [0.5]) = \begin{bmatrix} 0.5 & 1 \\ 0.5 & 1 \end{bmatrix}$.
  Since $M >_{lmax(lmin)} \rho(R\&F_1)$, $\rho(R\&F_1) \leftarrow M$ *and* $\delta_1(R\&F_1) \leftarrow Sav$.

• ***t=0***: $S_0 = \{R\&U_0\}$
$*$ $s = R\&U_0$, $A_{R\&U_0,0} = \{Sav, Adv\}$ *and* $\rho(R\&U_0) \leftarrow [0]$,

- *for* $a = Sav$:
  $M \leftarrow ConcatAndOrder((1, 0.2), \rho(R\&U_1), \rho(P\&U_1))$
  $= ConcatAndOrder((1, 0.2), \begin{bmatrix} 0.5 & 1 \\ 0.4 & 0.5 \end{bmatrix}, [0.3]) = \begin{bmatrix} 0.5 & 1 & 1 \\ 0.4 & 0.5 & 1 \\ 0 & 0.2 & 0.3 \\ 0 & 0 & 0.2 \end{bmatrix}$.
  Since $M >_{lmax(lmin)} \rho(R\&U_0)$, $\rho(R\&U_0) \leftarrow M$ *and* $\delta_0(R\&U_0) \leftarrow Sav$.
- *for* $a = Adv$:
  $M \leftarrow ConcatAndOrder((0.4, 1), \rho(R\&U_1), \rho(R\&F_1))$
  $= ConcatAndOrder((0.4, 1), \begin{bmatrix} 0.5 & 1 \\ 0.4 & 0.5 \end{bmatrix}, \begin{bmatrix} 0.5 & 1 \\ 0.5 & 1 \end{bmatrix}) = \begin{bmatrix} 0.5 & 1 & 1 \\ 0.5 & 1 & 1 \\ 0.4 & 0.5 & 1 \\ 0.4 & 0.4 & 0.5 \end{bmatrix}$.
  Since $M >_{lmax(lmin)} \rho(R\&U_0)$, $\rho(R\&U_0) \leftarrow M$ *and* $\delta_0(R\&U_0) \leftarrow Adv$.

$\Longrightarrow$ *The optimal policy computed by lexicographic backward induction w.r.t.* $lmax(lmin)$ *is:* $\delta(R\&U_0)Adv; \delta(R\&U_1) = Adv; \delta(R\&F_1) = Sav$.

4.3.2 Complexity analysis

The backwards induction algorithm only makes a polynomial number (in the size of the $\mathcal{MDP}$ definition) of calls to the *ConcatAndOrder* function: there are as many calls to this function as the number of actions in the MDP, which is $\sum_{t=1}^{h-1}\sum_{s\in S_t}|A_s|$. At each step t, for each state s in S_t: for each action in $|A_s|$, b matrices of size $b^{h-t-1}\cdot(h-t-1)$ are received, Concatenated as a $b^{h-t}\cdot(h-t)$ matrix, ordered - which costs $b^{h-t}\cdot(h-t)log(b^{h-t}.(h-t))$ and compared with the best matrix found so far - which costs $b^{h-t}\cdot(h-t)$. Hence a time complexity in $U = \sum_{t=1}^{h-1}\sum_{s\in S_t}|A_s|\cdot b^{h-t}\cdot(h-t)log(b^{h-t}\cdot(h-t))$. Denote n the (maximal) number of states in S_t and a the (maximal) number of actions in $|A_s|$, we get $U = n\cdot a\sum_{t=1}^{h-1} b^{h-t}\cdot(h-t)\cdot log(b^{h-t}\cdot(h-t))$ i.e. a time complexity in $O(n\cdot a\cdot(h-1)^2\cdot b^{2h-2}\cdot log((h-1)\cdot b^{h-1}))$ [1].

If we consider a constant branching factor, each A_s contains b actions which in turn can lead to b non-impossible states. Then we get a time complexity in $O(b^{2h}\cdot(h-1)^2\cdot log((h-1)\cdot b^{h-1})$: unsurprisingly, it is similar to the one got for $\Pi\mathcal{DT}$s. But recall that the $\mathcal{DT}$ corresponding to a $\mathcal{MDP}$ can be exponential in the size of the $\mathcal{MDP}$.

4.3.3 Bounded backward induction for lexicographic criteria

In fact, lexicographic comparisons, which take into account the whole matrix of subsequent trajectories, overcome the drowning effect but are very costly - exponential in the size of the finite-horizon $\Pi\mathcal{MDP}$. On the other hand selecting decisions on the basis of their sole optimistic/pessimistic utility is cheap but not discriminant enough. Hence the idea to restrict the reasoning to a sub-matrix - namely to the first l lines of the matrices of trajectories $\rho(s)$, i.e. π_τ vectors of the l most important trajectories (the l best for the optimistic case, the l worst for the pessimistic ones). Bounding the number of columns is not necessary, since the combinatorial explosion in Algorithm 4.3 is due to the number of lines in the matrices (because for the finite horizon, the number of columns is bounded by $h+1$). Hence we propose the following preference:

$$\delta(s) \geq_{lmaxlmin,l} \delta'(s) \text{ iff } [\rho(s)]_l \geq [\rho'(s)]_l \tag{4.2}$$

$\geq_{lmaxlmin,+\infty}$ corresponds to $\geq_{lmaxlmin}$.

Proposition 4.1.
Let M *be a matrix of trajectories, and* l *and* l' *be any lines of* M *such that* $l' > l$*, then* $\delta \succ_{lmaxlmin,l} \delta' \Rightarrow \delta \succ_{lmaxlmin,l'} \delta'$.

[1] The details of the calculation are similar to the ones made for $\Pi\mathcal{DT}$s.

Proof of Proposition 4.1.
Note that, for any $t \in T$, $s \in S_t$, we have:

$$[\rho(s)]_l = \left[\left(\begin{array}{c|c} \multicolumn{2}{c}{\cdots} \\ \hline \pi(s'_i|s,a) & \\ \cdots & \rho(s'_i) \\ \pi(s'_i|s,a) & \\ \hline \multicolumn{2}{c}{\cdots} \end{array} \right) \right]_l$$

Let A be an ordered $N \times M$ matrix (w.r.t. $\succeq_{lmaxlmin}$) s.t. $A_{(i)}$ denotes the line i.

Now, note that, if A and B are two ordered matrices:
$[A]_l >_{lmaxlmin} [B]_l$ if and only if $\exists i^* \leq l$, such that $\forall i < i^*, A_{(i)} =_{lmin} B_{(i)}$
and $A_{(i^*)} >_{lmin} B_{(i^*)}$.

Clearly, in this case replacing A and B with $\rho(s)$ when considering δ and $\rho'(s)$ when considering δ', if such a $i^* \leq l$ exists for a given l, the same i^* works for $l' > l$.

Thus, $\succ_{lmaxlmin,l'}$ refines $\succ_{lmaxlmin,l}$.

□

It is interesting to use this procedure in the backward induction algorithm; that is why we propose a variant of of Algorithm 4.3, which we call "*Bounded Lexicographic Backward Induction*" $(BLex\text{-}BI\text{-}MDP)$ by simply replacing line 14 by line 14':

$$14': \quad \rho(s) \leftarrow [M]_l$$

where $[M]_l$ denotes the restriction of M to its first l lines.

Clearly, this algorithm is not guaranteed to provide a lexi-optimal solution, but the policy is always at least as good as the one provided by u_{opt} (according to $lmax(lmin)$). Indeed, bounding the matrices is done *after* they have been ordered. Hence $M_{1,1}$ is equal to u_{opt} in the unbounded case and because the bounding is done after reordering, this property still holds when using bounded lexicographic backward induction. We deduce that the order on matrices (and thus on policies) refines the one provided by classical optimistic backward induction algorithm. The optimal policy for bounded lexicographic backward induction is optimal for the optimistic utility. Actually, the greater l, the more refined the comparison over the policies. This comparison goes to $\succ_{lmax(lmin)}$ when l goes to b^h.

The same behavior obviously holds in the pessimistic case: the optimal policy for bounded lexicographic backward induction is optimal for the pessimistic utility and the greater l, the more refined the comparison over the policies and the comparison goes to $\succ_{lmin(lmax)}$ when l goes to b^h.

The temporal complexity of bounded backward induction is decreased, compared to that of the lexicographic backward induction. Indeed, the number of calls to *ConcatAndOrder* of the algorithm does not change, however, the ConcatAndOrder algorithm is only called on sets of matrices which have at most l lines, instead of b^h. We deduce that the complexity of the algorithm is bounded by $= O(n \cdot l \cdot (h-1)^2 \cdot log((h-1) \cdot l)$, where n is the (maximal) number of states in S_t.

4.4 Experimental study

In this Section, we evaluate different solving algorithms proposed in this chapter. We have two criteria for each of the pessimistic and optimistic approaches: the basic possibilistic one and the lexicographic refinement. These criteria aim at solving the same decision problems: sequential decision under possibilistic uncertainty, represented by a $\Pi\mathcal{DT}$ or a finite-horizon $\Pi\mathcal{MDP}$.

We compare the algorithms with two measures:

- the CPU time.
- pairwise success rate: $Success_{\frac{A}{B}}$ is the percentage of solutions provided by an algorithm optimizing criterion A that are optimal with respect to criterion B; for instance, the less $Success_{\frac{u_{opt}}{lmax(lmin)}}$, the more important the drowning effect.

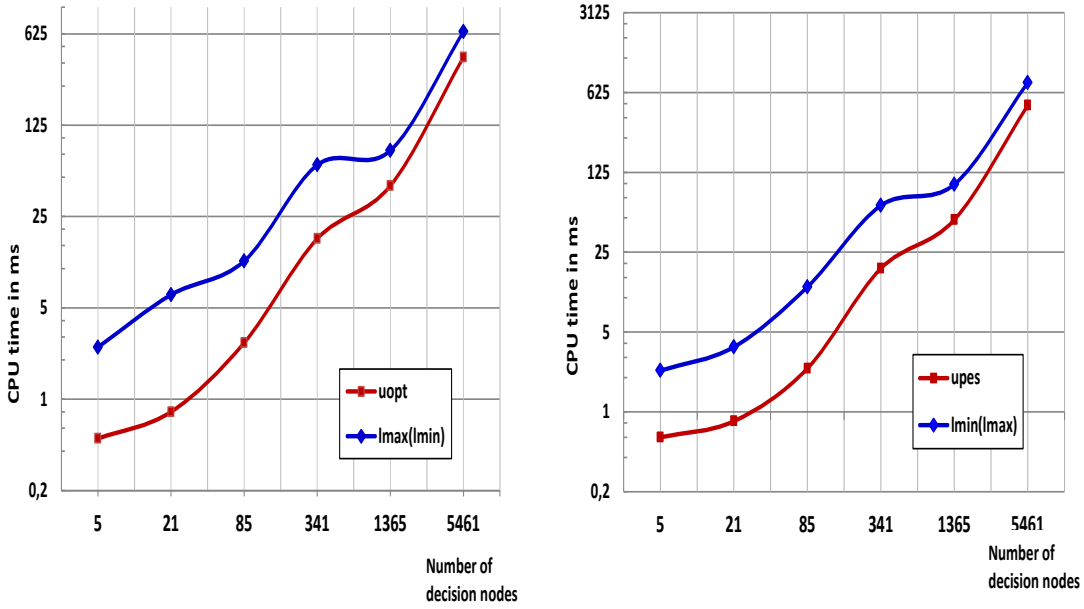
The algorithms corresponding to these criteria have been implemented in Java. The experiments have been performed on an Intel Core i5 processor computer (1.70 GHz) with 8GB DDR3L of RAM (all experiments in this section are made using the same computer).

4.4.1 Experimental results in $\Pi\mathcal{DT}$s

In what follows, we propose to compare the performance of *Lexicographic Backward Induction algorithm* (Algorithm 4.1), so-called $LexBI\text{-}DT$, to the classical *Backward Induction algorithm* (Algorithm 2.1), so-called $BI\text{-}DT$.

The tests were performed on randomly generated complete binary decision trees, from $h = 2$ to $h = 7$. The first node is a decision node: at each decision level from the root ($i = 1$) to the last level ($i = 7$) the tree contains 4^{i-1} decision nodes. This means that with $h = 2$ (resp. 3, 4, 5, 6, 7), the number of decision nodes is equal to 5 (resp. 21, 85, 341, 1365, 5461) The utility values are uniformly randomly drawn in the set $V = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$. Conditional possibilities relative to chance nodes are normalized, one edge having possibility one and the possibility degree of the other being uniformly drawn in L. For each value of h, 100 $\Pi\mathcal{DT}$s are generated.

Figure 4.3 presents the average execution CPU time for the four criteria, the two optimistic ones (Figure 4.5 (a)) and the two pessimistic ones (Figure 4.5 (a)). We observe that, whatever the optimized criterion, the CPU time increases linearly w.r.t. the number of decision nodes, which is in line with what we could expect. Furthermore, it remains affordable with big trees: the maximal CPU time is lower than $1s$ for a $\Pi\mathcal{DT}$ with 5461 decision nodes. It appears that u_{opt} is always faster than $lmax(lmin)$. The same conclusions are drawn when comparing $lmin(lmax)$ to u_{pes}. These results are easy to explain: the manipulation of matrices is obviously more demanding than the one of numbers.



(a) Average CPU time for optimistic criteria

(b) Average CPU time for pessimistic criteria

Figure 4.3: Average CPU time of (a) optimistic case and (b) possimistic case for $\Pi\mathcal{DT}$s

As to the success rate, the results are described in Figure 4.4. The percentage of solutions optimal w.r.t. u_{opt} (resp. for u_{pes}) that are also optimal w.r.t. $lmax(lmin)$ (resp. $lmin(lmax)$) is never more than 82%, and decreases when the horizon increases: the drowning effect is not negligible and increases with the length of the trajectories.

These experiments conclude in favor of the lexicographic refinements: the longer the horizon the more significant the drowning effect of u_{opt} and u_{pes}. $LexBI\text{-}DT$ algorithm computes the optimal solutions even when the horizon increases contrary to the classical $BI\text{-}DT$ algorithm.

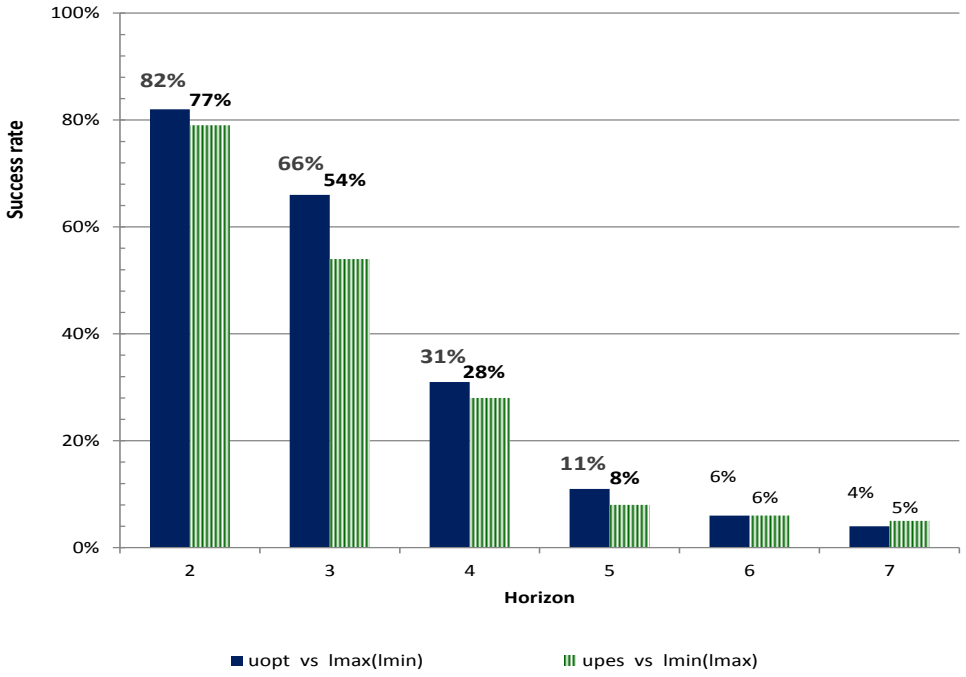


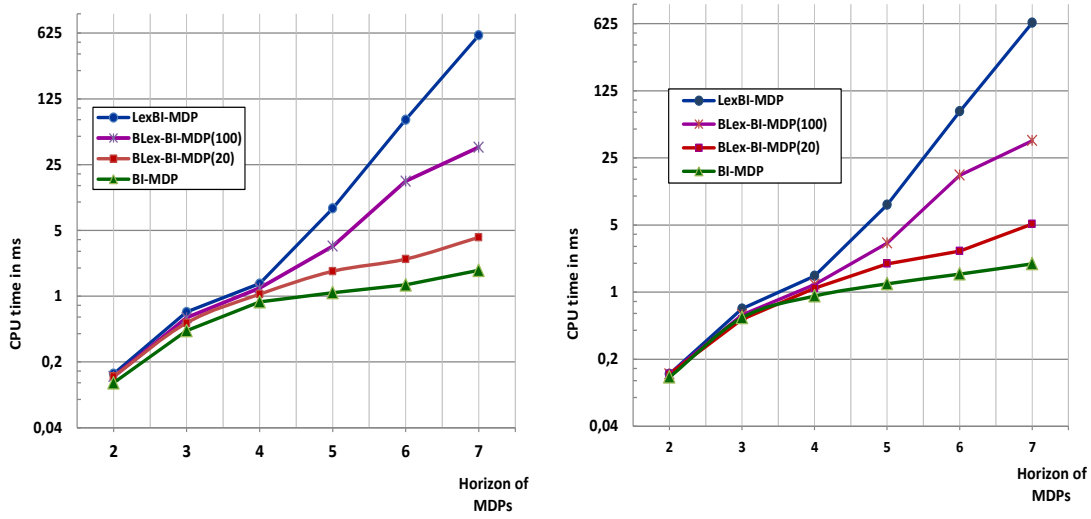
Figure 4.4: Success rate for $\Pi\mathcal{DT}$s with h=2 to 7

4.4.2 Experimental results in finite-horizon $\Pi\mathcal{MDP}$s

For finite-horizon $\Pi\mathcal{MDP}$s, we propose to compare the performance of *Bounded Lexicographic Backward Induction*, so-called $BLex\text{-}BI\text{-}MDP$, as an approximation of (unbounded) Lexicographic Backward Induction (Algorithm 4.3), so-called $LexBI\text{-}MDP$, and also Backward Induction (Algorithm 2.2), so-called ($BI\text{-}MDP$) for pessimistic and optimistic utilities (u_{opt} and u_{pes}), in randomly generated finite-horizon $\Pi\mathcal{MDP}$s for $h = 2$ to $h = 7$. In each stage, the $\mathcal{MDP}$ contains 20 states and the number of actions in each state is equal to 4. The output of each action is a distribution on two states randomly drawn (i.e. the branching factor is equal to 2). The utility values are uniformly randomly drawn in the set $V = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$. Conditional possibilities relative to decisions should be normalized. To this end, one choice is fixed to possibility degree 1 and the possibility degree of the remaining one is uniformly drawn in L. For each value of h, 100 finite-horizon $\Pi\mathcal{MDP}$s are generated.

Figure 4.5 presents the average execution CPU time for the three algorithms. We observe that the CPU time increases linearly w.r.t. the horizon for $BI\text{-}MDP$ for both u_{opt} (Figure 4.5 (a)) and u_{pes} (Figure 4.5 (b)). It seems to be also the case for $BLex\text{-}BI\text{-}MDP$. On the other hand, it increases exponentially for $LexBI\text{-}MDP$.

We also observe that $BLex\text{-}BI\text{-}MDP$, with $l = 20$, is slower than $BI\text{-}MDP$ but the CPU time remains affordable, as the maximal CPU time is $5ms$ for 100 finite-horizon $\Pi\mathcal{MDP}$s with 25 states when $l = 20$ and $h = 7$. Unsurprisingly, we can check that the $BLex\text{-}BI\text{-}MD$ is faster than $LexBI\text{-}MDP$ especially when the horizon increases: the manipulation of $l \times (h+1)$-matrices is obviously less expensive than the one of full matrices. The saving increases with the horizon.



(a) Average CPU time for optimistic criteria

(b) Average CPU time for pessimistic criteria

Figure 4.5: Average CU time of (a) optimistic case and (b) possimistic case for finite-horizon $\Pi\mathcal{MDP}$s

As to the success rate, the results of the optimistic versions of the algorithms are described in Figure 4.6. The percentage of optimal solutions for u_{opt} that are also optimal for $lmax(lmin)$ when considering the whole matrices is never more than 60%, and decreases when the horizon increases. Indeed, if we take an arbitrary optimistic optimal policy, the higher the problem size the lower its chance of being lexi-optimal. We observe that the drowning effect increases with the length of the trajectories.

It also appears that $BLex\text{-}BI\text{-}MDP$ provides a very good approximation for reasonable values of l. Of course, the greater l the greater the quality of the approximation. $BLex\text{-}BI\text{-}MDP$ provides the same optimal solution as the $LexBI\text{-}MDP$ in about 80% of cases, with $l = 100$. Moreover, even when the success rate of $BLex\text{-}BI\text{-}MDP$ decreases (when h increases), the quality of approximation is still good: never less than 70% of optimal actions returned, with $l = 100$.

These experiments conclude in favor of bounded lexicographic backward induction: its approximated solutions are comparable with the optimal policy in terms of quality for high l and increase when l increases, while it is much faster than the classical unbounded version.

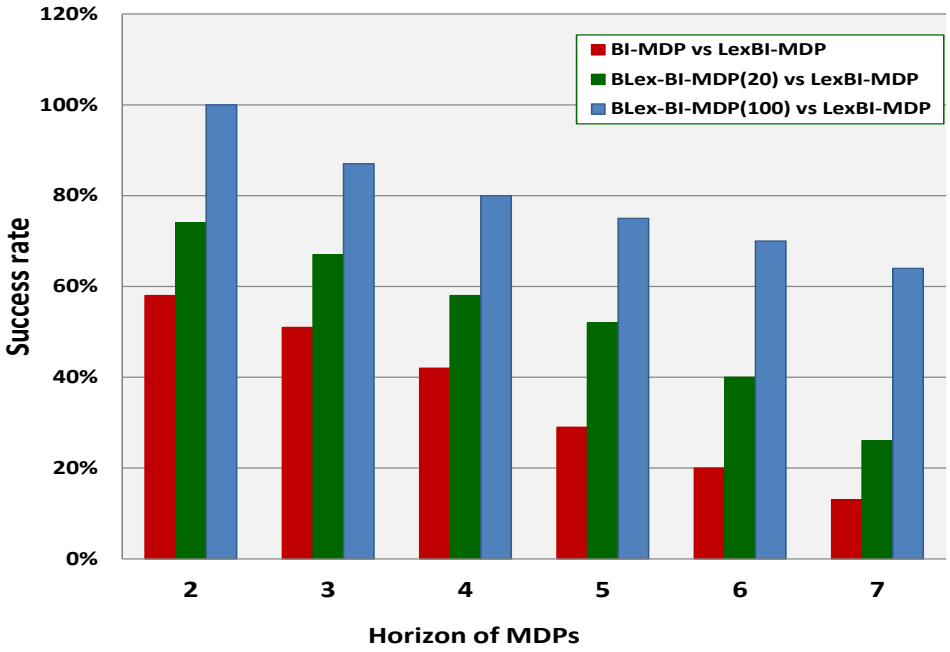


Figure 4.6: Success rate for optimistic criteria in finite-horizon Π$\mathcal{MDP}$s

4.5 Summary

In this Chapter, we have proposed new planning algorithms (based on Dynamic Programming), for Π$\mathcal{DT}$s and finite-horizon Π$\mathcal{MDP}$s. These algorithms allow to overcome the drowning effect by calculating lexicographic optimal policies. The complexity of the backward induction algorithm depends on the number of trajectories in the optimal policy - and is thus exponential in time and space (with respect to the size of the decision model). We show that an approximate policy can be computed in controlled, polynomial time and space using bounded matrices. Moreover, we have performed experiments on Π$\mathcal{DT}$s and finite-horizon Π$\mathcal{MDP}$s built randomly in order to show the discrimination power of lexicographic algorithms.

In the next Chapter, we show that, for Π$\mathcal{DT}$s and finite-horizon Π$\mathcal{MDP}$s, the lexicographic criteria can be captured by an EU criterion, relying on big stepped probabilities and utilities.

CHAPTRE 5

Expected utility refinements in sequential decision-making

Contents

5.1 Introduction

As we have detailed in Section 1.5.2, [Fargier and Sabbadin, 2003, Fargier and Sabbadin, 2005] have shown that, when the problem is not sequential, the comparison of possibilistic utility distributions by $\succeq_{lmax(lmin)}$ and $\succeq_{lmin(lmax)}$ can be captured by an expected utility, relying on infinitesimal probabilities and utilities.

In this Chapter, we show that such a result can be extended to finite-horizon sequential problems. We focus only on $\Pi\mathcal{DT}$s, since a finite-horizon $\Pi\mathcal{MDP}$ can be translated into a set of $\Pi\mathcal{DT}$s (one for each state). So, the comparison of the policies of a finite-horizon $\Pi\mathcal{MDP}$ can be refined using the same method as the one relative to $\Pi\mathcal{DT}$s.

The next Section develops our proposition, defining infinitesimal expected utilities, that refine qualitative utilities, in decision trees. Besides, we show that the 'qualitative' lexicographic criteria presented in Chapter 3 can be represented by these expected utilities. Then, in Section 5.3, we

propose an adaptation of backward induction algorithm in order to optimize these new EU-based refinements. Finally, Section 5.4 is dedicated to the experimental study in order to verify the quality of this latter algorithm comparing to the lexicographic backward induction one.

The main results of this chapter are published in [Ben Amor et al., 2015, Ben Amor et al., 2016a].

5.2 EU-refinements in decision trees

In this section we aim to extend the work of [Fargier and Sabbadin, 2003, Fargier and Sabbadin, 2005], on the refinements of qualitative utilities by expected utilities, to the sequential case i.e. to decision trees.

In order to refine possibilistic utilities in decision trees, we present here two criteria based on infinitesimal expected utilities as refinements of optimistic and pessimistic utilities, in lotteries. Thus we propose to transform the $\Pi\mathcal{DT}$ into a probabilistic one and then we optimize the probabilistic tree using a backward induction algorithm. The graphical components of the two trees are identical and so are the sets of admissible policies.

5.2.1 EU-refinement of optimistic utility

In the optimistic case the probability and utility distributions are chosen in such a way that the $lmax(lmin)$ and EU criteria do provide the same preference on Δ. To this extent, we build a transformation of the scale L, $\phi : V \subseteq [0,1] \rightarrow [0,1]$, that maps each possibility distribution to an additive distribution and each utility level into an additive one; this transformation is required to satisfy the following condition:

$$(R) : \forall \alpha, \alpha' \in V \text{ such that } \alpha > \alpha' : \phi(\alpha)^{h+1} > b^h \phi(\alpha'),$$

where b is the branching factor of the tree and h its depth. Then, For any chance node C_j, a local transformation ϕ_j is then derived from ϕ, such that ϕ_j satisfies both condition (R) and the normalization condition of probability theory.

Using ϕ and ϕ_j, we define the two following functions:

- ϕ^- s.t. $\forall \alpha \in V$, $\phi^-(\alpha) = \min\{\phi(\alpha),\ min_j\ \phi_j(\alpha))\}$,
- ϕ^+ s.t. $\forall \alpha \in V$, $\phi^+(\alpha) = \max\{\phi(\alpha),\ max_j\ \phi_j(\alpha))\}$.

ϕ^- and ϕ^+ are required to satisfy the condition:

$$(R') : \forall \alpha, \alpha' \in V \text{ such that } \alpha > \alpha' : \phi^-(\alpha)^{h+1} > b^h \phi^+(\alpha'),$$

Then condition (R') guarantees that if $u_{opt}(\delta) = \alpha > u_{opt}(\delta') = \alpha'$, then a comparison based on a sum-product approach on the new tree will also decide in favor of δ.

EU_{opt} denotes the preference relation provided by the EU-criterion on the probabilistic tree obtained by replacing each π_j by $\phi_j \circ \pi_j$ and the utility function u by $\phi \circ u$. We show that:

Proposition 5.1.
If ϕ satisfies (R) and the derived ϕ^+, ϕ^- satisfy (R'), then $\succeq_{EU_{opt}}$ refines $\succeq_{u_{opt}}$.

Proof of Proposition 5.1.
For any transformation function ϕ s.t. $\forall(\alpha, \alpha') \in V, \ \ \alpha > \alpha'$ it holds that $(R) : \phi(\alpha)^{h+1} > b^h\phi(\alpha')$. Note that, by definition, $\phi^+(\alpha) \geq \phi(\alpha) \geq \phi^-(\alpha), \forall\alpha \in V$.

Thus, by (R'),

$$\phi^-(\alpha)^{h+1} > b^h\phi^+(\alpha') \geq b^h\phi^-(\alpha'), \forall(\alpha, \alpha') \in V, \ \ \alpha > \alpha'.$$

Let δ and δ' be two strategies. Assume that $u_{opt}(\delta) = \alpha > u_{opt}(\delta') = \alpha'$ and let us show that $EU_{opt}(\delta) > EU_{opt}(\delta')$.

- $u_{opt}(\delta) = \alpha \Rightarrow \exists\tau^* = (\pi_{j_0}(x_{i_1}), \ldots, \pi_{j_{h-1}}(x_{i_h}), u(x_{i_h}))$ in δ s.t. $\min(\pi_{j_0}(x_{i_1}), \ldots, \pi_{j_{h-1}}(x_{i_h}), u(x_{i_h})) \geq \alpha$.
  Since $EU_{opt}(\delta) = \sum_{\tau}(\prod_{k=1}^{h} \phi_k(\pi_{j_{k-1}}(x_{i_k})) * \phi(\mu(x_{i_k})))$ and all terms of the sum are positive or zero, by keeping only trajectory τ^* in the sum, we get: $EU_{opt}(\delta) \geq \prod_{k=1}^{h} \phi_k(\pi_{j_{k-1}}(x_{i_k})) * \phi(\mu(x_{i_k})) \geq \prod_{k=1}^{h} \phi_k(\alpha) * \phi(\alpha)$.
  Since[1] $\phi^-(\alpha) \leq \phi_k(\alpha), \forall k, \forall\alpha \in L$, $EU_{opt}(\delta) \geq \prod_{k=1}^{h} \phi^-(\alpha) * \phi(\alpha)$.
  Then, since $\phi^-(\alpha) \leq \phi(\alpha)$, $EU_{opt}(\delta) \geq \prod_{k=1}^{h} \phi^-(\alpha) * \phi^-(\alpha)$.
  Thus, we get:
  $$EU_{opt}(\delta) \geq \phi^-(\alpha)^{h+1} \tag{5.1}$$

- $u_{opt}(\delta') = \alpha' \Rightarrow \forall\tau, \ \min(\pi_{j'_0}(x_{i_1}), \ldots, \pi_{j'_{h-1}}(x_{i'_h}), u(x_{i'_h})) \leq \alpha'$.
  Let us denote EU_τ the term of $EU_{opt}(\delta')$ corresponding to trajectory τ. We have $EU_\tau \leq \prod_{k=1}^{h} \phi_k(\alpha') * \phi(\alpha') \leq \prod_{k=1}^{h} \phi^+(\alpha') * \phi(\alpha') \leq \prod_{k=1}^{h} \phi^+(\alpha') * \phi^+(\alpha')$, since $\phi^+(\alpha') = \max\{\phi(\alpha'), \max_k \phi_k(\alpha')\}, \forall\alpha' \in V$.
  Then, $EU_\tau \leq \phi^+(\alpha')^{h+1} \leq \phi^+(\alpha')$ since $0 \leq \phi^+(\alpha') \leq 1$.
  Since δ' generates at most b^h trajectories, $EU_{opt}(\delta') \leq b^h EU_\tau$ and
  $$EU_{opt}(\delta') \leq b^h.\phi^+(\alpha'). \tag{5.2}$$

[1] Recall that $\phi^-(\alpha) = \min\{\phi(\alpha), \min_k \phi_k(\alpha)\}$.

Finally, using (5.1), (5.2) and (R'), we get $u_{opt}(\delta) > u_{opt}(\delta') \Rightarrow EU_{opt}(\delta) > EU_{opt}(\delta')$.

□

Proposition 5.2.
$\delta \succeq_{lmax(lmin)} \delta'$ *iff* $\delta \succeq_{EU_{opt}} \delta', \forall(\delta, \delta') \in \Delta$.

Proof of Proposition 5.2.
For the sake of notational simplicity, we will associate any trajectory τ (resp. τ') with the vector τ^{Ord} (resp. τ'^{Ord}) consisting in reordering $(\pi_1, \ldots, \pi_h, u_h)$ (resp. $(\pi'_1, \ldots, \pi'_h, u'_h)$) in increasing order.

Obviously, $\tau \succeq_{lmin} \tau'$ iff $\tau^{Ord} \succeq_{lmin} \tau'^{Ord}$. Note that $\delta \succeq_{lmax(lmin)} \delta'$ iff either

1. $\forall i, \tau_{\lambda(i)} \sim_{lmin} \tau'_{\lambda(i)}$ or
2. $\exists i^*, \forall i \leq i^*, \tau_{\lambda(i)} \sim_{lmin} \tau'_{\lambda(i)}$ and $\tau_{\lambda(i^*)} \succ_{lmin} \tau'_{\lambda(i^*)}$.

$\forall\ \alpha_k \in \tau^{Ord}$ (resp. $\alpha'_k \in \tau'^{Ord}$) s.t. $k = 1, h+1$, note the following facts concerning pairs of trajectories (τ, τ'):

1. $\tau \sim_{lmin} \tau' \Leftrightarrow \prod_{k=1}^{h+1} \phi(\alpha_k) = \prod_{k=1}^{h+1} \phi(\alpha'_k)$, since $\tau \sim_{lmin} \tau' \Leftrightarrow \tau^{Ord} \sim_{lmin} \tau'^{Ord}$.
   Then: $\delta \sim_{lmax(lmin)} \delta' \Leftrightarrow \tau_{\lambda(i)} \sim_{lmin} \tau'_{\lambda(i)}, \forall i$.
   $\Rightarrow \sum_t (\prod_{k=1}^{h+1} \phi(\alpha_k)) = \sum_t (\prod_{k=1}^{h+1} \phi(\alpha'_k))$
   $\Leftrightarrow EU_{opt}(\delta) = EU_{opt}(\delta')$.
   Thus $\delta \sim_{lmax(lmin)} \delta' \Leftrightarrow EU_{opt}(\delta) = EU_{opt}(\delta')$.
2. if $\delta >_{lmax(lmin)} \delta'$, then $\exists i^*\ \ s.t.\ \ \tau_{\lambda(i^*)} \succ_{lmin} \tau'_{\lambda(i^*)}$, then $\exists j^*, \forall j < j^*, \alpha_j = \alpha'_j$ and $\alpha_{j^*} > \alpha'_{j^*}$. Then, let us compare the product of transformed possibilities/utilities along τ and τ':

$$\begin{aligned} \prod_{k=1}^{h+1} \frac{\phi(\alpha_k)}{\phi(\alpha'_k)} &= \prod_{k=j^*}^{h+1} \frac{\phi(\alpha_k)}{\phi(\alpha'_k)} \\ &\geq \frac{\phi(\alpha_{j^*})}{\phi(\alpha'_{j^*})} * \frac{\phi(\alpha_{j^*})^{h-j^*}}{\phi(\phi(1_V)^{h-j^*})} \end{aligned} \tag{5.3}$$

   (lower and upper bounds on trajectories degrees)

$$\begin{aligned} &\geq \frac{\phi(\alpha_{j^*})^{h-j^*+1}}{\phi(\alpha'_{j^*})} && (\phi(1_V) \leq 1) \\ &\geq \frac{\phi(\alpha_{j^*})^{h}}{\phi(\alpha'_{j^*})} && (\phi(\alpha_{j^*}) \leq 1) \\ &> b^h. && \text{(From R)} \end{aligned}$$

Then, since trajectories are ordered along $\succeq_{lmax(lmin)}$, $\tau'_{\lambda(i^*)} \succ_{lmin} \tau'_{\lambda(i)}, \forall i > i^*$, and since there are no more than b^h such trajectories $\tau'_{\lambda(i)}$, we get that $EU_{opt}(\delta) > EU_{opt}(\delta')$. Thus $\delta \succ_{lmax(lmin)} \delta' \Rightarrow EU_{opt}(\delta) > EU_{opt}(\delta')$.

So, we have just proved that $\delta \succeq_{lmax(lmin)} \delta' \Rightarrow \delta \succeq_{EU_{opt}} \delta'$. Thus, $\succeq_{lmax(lmin)}$ is equivalent to $\succeq_{EU_{opt}}$. □

Example 5.1. *We illustrate in the following an example of transformation of the decision tree of Figure 5.1 with $h = b = 2$.*

First, let us build a function ϕ satisfying (R'). For this purpose, it is sufficient to construct the function $\phi : V \to \mathcal{R}$ as follows :
$\phi(1_V) = 1$, $\phi(\alpha_i) < \frac{\phi(\alpha_{i+1})^3}{4}$ *(if* $V = 1_V, \alpha_1, \ldots, \alpha_k = 0_V$*).*

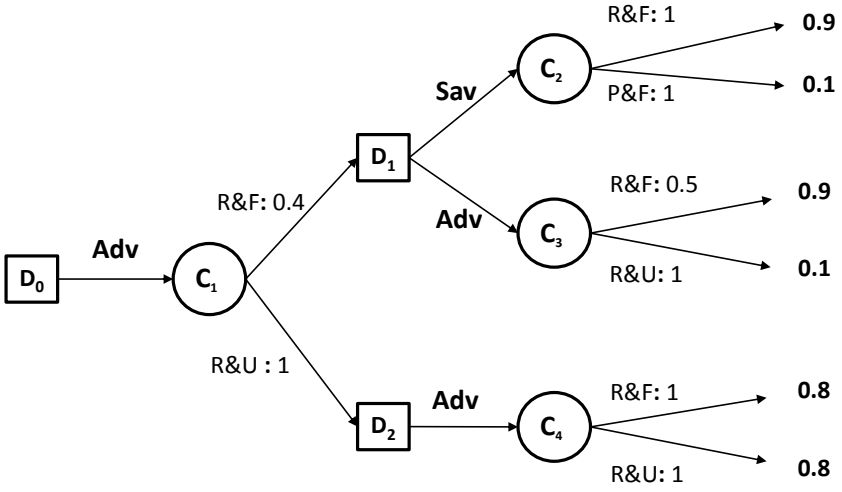


Figure 5.1: The $\Pi\mathcal{DT}$ of Example 5.1

Applying this function ϕ on the scale $V = \{0, 0.1, 0.4, 0.5, 0.8, 0.9, 1\}$ we obtain:

- $\phi(1) = \mathbf{1}$,
- $\phi(0.9) = \mathbf{0.2}$,
- $\phi(0.8) = \mathbf{0.001}$,
- $\phi(0.5) = \mathbf{10^{-10}}$,
- $\phi(0.4) = \mathbf{10^{-30}}$,
- $\phi(0.1) = \mathbf{10^{-91}}$.

We obtain the transformed probabilistic decision tree of Figure 5.2, by normalizing the transformed conditional distributions obtained in each node. For instance:

- *for node* $C_1 = \begin{cases} \phi_1(10^{-30}) = \frac{10^{-30}}{1+10^{-30}}, \textit{and} \\ \phi_1(1) = \frac{1}{1+10^{-30}}. \end{cases}$
- *for node* $C_2 = \begin{cases} \phi_2(1) = \frac{1}{1+1}, \textit{and} \\ \phi_2(1) = 0.5, \end{cases}$
- *for node* $C_3 = \begin{cases} \phi_3(10^{-10}) = \frac{10^{-10}}{1+10^{-10}}, \textit{and} \\ \phi_3(1) = \frac{1}{1+10^{-10}}, \end{cases}$
- *for node* $C_4 = \begin{cases} \phi_4(1) = 0.5, \textit{and} \\ \phi_4(1) = 0.5. \end{cases}$

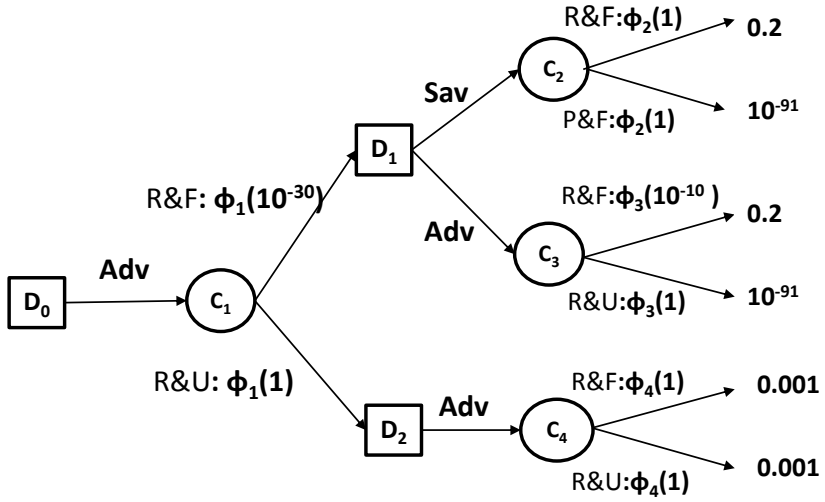


Figure 5.2: Transformed probabilistic decision tree of $\Pi\mathcal{DT}$ of Example 5.1

5.2.2 EU-refinement of pessimistic utility

In what follows, we focus on the pessimistic case. Consider the $\succeq_{lmin(lmax)}$ comparison, the utility degrees are not directly compared to possibility degrees π but to degrees $1-\pi$. Hence, it is possible to show that:

Proposition 5.3.
Let $\mathcal{DT}^{inv}$ the tree obtained from $\mathcal{DT}$ by using utility function $u' = 1 - u$ on leaves. It holds that: $u_{pes,\mathcal{DT}}(\delta) \geq u_{pes,\mathcal{DT}}(\delta')$ *iff* $u_{opt,\mathcal{DT}^{inv}}(\delta') \geq u_{opt,\mathcal{DT}^{inv}}(\delta)$

Proof of Proposition 5.3.
Let L_δ (resp. $L_{\delta'}$) be the equivalent simple lottery of the compound lottery representing the strategy δ. $L_\delta = \langle \pi_1/u_1, ..., \pi_i/u_i, ..., \pi_p/u_p \rangle$ (resp. $L_{\delta'} = \langle \pi'_1/u'_1, ..., \pi'_i/u'_i, ..., \pi'_p/u'_p \rangle$) i.e. $\pi_i = \pi(u_i)$ (resp. $\pi'_i = \pi(u'_i)$) is the possibility that the strategy leads to the outcome utility u_i (resp. u'_i).
Let us show that $u_{pes,\mathcal{DT}}(\delta) \geq u_{pes,\mathcal{DT}}(\delta') \Leftrightarrow u_{opt,\mathcal{DT}^{inv}}(\delta) \leq u_{opt,\mathcal{DT}^{inv}}(\delta')$ where $u_{pes,\mathcal{DT}}(\delta)$ denotes the pessimistic utility of δ when considering the original decision tree $\mathcal{DT}$ and $u_{opt,\mathcal{DT}^{inv}}(\delta)$ denotes the optimistic utility of δ when considering the decision tree $\mathcal{DT}^{inv}$, obtained from $\mathcal{DT}$ by using utility function $u' = 1 - u$.
We have, $u_{pes,\mathcal{DT}}(\delta) \geq u_{pes,\mathcal{DT}}(\delta') \Leftrightarrow u_{pes,\mathcal{DT}}(L_\delta) \geq u_{pes,\mathcal{DT}}(L_{\delta'})$.
It has been shown in [Fargier and Sabbadin, 2005] that:
$u_{pes,\mathcal{DT}}(L_\delta) \geq u_{pes,\mathcal{DT}}(L_{\delta'}) \Leftrightarrow u_{opt,\mathcal{DT}^{inv}}(L_{\delta'}) \geq u_{opt,\mathcal{DT}^{inv}}(L_\delta)$.

Hence, we deduce that: $u_{opt,\mathcal{DT}^{inv}}(\delta') \geq u_{opt,\mathcal{DT}^{inv}}(\delta)$.

□

As a consequence, we build an EU-based equivalent of $\succeq_{lmin(lmax)}$, denoted by $\succeq_{EU_{pes}}$, by replacing each possibility distribution π_i in $\mathcal{DT}$ by the probability distribution $\phi_i \circ \pi_i$, as for the optimistic case and each utility degree u by$\phi(1) - \phi(u)$. It is then possible to show that:

Proposition 5.4.
$\delta \succeq_{lmin(lmax)} \delta'$ *iff* $\delta \succeq_{EU_{pes}} \delta', \forall (\delta, \delta') \in \Delta$.

Proof of Proposition 5.4.
It is sufficient to show that:

$$\delta \succeq^{\mathcal{DT}}_{lmax(lmin)} \delta' \Leftrightarrow \delta' \succeq^{\mathcal{DT}^{inv}}_{lmin(lmax)} \delta, \tag{5.4}$$

where $\succeq^{\mathcal{DT}^{inv}}_{lmin(lmax)}$ is the pessimistic lexicographic comparison of strategies in the decision tree where the utilities of all leaves have been reversed ($u'(N) = 1 - u(N)$).

For any two strategies δ and δ':

- if $\delta \succ^{\mathcal{DT}}_{lmax(lmin)} \delta'$ then $\exists i^*, \forall i \leq i*, \tau_{\lambda(i)} \sim_{lmin} \tau'_{\lambda(i)}$
  and $\tau_{\lambda(i^*)} \succ_{lmin} \tau'_{\lambda(i^*)} \Leftrightarrow$
  $(\pi_1, \ldots, \pi_h, u_h) \succ_{lmin} (\pi'_1, \ldots, \pi'_h, u'_h)$.
  Now let invert each degree in both trajectories, we get:
  $((1 - \pi_1), \ldots, (1 - \pi_h), (1 - u_h)) \prec_{lmax} ((1 - \pi'_1), \ldots, (1 - \pi'_h), (1 - u'_h))$

$\Leftrightarrow ((1-\pi_1),\ldots,(1-\pi_h),u_h) \prec_{lmax} ((1-\pi'_1),\ldots,(1-\pi'_h),u'_h)$ i.e. $\tau_{\sigma(i^*)} \prec_{lmax} \tau'_{\sigma(i^*)}$ which is the $\succeq_{lmax}$ relation when considering $\mathcal{DT}^{inv}$. We get $\delta' \succeq^{\mathcal{DT}^{inv}}_{lmin(lmax)} \delta$.

- if $\delta \sim^{\mathcal{DT}}_{lmax(lmin)} \delta'$ then $\forall i,\ \tau_{\lambda(i)} \sim_{lmin} \tau'_{\lambda(i)} \Leftrightarrow$
  $(\pi_1,\ldots,\pi_h,u_h) \sim_{lmin} (\pi'_1,\ldots,\pi'_h,u'_h)$.
  If we invert each degree in both trajectories, we get:
  $((1-\pi_1),\ldots,(1-\pi_h),(1-u_h)) \sim_{lmax} ((1-\pi'_1),\ldots,(1-\pi'_h),(1-u'_h)$
  $\Leftrightarrow ((1-\pi_1),\ldots,(1-\pi_h),u_h) \sim_{lmax} ((1-\pi'_1),\ldots,(1-\pi'_h),u'_h)$ i.e. $\tau_{\sigma(i)} \sim_{lmax} \tau'_{\sigma(i)}$.
  We get $\delta' \sim^{\mathcal{DT}^{inv}}_{lmin(lmax)} \delta$.

Thus, $\succeq_{EU_{pes}}$ is equivalent to $\succeq_{lmin(lmax)}$. □

Propositions 5.2 and 5.4 show that lexicographic comparisons have a probabilistic interpretation - actually, using infinitesimal probabilities and utilities. This result comforts the idea, first proposed by [Benferhat et al., 1999] and then by [Fargier and Sabbadin, 2005], of a bridge between qualitative approaches and probabilities, through the notion of big stepped probabilities [Benferhat et al., 1999, Snow, 1999]. But here we make a step further, since the proposed transformations support sequential decision-making.

5.3 Backward induction algorithm for EU-refinements

Beyond this theoretical argument, this result suggests an alternative algorithm for the optimization of $lmax(lmin)$ (resp. $lmin(lmax)$). The proposed algorithm, denoted $EU\text{-}BI$ (see Algorithm 5.1 for the optimistic version), simply transform the $\Pi\mathcal{DT}$ into a probabilistic one, transforming u into $\phi \circ u$ and each π_i into a probability distribution $p_i = \phi_i \circ \pi_i$, and use an adaptation of classical, EU-based backward induction. This algorithm is optimistic (resp. pessimistic) i.e. optimize EU_{opt} (resp. EU_{pes}) if it uses an optimistic transformation (resp. pessimistic transformation) of the decision tree. Note that the horizon h and the branching factor b are known in advance. In this algorithm, we first transform the possibilistic scale V to a probabilistic one $V_p = (\phi \circ V)$ using the adequate transformation function that we get using only h and b. It is important that we keep the two vectors associated to the two scales, each value in V has its probabilistic counterpart in V_p in same position.

The principle of EU-based backward induction is as follows:

- When a leaf node N is reached, its expected utility is $u_{V_p}(N)$ the probabilistic utility in V_p that matches $u(N)$ the possibilistic utility in leaf N.

- When each chance node is reached, an optimal sub-strategy is built for each of its children: the EU of each sub-strategy is returned and the probability degree of having this children is computed, it is simply the probability in V_p that matches $\pi_N(Y_i)$ the possibilistic degree of having Y_i in chance node N. We also update the sum of the probability degrees in this chance node (we will use it for normalizing the probability distribution). Then it is possible to compute the expected utility of the current chance node: the sum-product of the normalized probability of each children and the corresponding expected utility in next stage.
- When a decision node N is reached, a decision $\delta(N)$ leading to a sub-strategy optimal is selected among all the possible decisions $C_j \in Succ(N)$, by compared the expected utility of each sub-strategy.

Algorithm 5.1: $EU\text{-}BI$: Backward-Induction-$\mathcal{DT}$-EU(N:Node)

Data: A probabilistic DT ; the policy, δ, is memorized as global variable
Result: Set δ for the tree rooted in N and returns its expected utility

begin
  $V_p \leftarrow transform(V, b, h)$
  // Leaves
  if $N \in \mathcal{N}_{\mathcal{U}}$ **then** $EU \leftarrow u_{V_p}(N)$;
  // Chance nodes
  if $N \in \mathcal{N}_C$ **then**
    $k \leftarrow |Succ(N)|$;
    foreach $Y_i \in Succ(N)$ **do**
      $EU^i \leftarrow Backward\text{-}Induction\text{-}\mathcal{DT}\text{-}EU(Y_i)$;
      $p_i \leftarrow \pi_{V_p}(Y_i)$;
      $Sum \leftarrow Sum + p_i$;
    $EU \leftarrow \sum_{i=1,k} ((p_i/Sum) \times EU^i)$;
  // Decision nodes
  if $N \in \mathcal{N}_D$ **then**
    $EU \leftarrow 0$;
    foreach $C_j \in Succ(N)$ **do**
      $U \leftarrow Backward\text{-}Induction\text{-}\mathcal{DT}\text{-}EU(C_j)$;
      if $U > EU$ **then**
        $EU \leftarrow U$;
        $\delta(N) \leftarrow label(N, C_j)$;
  return EU;

In a perfect world, the lexicographic approach and the big-stepped EU one solve the problem in the same way and provide the same optimal policies - the difference being that the lexicographic

backward induction is based on the comparison of matrices and the EU-backward induction is based on the computation of expected utilities in $\mathbb{R}^+$. The point is that the latter handles very small numbers; then either the program is based on an explicit handling of infinitesimals, and proceeds just like the matrix-based comparison, or it lets the programming language handle these numbers in its own way - and, given the precision of the computation, provides approximations.

5.4 Experimental study

In this Section we propose to compare the lexicographic criteria, and the EU approximations presented in the previous section. We compare the 2 variants of algorithms with two measures: the CPU time and the pairwise success rate ($Success_{\frac{A}{B}}$ where A and B are two decision criteria). We use the same experimental data of Chapter 4: 100 complete binary decision trees, for $h = 2$ to $h = 7$, that are randomly generated.

The backward induction algorithms corresponding to the optimistic and pessimistic criteria have been implemented in Java. As to the EU-based approaches, the transformation function depends on the horizon h and the branching factor b (here $b = 2$). We used $\phi(1_V) = 1$, and $\phi(\alpha_i) = \frac{\phi(\alpha_{i+1})^{h+1}}{b^h * 1.1}$.

Tables 5.1 and 5.2 present the execution CPU time of the proposed algorithms for respectively the optimistic criteria and the pessimistic criteria. Clearly, u_{opt} is always faster than EU_{opt}, which is 1.5 or 2 times faster than $lmax(lmin)$ The same conclusion is drawn when comparing $lmin(lmax)$ to u_{pes} and EU_{pes}. These results are easy to explain:
(i) the manipulation of matrices is obviously more expensive than the one of numbers and
(ii) the handling of numbers by min and max operations is faster than sum-product manipulations of very small numbers (infinitesimal).

Table 5.1: Average CPU time (in ms) for optimistic criteria in $\Pi\mathcal{DT}$s with h=2 to 7

	Number of decision nodes					
	5	**21**	**85**	**341**	**1365**	**5461**
$lmax(lmin)$	2.5	6.3	11.4	62	80	649
EU_{opt}	0.97	2.7	8.1	28.8	66	423
u_{opt}	0.5	0.8	2.7	16.9	43	414

Results relative to the success rate are presented in Figure 5.3. We can see that EU_{opt} (resp. EU_{pes}) performs well, as an approximation of $lmax(lmin)$ (resp. $lmin(lmax)$), indeed the percentage of solutions optimal for the former which are also optimal for the latter is greater than 65% in all cases, it is about 80% for $h = 3$ but it decreases when h increases. This is easily explained by the fact that the probabilities are infinitesimals and converge to 0 when the length of the branches (and thus the number of factors in the products) increases.

Table 5.2: Average CPU time (in ms) for pessimistic criteria in $\Pi\mathcal{DT}$s with h=2 to 7

	Number of decision nodes					
	5	**21**	**85**	**341**	**1365**	**5461**
$lmin(lmax)$	2.3	6	12.4	64	98	761
EU_{pes}	1.17	3.7	7.7	46	72	488
u_{pes}	0.6	0.83	2.4	18	48	481

These experiments conclude in favor of the lexicographic refinements in their full definition. When space and time are limited (or when h increases), it is interesting to use their approximations by expected utilities: they are better in terms of CPU time average and comparable in terms of average accuracy, but note that we lose about 20% of precision. The expected utilities criteria, nevertheless, are more decisive than possibilistic utilities (u_{opt} and u_{pes}), in all cases, and they are as fast as these latter.

5.5 Summary

This Chapter is devoted to the extension of the work of [Fargier and Sabbadin, 2003] on the refinement of the qualitative utilities by the expected utility to the case of the sequential decision. This refinement allows to establish a link between the possibilistic decision trees and probabilistic decision trees. The calculation of optimal policies by the refined expected utility can then be done by a backward induction algorithm, on the probabilistic transformed tree.

In the context of $\mathcal{MDP}$s, the comparison of policies of a $\Pi\mathcal{MDP}$ can be refined using the same method. The decision tree corresponding to the $\mathcal{MDP}$ is constructed; the trajectories of the $\mathcal{MDP}$ and those of the decision tree are in bijection. The rewards obtained in step $h-1$ are associated to the leaves of the tree. Then, it is possible to transform this decision tree using the same ϕ satisfying $\phi(\alpha)^{h+1} > b^{h+1}\phi(\alpha')$. The optimistic utility of a $\Pi\mathcal{MDP}$ policy is equal to its utility in the decision tree. For instance, if $\delta \succ_{u_{opt}} \delta'$ in the possibilistic $\mathcal{MDP}$, $\delta \succ_{u_{opt}} \delta'$ in the decision tree, and thus $\delta \succ_{EU_{opt}} \delta'$ which implies $\delta \succ_{EU_{opt}} \delta'$ in the $\mathcal{MDP}$.

The next Chapter focuses on the optimization of stationary $\mathcal{MDP}$s w.r.t. lexicographic criteria. We propose solving algorithms, based on the value iteration algorithm, to handle the finite-horizon case as well as the infinite-horizon case.

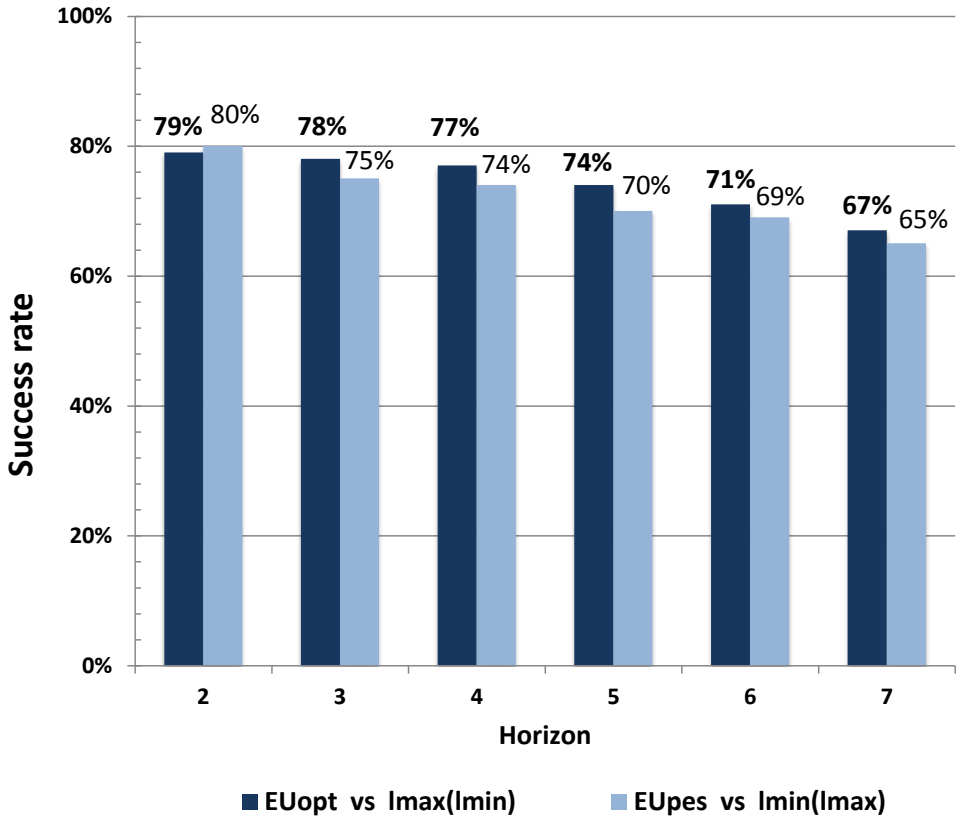


Figure 5.3: Success rate for lexicographic criteria vs EU-refinements in $\Pi\mathcal{DT}$

CHAPTRE 6

Optimizing lexicographic criteria in stationary $\Pi\mathcal{MDP}$s

Contents

6.1 Introduction

Previously, we have handled lexicographic optimization in finite-horizon decision models in which trajectories contained only one utility at the final step. However, as we have seen in Chapter 2, in Stationary Possibilistic Markov Decision Processes (stationary $\Pi\mathcal{MDP}$s) we consider intermediate utilities i.e. a satisfaction degree on each state s. So in this Chapter, we consider this framework, and we define value iteration algorithms to get lexicographic optimal policies in the finite-horizon and infinite horizon cases.

This Chapter is organized as follows: Section 6.2 presents some definitions needed all along the chapter. Then, Section 6.3 is devoted to adaptations of value iteration algorithm for the lexicographic criteria in fixed horizon stationary $\Pi\mathcal{MDP}$s. Section 6.4 presents our experimental study of these algorithms. Finally, Section 6.5 extends the latter algorithm to the infinite-horizon case.

Principle results of this chapter are published in [Ben Amor et al., 2017].

6.2 Refresher on stationary possibilistic Markov decision processes

Before describing optimizing algorithms in detail, we recall some definitions from Chapter 2 and 3 on stationary $\Pi\mathcal{MDP}$ and the basic notions of lexicographic criteria in this framework.

- Stationary $\Pi\mathcal{MDP}$ is defined by:
  - A finite set S of states.
  - A finite set A of actions, A_s denotes the set of actions available in state s;
  - A utility function u s.t. $u(s)$ is the intermediate satisfaction degree obtained in state $s \in S$.
  - A transition function i.e. a possibility distribution on each action $a \in A_s$ s.t. $\pi(.|s,a)$

  In this model the states, the actions and the transition functions do not depend on the stage of the problem.

- Given a stationary $\Pi\mathcal{MDP}$ with horizon h, we can associate to any of its trajectories $\tau = (s_0, a_0, s_1, \ldots, s_{h-1},\ a_{h-1}, s_h)$ the vector $v_\tau = (u_0, \pi_1, u_1, \pi_2, \ldots, \pi_{h-1}, u_h)$.

  Different trajectories can be compared using lexicographic comparisons (Equation 3.3 and 3.3) as follows:

$$\tau \succeq_{lmin} \tau' \text{ iff } (u_0, \pi_1, u_1, \ldots, \pi_h, u_h) \succeq_{lmin} (u'_0, \pi'_1, u'_1, \ldots, \pi'_h, u'_h)$$

$$\tau \succeq_{lmax} \tau' \text{ iff } (u_0, 1-\pi_1, u_1, \ldots, 1-\pi_h, u_h) \succeq_{lmax} (u'_0, 1-\pi'_1, u'_1, \ldots 1-\pi'_h, u'_h)$$

  The lexicographic criteria on policies are then defined by:
  $\delta \succeq_{lmax(lmin)} \delta'$ iff $\forall i,\ \tau_{\lambda(i)} \sim_{lmin} \tau'_{u(i)}$

$$or\ \exists i^*,\ \forall i < i^*, \tau_{\lambda(i)} \sim_{lmin} \tau'_{\lambda(i)}\ \ and\ \ \tau_{\lambda(i^*)} \succ_{lmin} \tau'_{\lambda(i^*)}$$

  $\delta \succeq_{lmin(lmax)} \delta'$ iff $\forall i,\ \tau_{\sigma(i)} \sim_{lmax} \tau'_{\sigma(i)}$

$$or\ \exists i^*,\ \forall i < i^*, \tau_{\sigma(i)} \sim_{lmax} \tau'_{\sigma(i)}\ \ and\ \ \tau_{\sigma(i^*)} \succ_{lmax} \tau'_{\sigma(i^*)}$$

  where $\tau_{\lambda(i)}$ (resp. $\tau'_{\lambda(i)}$) is the i^{th} best trajectory of δ (resp. δ') according to $\succeq_{lmin}$ and $\tau_{\sigma(i)}$ (resp. $\tau'_{\sigma(i)}$) is the i^{th} worst trajectory of δ (resp. δ') according to $\succeq_{lmax}$.

- The complementary $\Pi\mathcal{MDP}$, of a given $\Pi\mathcal{MDP}$ (S, A, π, u), is defined by $(S, A, \pi, \bar{u})$ where $\bar{u}(s) = 1 - u(s), \forall s \in S$. It simply gives complementary utilities. From the definitions of $\succeq_{lmax}$ and $\succeq_{lmin}$ and using Proof of Proposition 5.4, we can check that:

$$\tau \succeq_{lmax} \tau' \Leftrightarrow \bar{\tau}' \succeq_{lmin} \bar{\tau} \text{ and } \delta \succeq_{lmin(lmax)} \delta' \Leftrightarrow \bar{\delta}' \succeq_{lmax(lmin)} \bar{\delta}$$

  where $\bar{\tau}$ and $\bar{\delta}$ are obtained by replacing u with $\bar{u}$ in the trajectory/$\Pi\mathcal{MDP}$.

  Therefore, all results which we will prove in the following for $\succeq_{lmax(lmin)}$ also hold for $\succeq_{lmin(lmax)}$, if we take care to apply them to complementary strategies. Since considering $\succeq_{lmax(lmin)}$ involves less cumbersome expressions (no $1 - \cdot$), we will give the results for this criterion.

6.3 Lexicographic value iteration for finite-horizon stationary $\Pi\mathcal{MDP}$s

We propose, in the following, a value iteration algorithm for the computation of lexicographic optimal policies in the finite-horizon stationary $\Pi\mathcal{MDP}$s.

6.3.1 Fixed-horizon lexicographic value iteration

It is possible propose a *Lexicographic Value Iteration Algorithm*, denoted $Lex\text{-}VI$, (Algorithm 6.1 for the $lmax(lmin)$ variant; the $lmin(lmax)$ variant is similar) that computes a lexicographic optimal policy in a finite number of iterations. This algorithm is an iterative procedure that updates the utility of each state, represented by a finite matrix of trajectories, using the utilities of the neighboring states, until a halting condition is reached.

At stage t, the procedure updates the utility of any states $s \in S$ as follows:

- For each $a \in A_s$, a matrix $Q(s, a)$, that evaluates the "utility" of performing a in s at stage t, is built by a call to obtained by calling $ConcatAndOrder\text{-}S(\vec{\pi}, \vec{u}, \{\rho^1, \ldots, \rho^k\})$, outlined by Algorithm 6.2. This function combines the possibility $\pi(s'|s, a)$ and the utilities $u(s')$, of the states s' that may follows s when a is executed, with the matrices $U^{t-1}(s')$ of trajectories provided by these s'. The obtained matrix is then ordered.
- The $lmax(lmin)$ comparison is performed on the fly to memorize the best $Q(s, a)$
- The value of s at t, $U^t(s)$, is the one given by the action $\delta^t(s) = a$ which provides the best $Q(s, a)$. U^t and δ^t are memorized (and U^{t-1} can be forgotten).

Algorithm 6.1: $Lex\text{-}VI$: lmax(lmin)-Value Iteration

Data: A finite-horizon stationary Π$\mathcal{MDP}$ and an horizon h
δ^*, the policy built by the algorithm, is a global variable
`// δ a global variable starts as an empty set`
Result: Computes and returns δ^* for Π$\mathcal{MDP}$
begin
  $t \leftarrow 0$;
  foreach $s \in S$ **do** $U^t(s) \leftarrow ((u(s)))$;
  foreach $s \in S$, $a \in A_s$ **do** $TU_{s,a} \leftarrow T_{s,a} \otimes ((u(s')), s' \in succ(s,a))$;
  repeat
    $t \leftarrow t + 1$;
    foreach $s \in S$ **do**
      $Q^* \leftarrow ((0))$;
      foreach $a \in A$ **do**
        $Future \leftarrow \{U^{t-1}(s'), s' \in succ(s,a)\}$; `// Gather the matrices provided by the successors of s;`
        $Q(s,a) \leftarrow ConcatAndOrder\text{-}S(\vec{\pi}, \vec{u}, Future)$;
        if $Q^* \leq_{lmaxlmin} Q(s,a)$ **then**
          $Q^* \leftarrow Q(s,a)$;
          $\delta^t(s) \leftarrow a$
      $U^t(s) \leftarrow Q^*$
  until $t == h$;
  $\delta^*(s) \leftarrow argmax_a Q(s,a)$
  return δ^*;

Time and space complexities of this algorithm are nevertheless expensive, since it memorizes all the trajectories. At each step t its size may be about $b^t \cdot (2 \cdot t + 1)$, where b is the maximal number of possible successors of an action. The overall complexity of the algorithm is $O(|S| \cdot |A| \cdot |h| \cdot b^h \cdot log(b^h))$, since:

- The number of iterations is bounded by the size of the set of possible matrices of trajectories i.e. $|S| \cdot |A| \cdot |h|$,
- One iteration of the algorithm requires composition, ordering and comparing operations on b matrices. Since the composition and comparison of matrices are linear operations, the complexity of one iteration in worst case (when considering the bigger matrices) is in $O(b \cdot b^h \cdot (2 \cdot h + 1) \cdot log(b^h \cdot (2 \cdot h + 1))) = O(b^h \cdot log(b^h))$.

Algorithm 6.2: ConcatAndOrder-S($\vec{\pi}$,$\vec{u}$, $\{\rho^1, \ldots, \rho^k\}$)

Data: k matrices $\rho^1, \ldots, \rho^k$, a distribution π on $\{1, \ldots, k\}$, a set of k utilities
Result: ρ, the combination of $\rho^1, \ldots, \rho^k$ according to π and u

```
// Notations:
// L_ρ:  number of lines of ρ,
// C_ρ:  number of columns of ρ,
// ρ_(z):  the line z in ρ,
// ρ_{z,t}:  the element in line z and column t in ρ
begin
    NbLines ← Σ_{m=1}^{k} L_{ρ^m};
    max_C ← max_{m=1,k}(C_{ρ^m});
    Creates a matrix ρ with NbLines lines and max_C + 1 columns
    // Concatenation
    z ← 0;
    for m = 1, k do
        for z' = 1, L_{ρ^m} do
            z ← z + 1;
            for t = 1, C_{ρ^m} do ρ_{z,t} ← ρ^m_{z',t};
            for t = C_{ρ^m} + 1, max_C do ρ_{z,t} ← 0;
            ρ_{z,max_C+1} ← π(m);
            ρ_{z,max_C+2} ← u(m);
    // Ordering the elements of each line by increasing
       order
    for z = 1, NbLines do
        sortIncreasing(ρ_(z), ≥);
    // Ordering the lines by decreasing order according
       to lmax
    sortDecreasing(ρ, ≥_{lmax});
    return ρ;
```

Example 6.1. *The main steps for the evaluation of the finite-horizon stationary* $\Pi\mathcal{MDP}$ *(see Figure 6.1) of Counter-example 3.4 using lexicographic value iteration (Algorithm 6.1) w.r.t.* $lmax(lmin)$ *criterion are as follows.*

- *We have:*
  - $S = \{R\&U,\ R\&F, P\&U\}$,
  - $A = \{Adv, Sav, Stay\}$ *and*
  - $u(R\&U) = 0.5,\ \ u(R\&F) = 0.7,\ \ u(P\&U) = 0.3$.

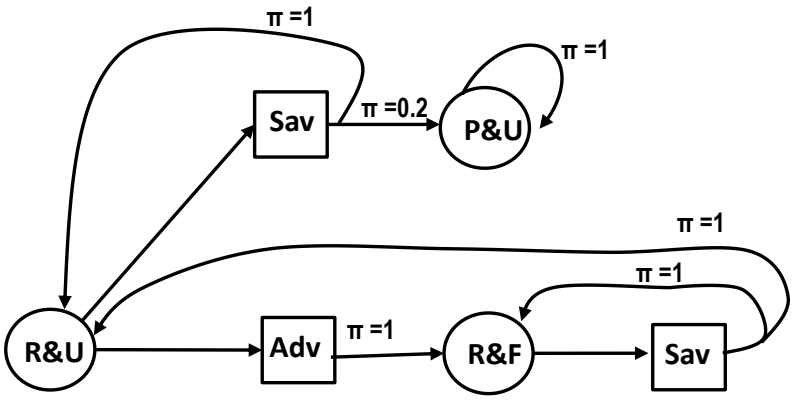


Figure 6.1: The stationary finite-horizon $\pi\mathcal{MDP}$ of Example 6.1

Let us execute the algorithm for $h = 3$ iterations.

- $h = 0$*: initialization:*
  $U^0(R\&U) = [0.5]$, $U^0(R\&F) = [0.7]$, $U^0(P\&U) = [0.3]$.

- $h = 1$*:*
  for $s = R\&U$

  - *for* $a = Sav$*:*
    Future = $\begin{bmatrix} 0.3 \\ 0.5 \end{bmatrix}$,
    $Q(R\&U, Sav) = ConcatAndOrder\text{-}S(\vec{\pi} = (0.2, 1), \vec{u} = (0.3, 0.5), \begin{bmatrix} 0.3 \end{bmatrix}, \begin{bmatrix} 0.5 \end{bmatrix})$
    $= \begin{bmatrix} 0.5 & 0.5 & 1 \\ 0.2 & 0.3 & 0.3 \end{bmatrix}$.
  - *for* $a = Adv$*:*
    Future = $\begin{bmatrix} 0.7 \end{bmatrix}$,
    $Q(R\&U, Adv) = ConcatAndOrder\text{-}S(\vec{\pi} = (1), \vec{u} = (0.7), \begin{bmatrix} 0.7 \end{bmatrix})$
    $= \begin{bmatrix} 0.7 & 0.7 & 1 \end{bmatrix}$.

  $\rightarrow$ *Since* $Q(R\&U, Adv) \succ_{lmax(lmin)} Q(R\&U, Sav)$*, so* $\delta^1(R\&U) = Adv$
  and $U^1(R\&U) = \begin{bmatrix} 0.7 & 0.7 & 1 \end{bmatrix}$.

  * *for* $s = R\&F$

  - *for* $a = Sav$*:*
    Future = $\begin{bmatrix} 0.7 \\ 0.5 \end{bmatrix}$,

$Q(R\&F, Sav) = ConcatAndOrder\text{-}S(\vec{\pi} = (1,1), \vec{u} = (0.7, 0.5), \begin{bmatrix}0.7\end{bmatrix}, \begin{bmatrix}0.5\end{bmatrix})$
$= \begin{bmatrix}0.7 & 0.7 & 1\\ 0.5 & 0.5 & 1\end{bmatrix}$.

$\rightarrow$ *Since* $Q(R\&F, Sav) \succ_{lmax(lmin)} [0]$, *so* $\delta^1(R\&F) = Sav$
and $U^1(R\&F) = \begin{bmatrix}0.7 & 0.7 & 1\\ 0.5 & 0.5 & 1\end{bmatrix}$.

* *for* $s = P\&U$

- *for* $a = Stay$:
  Future = $\begin{bmatrix}0.3\end{bmatrix}$,
  $Q(P\&U, Stay) = ConcatAndOrder\text{-}S(\vec{\pi} = (1), \vec{u} = (0.3), \begin{bmatrix}0.3\end{bmatrix})$
  $= \begin{bmatrix}0.3 & 0.3 & 1\end{bmatrix}$.

$\rightarrow$ *Since* $Q(P\&U, Stay) \succ_{lmax(lmin)} [0]$, *so* $\delta^1(P\&U) = Stay$
and $U^1(P\&U) = \begin{bmatrix}0.3 & 0.3 & 1\end{bmatrix}$.

- $h = 2$:
  for $s = R\&U$

  - *for* $a = Sav$:
    Future = $\begin{bmatrix}\begin{bmatrix}0.3 & 0.3 & 1\end{bmatrix}\\ \begin{bmatrix}0.5 & 0.5 & 1\end{bmatrix}\\ \begin{bmatrix}0.2 & 0.3 & 0.3\end{bmatrix}\end{bmatrix}$,
    $Q(R\&U, Sav) = ConcatAndOrder\text{-}S(\vec{\pi} = (0.2, 1), \vec{u} = (0.3, 0.5), \begin{bmatrix}0.3 & 0.3 & 1\end{bmatrix}, \begin{bmatrix}0.5 & 0.5 & 1\\ 0.2 & 0.3 & 0.3\end{bmatrix}) = \begin{bmatrix}0.5 & 0.5 & 0.5 & 1 & 1\\ 0.2 & 0.3 & 0.3 & 0.5 & 1\\ 0.2 & 0.3 & 0.3 & 0.3 & 1\end{bmatrix}$.
  - *for* $a = Adv$:
    Future = $\begin{bmatrix}0.7 & 0.7 & 1\\ 0.5 & 0.5 & 1\end{bmatrix}$,
    $Q(R\&U, Adv) = ConcatAndOrder\text{-}S(\vec{\pi} = (1), \vec{u} = (0.7), \begin{bmatrix}0.7 & 0.7 & 1\\ 0.5 & 0.5 & 1\end{bmatrix})$
    $= \begin{bmatrix}0.7 & 0.7 & 1 & 1\\ 0.5 & 0.5 & 0.7 & 1\end{bmatrix}$.

  $\rightarrow$ *Since* $Q(R\&U, Adv) \succ_{lmax(lmin)} Q(R\&U, Sav)$, *so* $\delta^2(R\&U) = Adv$
  and $U^2(R\&U) = \begin{bmatrix}0.7 & 0.7 & 0.7 & 1 & 1\\ 0.5 & 0.5 & 0.7 & 1 & 1\end{bmatrix}$.

  * *for* $s = R\&F$

– *for* $a = Sav$:

Future = $\begin{bmatrix} \begin{bmatrix} 0.7 & 0.7 & 0.7 & 1 \\ 0.5 & 0.5 & 1 & 1 \end{bmatrix} \\ \begin{bmatrix} 0.7 & 0.7 & 1 \end{bmatrix} \end{bmatrix}$,

$Q(R\&F, Sav) = ConcatAndOrder\text{-}S(\vec{\pi} = (1,1), \vec{u} = (0.7, 0.5), \begin{bmatrix} 0.7 & 0.7 & 0.7 & 1 \\ 0.5 & 0.5 & 1 & 1 \end{bmatrix}, \begin{bmatrix} 0.7 & 0.7 & 1 \end{bmatrix}) = \begin{bmatrix} 0.7 & 0.7 & 0.7 & 1 & 1 \\ 0.5 & 0.5 & 0.7 & 1 & 1 \\ 0.5 & 0.5 & 0.7 & 1 & 1 \end{bmatrix}$.

$\rightarrow$ *Since* $Q(R\&F, Sav) \succ_{lmax(lmin)} [0]$, *so* $\delta^2(R\&F) = Sav$
and $U^2(R\&F) = \begin{bmatrix} 0.7 & 0.7 & 0.7 & 1 & 1 \\ 0.5 & 0.5 & 0.7 & 1 & 1 \\ 0.5 & 0.5 & 0.7 & 1 & 1 \end{bmatrix}$.

* *for* $s = P\&U$

– *for* $a = Stay$:

Future = $\begin{bmatrix} 0.3 & 0.3 & 1 \end{bmatrix}$,

$Q(P\&U, Stay) = ConcatAndOrder\text{-}S(\vec{\pi} = (1), \vec{u} = (0.3), \begin{bmatrix} 0.3 & 0.3 & 1 \end{bmatrix})$
$= \begin{bmatrix} 0.3 & 0.3 & 0.3 & 1 & 1 \end{bmatrix}$.

$\rightarrow$ *Since* $Q(P\&U, Stay) \succ_{lmax(lmin)} [0]$, *so* $\delta^2(P\&U) = Stay$
and $U^2(P\&U) = \begin{bmatrix} 0.3 & 0.3 & 0.3 & 1 & 1 \end{bmatrix}$.

$\Longrightarrow$ *The stationary policy optimal w.r.t.* $lmax(lmin)$ *is:*
$\delta(R\&U) = Adv; \delta(R\&F) = Sav; \delta(P\&U) = Stay$.

6.3.2 Bounded lexicographic value iteration

We have seen that making the choices based on the qualitative utility functionals is not discriminant enough. Note that, at any stage t and for any state s, $[U^t(s)]_{1,1}$ (i.e. the top left value in $U^t(s)$) is precisely equal to $u_{opt}(s)$ at horizon t for the optimal policy. on the other hand, taking the whole matrix is discriminant, but exponentially costly ($O(|S| \cdot |A| \cdot |h| \cdot b^h \cdot log(b^h))$. Hence we propose to consider more than one line and one column, but less than the whole matrix -namely the first l lines and c columns of the ordered matrix $U^t(s)$; hence the definition of the following preference:

$$\delta \geq_{lmaxlmin,l,c} \delta' \text{ iff } [\rho_\delta]_{l,c} \geq [\rho_{\delta'}]_{l,c} \tag{6.1}$$

$\geq_{lmaxlmin,1,1}$ corresponds to $\succeq_{opt}$ and $\geq_{lmaxlmin,+\infty,+\infty}$ corresponds to $\geq_{lmaxlmin}$.

The combinatorial explosion in Algorithm 6.1 is due to the number of lines only, because at finite horizon, the number of columns is bounded by $2 \cdot h + 1$. Hence we should bound the number of considered lines. The following proposition shows that this approach is sound:

Proposition 6.1. *(Refinement relations)*

- *For any l, c, l' such that $l' > l$, $\delta \succ_{lmaxlmin,l,c} \delta' \Rightarrow \delta \succ_{lmaxlmin,l',c} \delta'$.*
- *For any l, c, $\delta \succ_{opt} \delta' \Rightarrow \delta \succ_{lmaxlmin,l,c} \delta'$.*

Proposition 6.1 means that $\succ_{lmaxlmin,l,c}$ refines u_{opt} and the order over the policies is refined for a fixed c when l increases. It tends to $\succ_{lmaxlmin}$ when $c = 2.h + 1$ and l tends to b^h.

Proof of Proposition 6.1.
In order to make the proofs more explicit and compact, let us introduce the following notations and operations defined on matrices of trajectories (typically, on $U(s)$ representing trajectories issued from s).

- **Composition operation** denoted $\otimes$:
  $U \otimes (N_1, \ldots, N_a)$: Let U be a $a \otimes b$ matrix and $N_1, \ldots, N_a$ be a series of a matrices of dimension $n_i \otimes c$ (they all share the same number of columns). The composition of U with $(N_1, \ldots, N_a)$ denoted $U \otimes (N_1, \ldots, N_a)$ is a matrix of dimension $(\sum_{1 \leq i \leq a} n_i) \otimes (b + c)$. For any $i \leq a, j \leq n_j$, the $(\Sigma_{i'<i} n_{i'}) + j)^{th}$ line of $U \otimes (N_1, \ldots, N_a)$ is the concatenation of the i^{th} line of U and the j^{th} line of N_i. The matrix $U(s)$ is typically the concatenation of the matrix $U = ((\pi(s'|s, a), u(s')), s' \in succ(s, a))$ with the matrices $N_{s'} = U(s')$.
- **Combination operation** denoted $\oplus$:
  $U \oplus V$: Let U be a $a \otimes b$ matrix and V be a $n \otimes b$ matrix. The combination of U with V is the matrix $W = U \oplus V$ of dimension $(a + n) \times b$ s.t. $\forall\, i \in \{1..(a + n)\}$, $\forall\, j \in \{1..b\}$:

$$W_{i,j} = \begin{cases} U_{i,j} & \text{if } i \in \{1..a\}, \\ V_{(i-a),j} & \text{if } i \in \{(a+1)..(a+n)\}. \end{cases}$$

- **Ordering operation**: Let U be a $n \times m$ matrix, $U^{lmaxlmin}$ is the matrix obtained by the operation $lmax(lmin)$: ordering the elements of the lines of U in increasing order and the lines of U according to $lmax$ (in decreasing order).

ConcatAndOrder−S function is simply defined as follows:

$$\begin{aligned} ConcatAndOrder\text{-}S(\vec{\pi}, \vec{u}, \{\rho^1, \ldots, \rho^k\}) &= (\pi, u) \otimes (\rho^1, \ldots, \rho^k) \\ &= (((\pi_1, u_1) \otimes \rho^1) \oplus \ldots \oplus ((\pi_k, u_k) \otimes \rho^k)). \end{aligned}$$

Let us now give the Proof of Proposition 6.1:

- Note that, for any t, s, we have:

  $\left[U^t(s)^{lmaxlmin}\right]_{l,c}$ has the form $\left[\left(\begin{array}{c|c} \multicolumn{2}{c}{\ldots} \\ \hline \pi(s'_i|s,a), u(s'_i) & \\ \ldots & U^{t-1}(s'_i)^{lmaxlmin} \\ \pi(s'_i|s,a), u(s'_i) & \\ \hline \multicolumn{2}{c}{\ldots} \end{array}\right)^{lmaxlmin}\right]_{l,c}$

  Formally, $\left[U^t(s)^{lmaxlmin}\right]_{l,c} = [((\pi(s'_1|s,a), u(s'_1))\otimes U^{t-1}(s'_1)^{lmaxlmin})\oplus((\pi(s'_2|s,a), u(s'_2))\otimes U^{t-1}(s'_2)^{lmaxlmin})\oplus\ldots\oplus((\pi(s'_k|s,a), u(s'_k))\otimes U^{t-1}(s'_k)^{lmaxlmin})]$,

  Let A be a $n \times m$ matrix, $A^{lmaxlmin}_{(i,x:y)}$ denote the part of the line i, of A, having $y - x$ elements from column x to y s.t. $x < y \leq m$

  Now, note that, if A and B are two matrices with exactly c columns: $\left[A^{lmaxlmin}\right]_{l,c} >_{lmaxlmin} \left[B^{lmaxlmin}\right]_{l,c}$ if and only if $\exists i^* \leq l$, such that $\forall i < i^*$, $A^{lmaxlmin}_{(i)} =_{lmin} B^{lmaxlmin}_{(i)}$ and $A^{lmaxlmin}_{(i^*)} >_{lmin} B^{lmaxlmin}_{(i^*)}$.

  Clearly, in this case replacing A and B with $U^t(s)^{lmaxlmin}$ when considering δ and $U'^t(s)^{lmaxlmin}$ when considering δ', if such a $i^* \leq l$ exists for a given l, the same i^* works for $l' > l$.

  Thus, $\succ_{lmaxlmin,l',c}$ refines $\succ_{lmaxlmin,l,c}$.

  Remark that the property does not hold for c. Increasing c does not refine the order $\succ_{lmaxlmin,l,c,t,s}$. Indeed, given $c < c'$, we can find a pair of matrices for which it holds all together that:

  - $A^{lmaxlmin}_{(1,1:c)} =_{lmin} B^{lmaxlmin}_{(1,1:c)}$,
  - $A^{lmaxlmin}_{(2,1:c)} >_{lmin} B^{lmaxlmin}_{(2,1:c)}$ and
  - $A^{lmaxlmin}_{(1,1:c')} <_{lmin} B^{lmaxlmin}_{(i^*,1:c')}$.

  Thus, $A^{lmaxlmin} \succ_{lmaxlmin,l=2,c} B^{lmaxlmin}$ and $B^{lmaxlmin} \succ_{lmaxlmin,l=2,c'} A^{lmaxlmin}$. One can easily build a decision problem and two policies corresponding to matrices A and B satisfying the above. Thus, increasing c does not lead to a more refined order.

- From the first point, above, it holds that

  $\succ_{lmaxlmin,l=1,c}$ refines $\succ_{lmaxlmin,l=1,c=1}$ and that $\succ_{lmaxlmin,l,c}$ refines $\succ_{lmaxlmin,l=1,c}$.

  So, $\succ_{lmaxlmin,l,c}$ refines $\succ_{lmaxlmin,l=1,c=1}$, which is equivalent to the order induced by u_{opt}. Thus, optimal solutions of $\succ_{lmaxlmin,l,c}$ are also optimal for u_{opt}, in all steps of the stationary $\Pi\mathcal{MDP}$.

□

Up to this point, the comparison by $\geq_{lmaxlmin,l,c}$ is made on the basis of the first l lines and c columns of the *full* matrices of trajectories. This does obviously not reduce their size. The important following Proposition allows us to make the l, c reduction of the ordered matrices *at each step* (after each composition), and not only at the very end, thus keeping space and time complexities polynomial.

Proposition 6.2. *(Concatenation of reduced matrices of trajectories)*
Let U be a $a \times b$ matrix and $N_1, \ldots, N_a$ be a series of a matrices of dimension $a_i \times c$. It holds that:
$[(U \otimes (N_1, \ldots, N_a))^{lmaxlmin}]_{l,c} = [(U \otimes ([N_1^{lmaxlmin}]_{l,c}, \ldots, [N_a^{lmaxlmin}]_{l,c}))^{lmaxlmin)}]_{l,c}$.

Proof of Proposition 6.2.
Note that

$$\Big[(U \otimes (N_1, \ldots, N_a))^{lmaxlmin}\Big]_{l,c} = [(U_1 \otimes N_1) \oplus (U_2 \otimes N_2) \oplus ... \oplus (U_k \otimes N_k)]^{lmaxlmin},$$

Now, note the two following facts:

Fact 1: $(A \oplus B)^{lmaxlmin} = \Big((A)^{lmaxlmin} \oplus (B)^{lmaxlmin}\Big)^{lmaxlmin}$, The reason is that: $A^{lmaxlmin}$ first reorders each line in $lmin$ order, which can be done independently for each line and then all ordered lines are ordered through $lmax$. This second step can be done separately for each submatrix, provided that the lines are leximax-reordered once more, which is done by the external $lmax(lmin)$ operator.

Fact 2: $\Big(U_{(i)} \otimes A\Big)^{lmaxlmin} = \Big(U_{(i)} \otimes (A)^{lmaxlmin}\Big)^{lmaxlmin}$,

This second fact holds since adding identical elements to each line of a matrix does not modify the leximin ordering of the lines. In the right hand term of the equality, the outer $lmaxlmin$ operator only inserts the terms of $U_{(i)}$ in all lines of matrix $A^{lmaxlmin}$.

Now, from Fact 1, we get:
$$\begin{aligned}(U \otimes (N_1, \ldots, N_a))^{lmaxlmin} = [&(U_1 \otimes N_1)^{lmaxlmin} \oplus (U_2 \otimes N_2)^{lmaxlmin} \oplus ... \oplus \\ &(U_k \otimes N_k)^{lmaxlmin}]^{lmaxlmin},\end{aligned}$$

And then, from Fact 2:
$$\begin{aligned}(U \otimes (N_1, \ldots, N_a))^{lmaxlmin} = [&(U_1 \otimes (N_1)^{lmaxlmin})^{lmaxlmin} \oplus (U_2 \otimes (N_2)^{lmaxlmin})^{lmaxlmin} \\ &\oplus ... \oplus (U_k \otimes (N_k)^{lmaxlmin})^{lmaxlmin}]^{lmaxlmin},\end{aligned}$$

Now, using Fact 1 again, in the other direction of the equality:

$$\begin{aligned}(U \otimes (N_1, \ldots, N_a))^{lmaxlmin} &= [(U_1 \otimes (N_1)^{lmaxlmin})^{lmaxlmin} \oplus (U_2 \otimes (N_2)^{lmaxlmin})^{lmaxlmin} \\ &\quad \oplus ... \oplus (U_k \otimes (N_k)^{lmaxlmin})^{lmaxlmin}]^{lmaxlmin} \\ &= \Big(U \otimes ((N_1)^{lmaxlmin}, \ldots, (N_a)^{lmaxlmin})\Big)^{lmaxlmin} \qquad (6.2)\end{aligned}$$

From (6.2), we have, of course,

$$\Big[(U \otimes (N_1, \ldots, N_a))^{lmaxlmin}\Big]_{l,c} = \Big[\Big(U \otimes ((N_1)^{lmaxlmin}, \ldots, (N_a)^{lmaxlmin})\Big)^{lmaxlmin}\Big]_{l,c}.$$

$$\Big[(U \times (N_1, \ldots, N_a))^{lmaxlmin}\Big]_{l,c} = \Big[\Big(U \times (N_1^{lmaxlmin}, \ldots, N_a^{lmaxlmin})\Big)^{lmaxlmin}\Big]_{l,c}.$$

Now, noticing that:

- The $N_i^{lmaxlmin}$ are lmax(min)-ordered, as well as the lines $\Big(U_{(i)}\Big)$,
- the $N_i^{lmaxlmin}$ have exactly c columns and
- since the resulting (reordered) matrix has l lines, it cannot contain more than l lines of any of the $N_i^{lmaxlmin}$ matrices.

We can safely replace the inner $N_i^{lmaxlmin}$ matrices with their sub-matrices $\Big[N_i^{lmaxlmin}\Big]_{l,c}$ and get the result.

□

In summary, the idea of our Algorithm, that we call *Bounded Lexicographic Value Iteration* $(BLex\text{-}VI)$, see Algorithm 6.3, is to compute policies that are close to lexi-optimality, by keeping a sub matrix of each current value matrix - namely the first l lines and c columns.

Following the complexity analysis of Algorithm 6.1, the time complexity of $(BLex\text{-}VI)$ is: $O(|S| \cdot |A| \cdot |h| \cdot b \cdot (l \cdot c) \cdot log(l \cdot c))$. Hence, this algorithm provides in in polynomial time a policy that is always as least as good as the one provided by u_{opt} (according to $lmax(lmin)$) and tends to lexicographic optimality when $c = 2 \cdot h + 1$ and l tends to b^h.

Algorithm 6.3: $BLex\text{-}VI$: Bounded-lmax(lmin)-Value Iteration

Data: A stationary $\Pi\mathcal{MDP}$ and an horizon h
δ^*, the policy built by the algorithm, is a global variable
// δ a global variable starts as an empty set
Result: Computes and returns δ^* for $\Pi\mathcal{MDP}$
begin
  $t \leftarrow 0$;
  foreach $s \in S$ **do** $U^t(s) \leftarrow ((u(s)))$;
  foreach $s \in S$, $a \in A_s$ **do** $TU_{s,a} \leftarrow T_{s,a} \otimes ((u(s')), s' \in succ(s,a))$;
  repeat
    $t \leftarrow t + 1$;
    foreach $s \in S$ **do**
      $Q^* \leftarrow ((0))$;
      foreach $a \in A$ **do**
        $Future \leftarrow (U^{t-1}(s'), s' \in succ(s,a))$; // Gather the matrices provided by the successors of s;
        $Q(s,a) \leftarrow [ConcatAndOrder\text{-}S(\vec{\pi}, \vec{u}, Future)]_{l,c}$;
        if $Q^* \leq_{lmaxlmin} Q(s,a)$ **then**
          $Q^* \leftarrow Q(s,a)$;
          $\delta^t(s) \leftarrow a$
      $U^t(s) \leftarrow Q^*$
  until $t == h$;
  $\delta^*(s) \leftarrow argmax_a Q(s,a)$
  return δ^*;

6.4 Experimental study

In this Section, we compare the performance of *Bounded lexicographic value iteration* so-called $BLex\text{-}VI$ (Algorithm 6.3) as an approximation of (unbounded) *lexicographic value iteration* so-called $Lex\text{-}VI$ (Algorithm 6.1), in the $lmax(lmin)$ variant.

We evaluate the performance of the algorithms by carrying out simulations on randomly generated stationary $\Pi\mathcal{MDP}$ with $|S| = 25$. The number of actions in each state is equal to 4. The output of each action is a distribution on two states randomly drawn (i.e. the branching factor is equal to 2). The utility values are uniformly randomly drawn in the set $L = \{0.1, 0.3, 0.5, 0.7, 1\}$. Conditional possibilities relative to decisions should be normalized. To this end, one choice is fixed to possibility degree 1 and the possibility degree of the other one is uniformly drawn in L. For each experience, 100 stationary $\Pi\mathcal{MDP}$ are generated.

The two algorithms are compared w.r.t. 2 measures:
(i) CPU time and
(ii) Pairwise success rate (the same of the previous experiments): here it presents the percentage of optimal solutions provided by Bounded value iteration with fixed (l, c) w.r.t. the lmax(lmin) criterion in its full generality. The higher $Success$, the more important the effectiveness of cutting matrices with $BLex\text{-}VI$; the lower this rate, the more important the drowning effect.

Figure 6.2 presents the average execution CPU time for the two algorithms. Obviously, for both $Lex\text{-}VI$ and $BLex\text{-}VI$, the execution time increases with the horizon. Also, we observe that the CPU time of $BLex\text{-}VI$ increases according to the values of (l, c) but it remains affordable, as the maximal CPU time is lower than $1s$ for stationary $\Pi\mathcal{MDP}$s with 25 states and 4 actions when $(l, c) = (40, 40)$ and $h = 25$. Unsurprisingly, we can check that the $BLex\text{-}VI$ (regardless of the values of (l, c)) is faster than $Lex\text{-}VI$ especially when the horizon increases: the manipulation of l, c-matrices is obviously less expensive than the one of full matrices. The saving increases with the horizon.

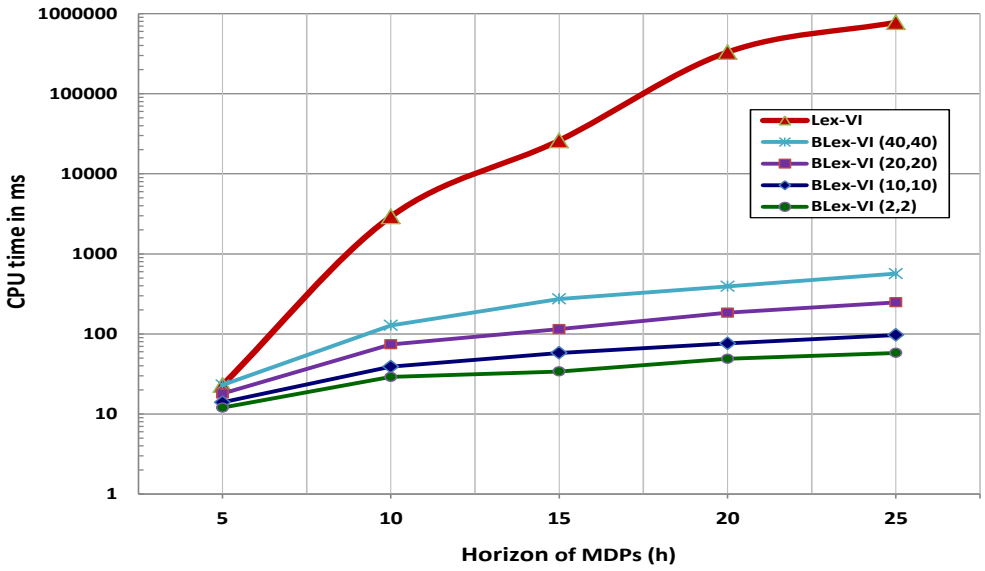


Figure 6.2: Average CPU-time for optimistic criteria in stationary $\Pi\mathcal{MDP}$s

As with the success rate, the results are described in Figure 6.3. It appears that $BLex\text{-}VI$ provides a very good approximation especially when increasing (l, c). It provides the same optimal solution as the $Lex\text{-}VI$ in about 90% of cases, with an $(l, c) = (200, 200)$. So, the bigger the matrices the more efficient is the approximation, and the smaller the matrices the further the drowning effect is present. Moreover, even when the success rate of $BLex\text{-}VI$ decreases (when h increases), the quality of approximation is still good: never less than 70% of optimal actions returned, with $h = 25$.

These experiments conclude in favor of *bounded lexicographic value iteration*: its approximated solutions are comparable in terms of quality for high (l, c) and increase when (l, c) increase, while it is much faster than the unbounded version.

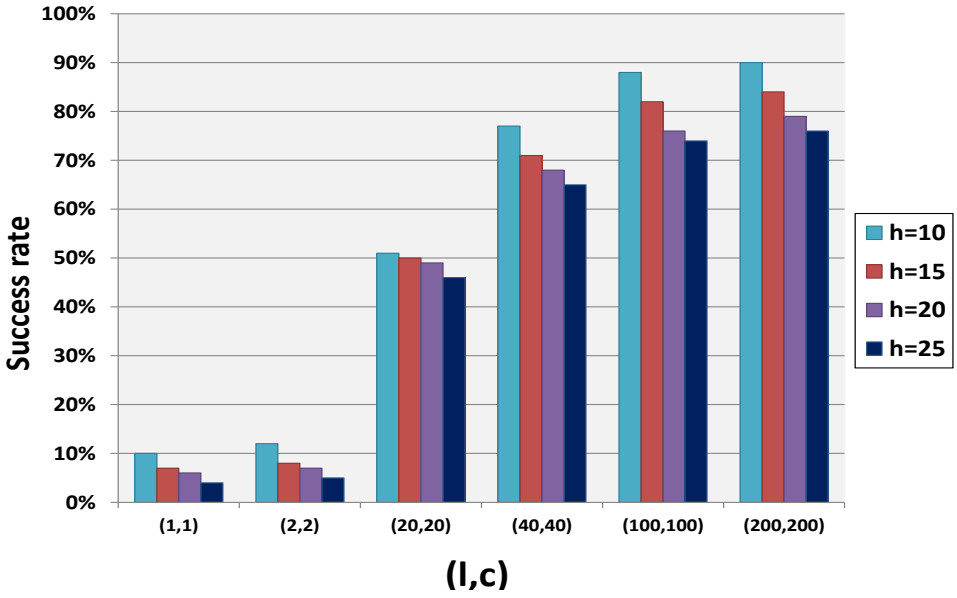


Figure 6.3: Success rate for optimistic criteria in stationary $\Pi\mathcal{MDP}$s

6.5 Lexicogaphic value iteration for infinite-horizon stationary $\Pi\mathcal{MDP}$s

In the infinite-horizon case, the comparison of matrices of trajectories does not apply to rank-order the policies. The length of the trajectories may be infinite, and also their number infinite as well.

This problem is well known in classical probabilistic $\mathcal{MDP}$s where a discount factor is used that attenuates the influence of later utility degrees - thus allowing the convergence of the solving algorithm (value iteration or policy iteration) [Puterman, 1994]. On the contrary, classical $\Pi\mathcal{MDP}$s, proposed by [Sabbadin, 2001], do not need any discount factor. Indeed, the limitation of u_{opt} and u_{pes} to one number as a utility of a policy (i.e. $l = c = 1$) plays the role of a discount factor and possibilistic Value Iteration (see Algorithm 2.3), based on the evaluation of qualitative utilities, converges for infinite horizon $\Pi\mathcal{MDP}$s [Sabbadin, 2001]. However, decide using u_{opt} and u_{pes} (based on the first element of the matrix) is too drastic; it is nevertheless possible to make the comparison using $\geq_{lmaxlmin,l,c}$ with $l, c > 1$. Thus, we propose below a *Bounded Lexicographic Value Iteration algorithm* for infinite-horizon stationary $\Pi\mathcal{MDP}$s, denoted $BLex$-IH-VI (see Algorithm 6.4). This algorithm converges when from a given stage t, the value of

a policy is stable if computed with the bounded $lmax(lmin)$ criterion. Let us denote $U^t(s)$ the matrix issued from s at horizon t when δ is executed, it holds that:

Proposition 6.3. *(Convergence of Bounded lexicographic value iteration)*
$\forall\, l, c, \exists\, t$ *such that,* $\forall\, t' \geq t$, $(U^t)^{lmaxlmin}_{l,c}(s) = (U^{t'})^{lmaxlmin}_{l,c}(s)\ \ \forall s$. *Hence, bounded lexicographic value iteration converges for infinite-horizon stationary* $\Pi\mathcal{MDP}$*s.*

Proof of Proposition 6.3.
In, bounded lexicographic value iteration, at each time step, $U^t(s)$ is composed of a set of trajectories $\tau^i_t = \langle s^i_0, a^i_0, s^i_1, \ldots, s^i_t, a^i_t, s^i_{t+1}\rangle$, which can be identified with the set of possibilities/utilities of each transition $(s^i_{t'}, a^i_{t'}, s'^i_{t'+1})$ s.t. $t \geq t' \geq 0$, these are obtained from $\{\langle \pi^i_{t'}, u^i_{t'}\rangle\}_{t'=0,t}$. Thus, τ^i_t has $2t$ elements. Let $v^{i,\alpha}_t$ counts the number of times a level $\alpha \in V$ has been obtained by $\pi^i_{t'}$ or $u^i_{t'}$ during the trajectory τ^i_t. Statistics $< v^{i,1}_t, \ldots, v^{i,|V|}_t >$ can be maintained for every trajectory.

As t increases all $(v^{i,\alpha}_t)$, which are non-decreasing sequences, converge toward a finite or infinite limit. We let

$$\lim(i,\alpha) =_{def} \lim_{t\to\infty}(v^{i,\alpha}_t),\ s.t.\ lim(i,\alpha) \in \mathcal{N} \cup \{+\infty\},\ \forall\, \tau^i_t, \alpha.$$

Thus, if we denote by U_i, the limit of the line vector in $U^t(s)$ corresponding to trajectory τ^i_t when t goes to infinity, we have:

$$U_i = \left[\underbrace{\alpha_1, \ldots, \alpha_1}_{\lim(i,1)}, \underbrace{\alpha_2, \ldots, \alpha_2}_{\lim(i,2)}, \ldots\right].$$

Let us now consider $U^*_{l,c}$, the $l \times c$ matrix made from the l first (in leximax order) line vectors $[U_i]_{1,c}$. Then, obviously,

$$\lim_{t\to\infty}\left[U^t(s)^{lmaxlmin}\right]_{l,c} = U^*_{l,c}.$$

Thus, bounded lexicographic value iteration algorithm converges.

Besides, we show now that if $[U^t(s)^{lmaxlmin}]_{l,c} = [U^{t-1}(s)^{lmaxlmin}]_{l,c}$, thus, $\forall\, , t' \geq t$, we have $[U^{t'}(s)^{lmaxlmin}]_{l,c} = [U^t(s)^{lmaxlmin}]_{l,c}$:

let us consider the following hypothesis:

$$H:\ [U^t(s)^{lmaxlmin}]_{l,c} = [U^{t-1}(s)^{lmaxlmin}]_{l,c}.$$

Considering H, let us calculate $[U^{t+1}(s)^{lmaxlmin}]_{l,c}$:
$Future^{t+1} = (U^t(s'), s' \in succ(s,a)) = Future^t$,
$\rightarrow\ Q^{t+1}(s,a) = [(TU_{s,a} \otimes Future^t)^{lmaxlmin}]_{l,c} = [(TU_{s,a} \otimes Future^{t-1})^{lmaxlmin}]_{l,c} = Q^t(s,a)$.
We deduce that, $(U^{t+1})^{lmaxlmin}_{l,c} = (U^{t-1})^{lmaxlmin}_{l,c}$. □

The overall complexity of *Bounded lmax(lmin)-Value Iteration* algorithm is $O(|V| \cdot (l \cdot c) \cdot |S| \cdot |A| \cdot (l \cdot c) \cdot b \cdot log(l \cdot c)) = O(|V| \cdot (l \cdot c)^2 \cdot |S| \cdot |A| \cdot b \cdot log(l \cdot c))$.

Algorithm 6.4: $BLex\text{-}IH\text{-}VI$: Bounded-infinite-horizon-lmax(lmin)-Value Iteration

Data: A stationary $\Pi\mathcal{MDP}$ and an horizon h
δ^*, the policy built by the algorithm, is a global variable
1 `// δ a global variable starts as an empty set`
Result: Computes and returns δ^* for $\Pi\mathcal{MDP}$
2 **begin**
3 $t \leftarrow 0$;
4 **foreach** $s \in S$ **do** $U^t(s) \leftarrow ((u(s)))$;
5 **foreach** $s \in S$, $a \in A_s$ **do** $TU_{s,a} \leftarrow T_{s,a} \otimes ((u(s')), s' \in succ(s,a))$;
6 **repeat**
7 $t \leftarrow t + 1$;
8 **foreach** $s \in S$ **do**
9 $Q^* \leftarrow ((0))$;
10 **foreach** $a \in A$ **do**
11 $Future \leftarrow (U^{t-1}(s'), s' \in succ(s,a))$; `// Gather the matrices provided by the successors of s;`
12 $Q(s,a) \leftarrow [(TU_{s,a} \otimes Future)^{lmaxlmin}]_{l,c}$;
13 **if** $Q^* \leq_{lmaxlmin} Q(s,a)$ **then**
14 $Q^* \leftarrow Q(s,a)$;
15 $\delta^t(s) \leftarrow a$
16 $U^t(s) \leftarrow Q^*$
17 **until** $(U^t)^{lmaxlmin}_{l,c} == (U^{t-1})^{lmaxlmin}_{l,c}$;
18 $\delta^*(s) \leftarrow argmax_a Q(s,a)$
19 **return** δ^*;

6.6 Summary

In this Chapter, we have studied the lexicographic optimization of policies in stationary possibilistic Markov decision processes.

First, considering finite-horizon stationary possibilistic Markov decision processes, we have proposed a lexicographic value iteration algorithm for the computation of lexicographic optimal policies, based on the whole matrices of trajectories. Since this algorithm is computationally expensive, we have proposed a bounded lexicographic value iteration algorithm as an approximation

of the full lexicographic procedure. The principle of this algorithm is to forget the less useful part of the matrix of trajectories and to decide based on the best part i.e. the (l, c) sub-matrix. When we fix the size of the matrices, the complexity of the algorithm becomes polynomial. We have compared the bounded lexicographic algorithm and the classical lexicographic one on randomly generated problems with different horizons and different values of (l, c). It appears that the bounded variant is an interesting algorithm: it is very fast and it provides a good approximation, with large (l, c).

For the infinite-horizon case, using the classical lexicographic value iteration algorithm, the matrices of trajectories increase infinitely. Hence, we use the same idea of bounding matrices and the only thing that changes is the condition of convergence, since we do not know the number of iteration in advance. Moreover, we have proved that, using this algorithm, the complexity remains polynomial in (l, c), even when the horizon is infinite. Note that it is possible to extend this work to the pessimistic case, so we compute the pessimistic utility of each trajectory and compare these utilities using $lmin(lmax)$ criterion whereas $lmax(lmin)$.

CONCLUSION

The contributions of this thesis are mainly related to the preliminary works of Fargier et Sabbadin [Fargier and Sabbadin, 2003, Fargier and Sabbadin, 2005]. They have proposed refinements of optimistic and pessimistic qualitative utilities using a special form of classical expected utility equivalent to qualitative lexicographic ordering. The main objective of our work was to extend these criteria for the sequential decision-making framework.

The problematic of this work, exposed in Chapter 3, is how to overcome the drowning effect of qualitative utilities when comparing policies. Therefore, we have proposed two lexicographic criteria that compare policies based on their corresponding matrices of trajectories. These comparisons satisfy the principle of efficiency and strict monotonicity. These properties allow us to define lexicographic backward induction algorithms for possibilistic decision trees and for the finite-horizon possibilistic Markov decision processes. We have proved that, for the two sequential models, the lexicographic algorithms are discriminant and avoid the drowning effect of classical possibilistic algorithms. The algorithms proposed are polynomial in the size of the model and allow to get an optimal discriminant policy. But the matrices of trajectories grow exponentially with the horizon, so it is possible to bound the number of lines and to decide only using the bounded part of the matrices. Hence, we propose a bounded version of the lexicographic backward induction which has polynomial complexity in the horizon and the number of lines and columns of the bounded matrices. Then, we have experimentally compared the lexicographic algorithms (the two version) proposed and the classical possibilistic algorithms. It appears that the backward induction algorithm with bounded matrices is fast and provides very good approximations, especially when the number of lines kept is high. This approximation algorithm has been extended (in Chapter 6) to stationary possibilistic Markov decision processes that have intermediate utilities, when the horizon is finite or infinite.

On the theoretical side, we have proved that lexicographic criteria encode expected utility criteria based on the transformation of the possibilistic model to probabilistic one, using big stepped probabilities.

As short term future work, various algorithmic extensions should be concerned such as:

- We may think about the adaptation of the classical Possibilistic Policy Iteration algorithm (see Algorithm 2.4) [Sabbadin, 2001] for the lexicographic procedures, which is not too difficult to realize
- In addition, as far as the infinite horizon case is concerned, other types of lexicographic refinements could be proposed. One of these options could be to avoid the duplication of the set of transitions that occur several times in a single trajectory and consider only those which are observed.
- The next step is obviously to develop simulation-based algorithms for finding lexicographic solutions in Markov decision process. Reinforcement Learning algorithms [Sutton and Barto, 1998] allow to solve large size (probabilistic) Markov decision process by making use of simulated trajectories of states to optimize a strategy. It is not immediate to develop reinforcement learning algorithms for possibilistic Markov decision processes, since no unique stochastic transition function corresponds to a possibility distribution. However, uniform simulation of trajectories (with random choice of actions) may be used to generate an approximation of the possibilistic decision tree (provided that both transition possibilities and utility of the leaf are given with the simulated trajectory). So, interleaving simulations and lexicographic dynamic programming may lead to reinforcement learning-type algorithms for approximating lexicographic-optimal policies for (large) possibilistic Markov decision processes. Such algorithms would use samplings of the trajectories instead of full dynamic programming or quantile-based reinforcement learning approaches [Gilbert and Weng, 2016].

The work presented in this thesis has answered several questions concerning discriminant policies in sequential decision-making models. It thus opened up many promising research tracks:

- The axiomatization of the lexicographic criteria is possible and it may be envisaged in further research. It would consist in the VNM axioms of rational decision since our work is more close to lotteries framework and that we do not use the reduction of lotteries.
- In [Weng, 2005], Weng has proposed a refinement of binary possibilistic utilities (BPU) and as a particular case, to classical, optimistic and pessimistic, possibilitistic utilities. This refinement allows to improve the discrimination power of BPU. But since in [Weng, 2005] treatment indeed, two similar trajectories of the same strategy are merged, the resulting criterion thus suffers from a drowning effect and does no satisfy strict monotonicity: as such, it cannot be represented by an EU-based criterion which "counts" trajectories (weighted by their probabilities). We actually do refine [Weng, 2005]'s criterion. Incorporating our lexicographic refinements in BPU would lead to a more powerful refinement and suggest a probabilistic interpretation of efficient BPU. It also leads to new algorithms that are more discriminant than their original counterparts.

- Another problem remains open: how to refine the coarse ranking produced by Sugeno integral, especially when we consider sequential decision problems ? [Dubois and Fargier, 2007] proposes to use a mapping from the qualitative scale (of both utility and uncertainty) to the reals, hence, Sugeno integral may be refined by a Choquet integral. In our opinion, the extension of this work to sequential decision-making is a complicated task: first we have to define what is it a Sugeno of a matrix of trajectories and how we calculate the Sugeno of a compound lottery (or a policy). Then, we have to decide which solving algorithm shall be used (dynamic programming, branch and bound, etc.), it depends on the axioms and properties of the new refined criterion (Choquet integral or another one). Note that [Dubois and Fargier, 2007]'s refined criterion of Sugeno integral is not yet axiomatized.

- Another line of research is to consider collective decision-making under possibilistic uncertainty [Ben Amor et al., 2015b, Ben Amor et al., 2015a]. In fact, since it is based on optimistic and pessimistic possibilistic utilities, it suffers from a lack of decisiveness. So, it is interesting to look for collective lexicographic decision rules as an efficient counterpart of collective qualitative ones, in order to found discriminant decisions, which satisfies the collectivity, in sequential decision-making models such as possibilistic decision trees and possibilistic Markov decision processes.

- Finally, our lexicographic approach is may be useful for optimizing policies in other possibilistic variants of sequential decision models, such as Partially Observable Markov Decision Processes (POMDPs) and Factored Markov Decision Processes (FMDPs). One may also think about adapting the lexicographic criteria to possibilistic planning problems that can be used to search optimal plan or a plan that lead to a goal state [da Costa Pereira et al., 1997]. Note that, the integration of the discriminative lexicographic criteria is interesting but it eventually makes these problems more complex.

Bibliography

[Allais, 1953] Allais, M. (1953). Le Comportement de l'Homme Rationnel devant le Risque: Critique des Postulats et Axiomes de l'Ecole Americaine. *Econometrica*, 21(4):503–546.

[Bauters et al., 2016] Bauters, K., Liu, W., and Godo, L. (2016). Anytime algorithms for solving possibilistic MDPs and hybridMDPs. In *Proceedings of 9th International Symposium on Foundations of Information and Knowledge Systems (FoIKS'2016)*, pages 24–41.

[Bellman, 1957a] Bellman, R. (1957a). *Dynamic Programming*. Princeton University Press.

[Bellman, 1957b] Bellman, R. (1957b). A markovian decision process. *J. of Mathematics and Mechanics*, 6.

[Ben Amor et al., 2015] Ben Amor, N., El Khalfi, Z., Fargier, H., and Sabbadin, R. (2015). Raffinement de la décision séquentielle possibiliste : de l'utilité qualitative optimiste à l'utilité espérée. page 196–203.

[Ben Amor et al., 2016a] Ben Amor, N., El Khalfi, Z., Fargier, H., and Sabbadin, R. (2016a). Lexicographic refinements in possibilistic decision trees. In *Proceedings of the 22nd European Conference on Artificial Intelligence (ECAI'2016)*, pages 202–208.

[Ben Amor et al., 2016b] Ben Amor, N., El Khalfi, Z., Fargier, H., and Sabbadin, R. (2016b). Lexicographic refinements in possibilistic markov decision processes : The finite horizon case. In *Proceedings of 25e Rencontres francophones sur la Logique Floue et ses Applications (LFA'2016)*.

[Ben Amor et al., 2017] Ben Amor, N., El Khalfi, Z., Fargier, H., and Sabbadin, R. (2017). Efficient policies for stationary possibilistic markov decision processes. In *Proceedings of 14th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU'2017)*, pages 306–317.

[Ben Amor et al., 2015a] Ben Amor, N., Essghaier, F., and Fargier, H. (2015a). Décision collective sous incertitude possibiliste. Principes et axiomatisation. *Revue d'Intelligence Artificielle, La logique floue et ses applications*, 29(5):515–542.

[Ben Amor et al., 2015b] Ben Amor, N., Essghaier, F., and Fargier, H. (2015b). Egalitarian collective decision making under qualitative possibilistic uncertainty: Principles and characterization. In *Proceedings of the 29th Conference on Artificial Intelligence (AAAI'2015)*, pages 3482–3488.

[Ben Amor et al., 2010] Ben Amor, N., Fargier, H., and Guezguez, W. (2010). Necessity-based choquet integrals for sequential decision making under uncertainty. In *Proceedings of the Computational Intelligence for Knowledge-based Systems Design, and 13th International Conference on Information Processing and Management of Uncertainty (IPMU'2010)*, IPMU'10.

[Ben Amor et al., 2014] Ben Amor, N., Fargier, H., and Guezguez, W. (2014). Possibilistic sequential decision making. *International Journal of Approximate Reasoning*, 55:1269–1300.

[Benferhat et al., 1999] Benferhat, S., Dubois, D., and Prade, H. (1999). Possibilistic and standard probabilistic semantics of conditional knowledge bases. *Journal of Logic and Computation*, 9:873–895.

[Bernoulli, 1738] Bernoulli, D. (1738). *Exposition of a New Theory on the Measurement of Risk*. Econometrica 22.

[Cayrol et al., 2014] Cayrol, C., Dubois, D., and Touazi, F. (2014). Ordres partiels entre sous-ensembles d'un ensemble partiellement ordonné. Research report RR–2014-02–FR, IRIT, Université Paul Sabatier, Toulouse.

[Choquet, 1954] Choquet, G. (1954). Theory of capacities. *Annales de l'institut Fourier*, 5:131–295.

[da Costa Pereira et al., 1997] da Costa Pereira, C., Garcia, F., Lang, J., and Martin-Clouaire, R. (1997). Possibilistic planning: Representation and complexity. In *Proceedings of 4th European Conference on Planning, (ECP'1997)*, pages 143–155.

[Drougard et al., 2013] Drougard, N., Teichteil-Königsbuch, F., Farges, J.-L., and Dubois, D. (2013). Qualitative possibilistic mixed-observable MDPs. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI'2013)*, pages 192–201.

[Drougard et al., 2014] Drougard, N., Teichteil-Konigsbuch, F., Farges, J.-L., and Dubois, D. (2014). Structured possibilistic planning using decision diagrams. In *Proceedings of the 28th Conference on Artificial Intelligence (AAAI'2014)*, pages 2257–2263.

[Dubois and Fargier, 2007] Dubois, D. and Fargier, H. (2007). Lexicographic refinements of sugeno integrals. In *Proceedings of 9th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU'2007)*, pages 611–622.

[Dubois et al., 2003] Dubois, D., Fargier, H., and Perny, P. (2003). Qualitative decision theory with preference relations and comparative uncertainty: An axiomatic approach. *Artif. Intell.*, 148(1-2):219–260.

[Dubois et al., 1996] Dubois, D., Fargier, H., and Prade, H. (1996). Refinements of the maximin approach to decision-making in fuzzy environment. *Fuzzy Sets and Systems*, 81:103–122.

[Dubois et al., 1998a] Dubois, D., Godo, L., Prade, H., and Zapico, A. (1998a). Making decision in a qualitative setting: from decision under uncertainty to case-based decision. In *Proceedings of the 6th International Conference on Principles of Knowledge Representation and Reasoning (KR'1998)*, pages 594–605.

[Dubois et al., 1999] Dubois, D., Godo, L., Prade, H., and Zapico, A. (1999). On the possibilistic decision model: From decision under uncertainty to case-based decision. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 7(6):631–670.

[Dubois et al., 2000] Dubois, D., Godo, L., Prade, H., and Zapico, A. (2000). Advances in qualitative decision theory: Refined rankings. In *Proceedings of Advances in Artificial Intelligence: International Joint Conference 7th Ibero-American Conference on AI 15th Brazilian Symposium on AI (IBERAMIA-SBIA'2000)*, pages 427–436.

[Dubois et al., 2001a] Dubois, D., Marichal, J.-L., Prade, H., Roubens, M., and Sabbadin, R. (2001a). The use of the discrete sugeno integral in decision-making: A survey. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 9:539–561.

[Dubois and Prade, 1988] Dubois, D. and Prade, H. (1988). *Possibility Theory: An Approach to Computerized Processing of Uncertainty*. Plenum Press, New York.

[Dubois and Prade, 1995] Dubois, D. and Prade, H. (1995). Possibility theory as a basis for qualitative decision theory. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI'1995)*, pages 1925–1930.

[Dubois et al., 1998b] Dubois, D., Prade, H., and Sabbadin, R. (1998b). Qualitative decision theory with sugeno integrals. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI'1998)*, pages 121–128.

[Dubois et al., 2001b] Dubois, D., Prade, H., and Sabbadin, R. (2001b). Decision-theoretic foundations of qualitative possibility theory. *European Journal of Operational Research*, 128:459–478.

[Dubois and Rico, 2016a] Dubois, D. and Rico, A. (2016a). Axiomatisation of discrete fuzzy integrals with respect to possibility and necessity measures. In *Proceedings of 13th International Conference on Modeling Decisions for Artificial Intelligence (MDAI'2016)*, pages 94–106.

[Dubois and Rico, 2016b] Dubois, D. and Rico, A. (2016b). Axiomatisation of discrete fuzzy integrals with respect to possibility and necessity measures. In *Proceedings of the 13th Conference on Modeling Decisions for Artificial Intelligence (MDAI'2016)*, pages 94–106.

[Ellsberg, 1961] Ellsberg, D. (1961). Risk, ambiguity, and the savage axioms. *The Quarterly Journal of Economics*, 75(4):643–669.

[Fargier and Perny, 1999] Fargier, H. and Perny, P. (1999). Qualitative models for decision under uncertainty without the commensurability assumption. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI'1999)*, pages 188–195.

[Fargier and Sabbadin, 2000] Fargier, H. and Sabbadin, R. (2000). Can qualitative utility criteria obey the sure thing principle ? In *Proceedings of the 8th Conference on Information Processing Managment of Uncertainly (IPMU'2000)*, pages 821–826.

[Fargier and Sabbadin, 2002] Fargier, H. and Sabbadin, R. (2002). Can qualitative utility criteria obey the sure thing principale? In Bouchon-Meunier, B., Gutierrez-Rios, J., Magdalena, L., and Yager, R., editors, *Technologies for Constructing Intelligent Systems 1 Tools*, pages 167–178. Physica-Verlag.

[Fargier and Sabbadin, 2003] Fargier, H. and Sabbadin, R. (2003). Qualitative decision under uncertainty: Back to expected utility. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI'2003)*, pages 303–308.

[Fargier and Sabbadin, 2005] Fargier, H. and Sabbadin, R. (2005). Qualitative decision under uncertainty: back to expected utility. *Artificial Intelligence*, 164:245–280.

[Garcia and Sabbadin, 2006] Garcia, L. and Sabbadin, R. (2006). Possibilistic influence diagrams. In *Proceedings of the 17th European Conference on Artificial Intelligence (ECAI'2006)*, pages 372–376.

[Giang and Shenoy, 2001] Giang, P. and Shenoy, P. P. (2001). A comparison of axiomatic approaches to qualitative decision making using possibility theory. In *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence (UAI'2001)*, pages 162–170.

[Giang and Shenoy, 2005] Giang, P. and Shenoy, P. P. (2005). Two axiomatic approaches to decision making using possibility theory. *European Journal of Operational Research*, 162:450–467.

[Gilbert and Weng, 2016] Gilbert, H. and Weng, P. (2016). Quantile reinforcement learning. *Computing Research Repository*.

[Gilbert et al., 2017] Gilbert, H., Weng, P., and Xu, Y. (2017). Optimizing quantiles in preference-based markov decision processes. In *Proceedings of the 31st Conference on Artificial Intelligence (AAAI'2017)*, pages 3569–3575.

[Godo and Zapico, 2001] Godo, L. and Zapico, A. (2001). On the possibilistic-based decision model: Characterization of preference relations under partial inconsistency. *Applied Intelligence*, 14:319–333.

[Godo and Zapico, 2005] Godo, L. and Zapico, A. (2005). Lexicographic refinements in the context of possibilistic decision theory. In *Proceedings of the 4th EUSFLAT & 11th LFA Conference*, pages 559 – 564.

[Grabisch et al., 2000] Grabisch, M., Murofushi, T., and Sugeno, M. (2000). *Fuzzy Measures and Integrals: Theory and Applications*. Studies in Fuzziness and Soft Computing. Physica-Verlag HD.

[Howard, 1960] Howard, R. A. (1960). *Dynamic Programming and Markov Processes*. MIT Press, Cambridge, MA.

[Howard and Matheson, 1984] Howard, R. A. and Matheson, J. E. (1984). Influence diagrams. *In the principles and applications of decision analysis*, 2.

[Lehmann, 2002] Lehmann, D. J. (2002). Expected qualitative utility maximization. *Computing Research Repository*, cs.GT/0202023.

[Marichal, 2001] Marichal, J.-L. (2001). An axiomatic approach of the discrete sugeno integral as a tool to aggregate interacting criteria in a qualitative framework. *IEEE Transactions on Fuzzy Systems*, 9:164–172.

[Montes et al., 2014] Montes, I., Miranda, E., and Montes, S. (2014). Decision making with imprecise probabilities and utilities by means of statistical preference and stochastic dominance. *European Journal of Operational Research*, 234:209–220.

[Moulin, 1988] Moulin, H. (1988). *Axioms of Cooperative Decision Making*. Cambridge University Press.

[Neumann and Morgenstern, 1944] Neumann, J. V. and Morgenstern, O. (1944). *Theory of games and economic behavior*.

[Pearl, 1993] Pearl, J. (1993). From conditional oughts to qualitative decision theory. In *Proceedings of the 9th International Conference on Uncertainty in Artificial Intelligence (UAI'1993)*, pages 12–20.

[Perny et al., 2005] Perny, P., Spanjaard, O., and Weng, P. (2005). Algebraic markov decision processes. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI'2005)*, pages 1372–1377.

[Puterman, 1994] Puterman, M. L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition.

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

[Raiffa, 1968] Raiffa, H. (1968). *Decision Analysis: Introductory Lectures on Choices under Uncertainty*. Addison Wesley.

[Rebille, 2006] Rebille, Y. (2006). Decision making over necessity measures through the choquet integral criterion. *Fuzzy Sets and Systems*, pages 3025–3039.

[Sabbadin, 1999] Sabbadin, R. (1999). A possibilistic model for qualitative sequential decision problems under uncertainty in partially observable environments. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI'1999)*, pages 567–574.

[Sabbadin, 2001] Sabbadin, R. (2001). Possibilistic Markov decision processes. *Engineering Applications of Artificial Intelligence*, 14:287–300.

[Sabbadin et al., 1998] Sabbadin, R., Fargier, H., and Lang, J. (1998). Towards qualitative approaches to multi-stage decision making. *International Journal of Approximate Reasoning*, 19:441–471.

[Savage, 1954] Savage, L. J. (1954). *The Foundations of Statistics*. J. Wiley, New York. second revised edition, 1972.

[Shafer, 1976] Shafer, G. (1976). *A Mathematical Theory of Evidence*. Princeton University Press.

[Snow, 1999] Snow, P. (1999). Diverse confidence levels in a probabilistic semantics for conditional logics. *Artificial Intelligence*, 113:269–279.

[Sugeno, 1974] Sugeno, M. (1974). *Theory of fuzzy integrals and its applications*. PhD thesis, Tokyo Institute of Technology.

[Sutton and Barto, 1998] Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning:An Introduction*. MIT Press.

[Szörényi et al., 2015] Szörényi, B., Busa-Fekete, R., Weng, P., and Hüllermeier, E. (2015). Qualitative multi-armed bandits: A quantile-based approach. In *Proceedings of the 32nd International Conference on Machine Learning (ICML'2015)*, pages 1660–1668.

[Wald, 1971] Wald, A. (1971). *Statistical decision functions*. Chelsea Pub. Co.

[Walley, 1991] Walley, P. (1991). *Statistical Reasoning with Imprecise Probabilities*. Chapman & Hall.

[Weng, 2005] Weng, P. (2005). Qualitative decision making under possibilistic uncertainty: Toward more discriminating criteria. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence (UAI'2005)*, pages 615–622.

[Weng, 2011] Weng, P. (2011). Markov decision processes with ordinal rewards: Reference point-based preferences. In *Proceedings of the 21st International Conference on Automated Planning and Scheduling (ICAPS'2011)*, pages 282–289.

[Whalen, 1984] Whalen, T. (1984). Decision making under uncertainty with various assumptions about available information. *IEEE Trans. on Systems, Man and Cybernetics*, 14:888–900.

[Yager, 1997] Yager, Ronald R., K. J., editor (1997). *The Ordered Weighted Averaging Operators: Theory and Applications*. Kluwer Academic Publishers, Norwell, MA, USA.

[Yager, 1979] Yager, R. R. (1979). Possibilistic decision making. *IEEE Trans. on Systems, Man and Cybernetics*, 9:388–392.

[Yue et al., 2012] Yue, Y., Broder, J., Kleinberg, R., and Joachims, T. (2012). The k-armed dueling bandits problem. *Journal of Computer and System Sciences*, 78(5):1538–1556.

[Zadeh, 1965] Zadeh, L. (1965). Fuzzy sets. *Information Control*, 8:338–353.

[Zadeh, 1978] Zadeh, L. (1978). Fuzzy sets as a basis for theory of possibility. *Fuzzy Sets and Systems*, 1:3–28.