Learning Majority-Rule models with partially monotone data

Pegdwendé Minoungou

▶ **To cite this version:**

Pegdwendé Minoungou. Learning Majority-Rule models with partially monotone data. Automatic Control Engineering. Université Paris-Saclay, 2022. English. NNT : 2022UPAST058 . tel-03717431

HAL Id: tel-03717431
https://theses.hal.science/tel-03717431

Submitted on 8 Jul 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

université
PARIS-SACLAY

Learning Majority-Rule models with partially monotone data

Apprentissage de modèles à règle majoritaire à partir de données partiellement monotones

Thèse de doctorat de l'université Paris-Saclay

École doctorale n° 573 :
Interfaces : matériaux, systèmes, usages (INTERFACES)
Spécialité de doctorat : Informatique
Graduate School : Sciences de l'ingénierie et des systèmes,
Référent : CentraleSupélec

Thèse préparée dans l'unité de recherche Mathématiques et Informatique pour la Complexité et les Systèmes (Université Paris-Saclay, CentraleSupélec), sous la direction de Vincent MOUSSEAU, Professeur, le co-encadrement de Wassila OUERDANE, Maîtresse de Conférences et de Paolo Scotton, Docteur, tuteur industriel (IBM)

Thèse soutenue à Paris-Saclay, le 13 Mai 2022, par

Pegdwendé MINOUNGOU

Composition du jury

Patrick Meyer Professeur, IMT Atlantique, Brest	Président & Examinateur
Meltem Oztürk Maîtresse de Conférences HDR, PSL, Université Paris-Dauphine	Rapportrice & Examinatrice
Marc Pirlot Professeur, Université de Mons, Belgique	Rapporteur & Examinateur
Miguel Couceiro Professeur, Université de Lorraine, Nancy	Examinateur
Vincent Mousseau Professeur, CentraleSupélec, Université Paris-Saclay	Directeur de thèse
Wassila Ouerdane Maîtresse de Conférences, CentraleSupélec, Université Paris-Saclay	Co-encadrante de thèse

THESE DE DOCTORAT
NNT : 2022UPAST058

Remerciements

Cela a été une expérience très intéressante et enrichissante de mener ces travaux de recherche. Trois ans plus tôt, je n'aurais jamais pu imaginer le parcours de cette thèse qui peut s'apparenter à la fois à un marathon - tant il m'a fallu composer avec le temps et la persévérence - et à une exploration minière, où la curiosité intellectuelle a nourrit ma motivation pour toujours aller plus loin dans ces travaux. Pegdwendé ! Que je suis reconnaissant d'avoir atteint cet objectif !

Je tiens à remercier tout ceux qui m'ont soutenu durant ces années, en particulier mon équipe encadrante. Je remercie tout d'abord mon directeur de thèse Vincent Mousseau pour son encadrement, sa bienveillance et sa disponibilité. Je suis reconnaissant à Wassila Ouerdane pour ses conseils, notamment sa bienveillance et sa rigueur. Je tiens à dire un grand merci à Paolo Scotton pour son aide et ses encouragements tout le long de la thèse. Vous avez contribué à faire de ces moments, des moments agréables et j'ai beaucoup appris à vos côtés. Merci pour votre confiance !

Je veux aussi remercier les rapporteurs de la thèse, Meltem Oztürk et Marc Pirlot d'avoir accepté la tâche qui a été la leur. Merci d'avoir pris le temps de relire de bout en bout la thèse, d'avoir apporté des corrections et émis des commentaires très pertinents sur mon travail. J'exprime ma gratitude envers les examinateurs de ma thèse, Miguel Couceiro, et en particulier Patrick Meyer (président du jury) qui a aussi évalué mon travail à mi-parcours de la thèse. Une fois encore merci pour vos remarques instructives et l'intérêt porté à mon sujet de thèse.

Je suis reconnaissant à Christian de Sainte Marie qui m'a toujours apporté son regard éclairé sur mes travaux. Merci pour ta disponibilité et ta bienveillance.

Merci à mes collègues de CentraleSupelec Ali, Yassine, Manuel, Matthieu pour leur présence, les bons moments partagés ensemble et leur soutien multiforme.

Je remercie également l'équipe de IBM Research Paris, Marine, Maxence, Shubham, Yusik et Remy pour les conseils, les corrections sur le manuscrit et pour la sympathie manifestée à mon égard. Merci à Lex (on doit "jouer la belle" au scrabble :-)) , Nicolas, Thomas et Yunpeng d'IBM pour votre sympathie et nos différents échanges sur ma thèse et surtout en dehors.

Je conclus mon propos en remerciant toute ma famille (ma mère, mes frères

et soeurs, et ma famille élargie :-)) qui a m'a encouragé et soutenu de manière multiforme durant ces années de thèse.

Merci à tous !

Abstract

The field of Multiple Criteria Decision Analysis (MCDA) deals with alternatives evaluated by several criteria, aiming to recommend the "best" decision to the decision-maker (DM). In this context, we are interested in the indirect learning paradigm which is comparable to machine learning tasks as it consists of inferring from past observations of the DM, the model parameters that suit the DM's preferences. Our model (MR-Sort) stems from the MCDA family of outranking models, where an alternative a outranks another alternative b if there is a strong support of criteria (a majority in MR-Sort) that favors a compared to b. In the literature, methods and algorithms used for sorting problems - classification into predefined and ordered categories - always infer MR-Sort models with known criteria preference directions and monotone (increasing or decreasing) preferences. In this thesis, we extend the state-of-the-art to single-peaked (and single-valley) criteria which improves the expressivity of MR-Sort models. A single-peaked criterion relates to two successive monotonicities (increasing then decreasing). Therefore we investigate the problem of learning the MR-Sort parameters from monotone and single-peaked preferences regardless of the knowledge of preference directions of criteria. We propose an exact method and heuristics, and conducted experiments to assess and compare our algorithms regarding the computational cost, the classification accuracy and the preference directions retrieval.

Résumé

Le domaine de l'Aide à la Décision Multicritère (ADMC), s'interesse à évaluer des alternatives suivant des critères dans le but de recommander la "meilleure" solution au décideur. Dans ce contexte, nous considerons le paradigme d'apprentissage de préférences - comparable à l'approche en Machine Learning - qui consiste à déduire à partir des observations passées du décideur, les paramètres du modèle qui correspondent au mieux à ses préférences. Notre modèle (MR-Sort) est issu de la famille des modèles de surclassement, dans lequel une alternative a surclasse une autre alternative b s'il existe une forte coalition de critères (majorité) favorable au surclassement de a par rapport à b. Dans la littérature de l'ADMC, les méthodes et algorithmes étudiés pour les problèmes de tri - classification dans des catégories prédéfinies et ordonnées - ont toujours eu pour but l'inférence de modèles MR-Sort connaissant le sens de préférence des critères et à partir de préférences monotones (croissantes ou décroissantes). Dans cette thèse, nous étendons l'état de l'art à l'étude des préférences dites "single-peaked" (resp. "single-valley"), qui améliorent l'expressivité des modèles MR-Sort. Un critère single-peaked est caractérisé par deux monotonies successives (croissante puis décroissante). Ainsi, nous étudions des problèmes d'apprentissage des paramètres de MR-Sort à partir de préférences monotones et single-peaked, quelle que soit la connaissance à priori des sens de préférences des critères. Nous proposons une méthode exacte, des heuristiques et des tests pour évaluer et comparer nos algorithmes suivant le temps de calcul, le taux de classification et de restitution des sens de préferences.

List of Notations

MCDA Generalities

n	Number of criteria
m	Number of alternatives
p	Number of categories
$\mathcal{N} = \{1, .., i, .., n\}$	Set of criteria
X_i	Evaluation scale of the criterion i
$\succsim_i$	Preference order on X_i
$<_i$	Order on $\mathbb{R}$ of the values of X_i
$X = \prod_{i=1}^{n} X_i$	Cartesian product of criteria scales
$A^\star \subseteq X$	Set of reference alternatives
$A^{test} \subseteq X$	Set of generated alternatives for tests (test set)
a_i^j	Evaluation of alternative a_j on criterion i
$a^j = (a_1^j, ..., a_n^j) \in X$	Alternative a^j
$X_i^\star = \{a_i^j; \forall a^j \in A^\star\}$	Evaluation scale of the criterion i induced by the reference alternatives
$X^\star = \prod_{i=1}^{n} X_i^\star$	Cartesian product of criteria scales induced by the reference alternatives
$c(a^j) \in \{C^1, .., C^h, .., C^p\}$	Desired category for alternative a^j
$C = (c(a^j), \forall a^j \in A^\star)$	Set of desired categories for alternatives in $A^\star$
$\triangleright$	Order on categories
$L = (A^\star, C)$	Learning set

MR-Sort

w_i	Weight of criterion i
$w = (w_1, ..., w_n)$	Vector of criteria weights
$\lambda \in]0, 1]$	Majority threshold
b_i^h	Frontier value on criteria i delimiting approved values in C^h from the ones in C^{h+1}
$b^h = (b_0^h, .., b_n^h) \in X$	Limit profile delimiting category C^h from C^{h+1}
$\langle b \rangle = (b^0, .., b^p)$	Limit profile vector
$\mathcal{A}_i^h \in X_i$	Approved set of values on criterion i in favor to an assignment to category C^{h+1}
$\mathcal{F}^h \subseteq 2^{\mathcal{N}}$	Set of sufficiently strong subsets of criteria related to category C^h
$\mathcal{M}$	MR-Sort model (initial)

Preference directions

d_i	Preference direction of criterion i
$d = (d_1, .., d_n)$	Vector of preference directions
q, $q \leq n$	Number of criteria with unknown preference directions
$\mathcal{Q}$, $\mathcal{Q} \subseteq \mathcal{N}$	Set of criteria with unknown preference directions
$\mathcal{Q}'$	Duplicated set of criteria with opposite preference directions to those in $\mathcal{Q}$
$(i, i'), i \in \mathcal{Q}, i' \in \mathcal{Q}'$	Associated couple of a duplicated criterion i and its duplicate i'
$\mathcal{AC}$	Set of the q associated couples of criteria (i, i')
$IMS_{q\|n}$	Inverse MR-Sort problem with q unknown preference directions over n criteria
$IMS_{0\|1}^i$	Inverse MR-Sort problem where the unique criterion in the model is i

Single-peaked criteria

s, $s \leq n$	Number of single-peaked criteria
$\mathcal{S}$, $\mathcal{S} \subseteq \mathcal{N}$	Set of single-peaked and single-valley criteria
p_i	Peak value of the single-peaked criterion i
$\overline{b_i^h}$	Top frontier value of the single-peaked criterion i delimiting approved values in C^h from the ones in C^{h+1}
$\underline{b_i^h}$	Bottom frontier value of the single-peaked criterion i delimiting approved values in C^h from the ones in C^{h+1}
$b_i^{\perp h}$	Middle value of the interval $[\underline{b_i^h}, \overline{b_i^h}]$
$\mathcal{M}^{\mathcal{SP}}$	MR-Sort model including some single peaked/valley criteria
$IMSSP_{s\|n}$	Inverse MR-Sort problem with s single-peaked criteria over n criteria

Algorithms parameters and metrics

μ	Threshold rate on the scale X_i
CA_i	Restoration rate of the learned model on the restricted problem $IMS^i_{0\|1}$
rr	Renewal rate
σ	Coefficient of renewal
$\overline{CA}$	Average restoration rate
N_{mod}	Number of models in the population
f_w	Function of the readapted metaheuristic that matches each iteration to a minimum weight value
f_φ	Function that matches each iteration to the desired order of interquantile range for the generation of profiles values
π_i^+, π_i^-	For a given model, the probability of generating resp. a gain criterion, cost criterion for i
$\pi_i = [\pi_i^+, \pi_i^-]$	For a given model, the probability distribution of generating monotone preference directions for criterion i
w_*	The minimal value used as a limit on the criteria weights in the readapted formulation
φ^{th}	Maximum order of interquantile range of X_i
N_o	Number of iterations of the outer loop
N_{it}	Number of iterations of the inner loop
it	The current iteration of the metaheuristic
ρ	Percentage of noise in the model (i.e. percentage of alternatives that are purposely misclassified to introduce noise)
CA_v	Restoration rate (validation)
CA_g	Restoration rate (generalization)
PDR_{all}	Preference direction restoration rate considering the probability to restore all preference directions in Q
PDR_{one}	Preference direction restoration rate of a preference direction on average in Q

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The field of Multi-Criteria Decision Analysis (MCDA) stems from the broader area of Decision Theory. It takes advantage of the existence of multiple viewpoints (which we designate by *criteria*) on objects (which we refer as *alternatives*) to make a holistic decision. The main actor concerned by this decision is the Decision Maker (DM), whose role is to take the best decision according to his or her preferences. The second actor is the Decision Analyst (DA), who interacts with the DM, provides his expertise in the domain of the decision and guides the DM throughout the decision process. Therefore the DM helped by the DA are both part of the decision process, which can be summarized in three steps : identifying and formulating the problem, choosing the appropriate resolution tool, evaluating and recommending the best decision [95].

Decision problems may refer to performing one of three possible tasks, as introduced by [80] : choosing among a set of alternatives, ranking alternatives, or sorting alternatives into categories (which are predefined and ordered).

Over time, several approaches about the rationale behind the decision and its aim have emerged [34]. This has given birth to four approaches : normative, descriptive, prescriptive and constructive approaches. The *normative approach* relates to the universal and rational behaviour typically adopted by the DM. The *descriptive approach* puts an emphasis on how the DM makes the decision. The *prescriptive approach* aims to sketch the model that describes the DM preferences in order to properly suggest relevant alternatives. Finally, the *constructive approach* consists of building progressively a representation of the DM preferences with the help of the DM himself.

We ground our work within the constructive approach with the objective of learning new preference structures in the presence of partially monotone data. As we operate in the context of classification problems [76], we adopt the following definition of *partially monotone data*. We consider data where relationships between the criteria evaluations (inputs) and the assignments into categories (output

decisions) are either monotone or non monotone (with local monotone relationships) [14]. In other words, the latter relationship is monotone by pieces.

The level of glycemia in brittle diabetes diagnosis, the tire pressure of a car in assessing the road safety, the amount of money to engage into risky prize games, the range of skills required in order to be hired for a specific job are all relationship examples illustrating criteria values that do not only increase (or only decrease) with the decision ; they are also characterized by at least one change of monotonicity relationship. In this thesis, we will focus on criteria with a single change of monotonicity, that is *single-peaked criteria* (increasing then decreasing, i.e the optimum is the maximum) and *single-valley criteria* (decreasing then increasing, i.e the optimum is the minimum) .

The handling of monotonicity features in classification tasks is a current subject of research in disciplines such as Machine Learning or Data mining [102, 100, 25]. In MCDA, this topic has been discussed when considering additive models [64, 53, 58] - which refer to models that aggregate alternatives into a single value - but not yet with outranking models in particular the Majority Rule Sorting method (MR-Sort) [63].

The main idea of outranking models for sorting tasks is to assign alternatives to a predefined and ordered set of categories in accordance to preferences conditions for some given criteria coalitions based on the relative importance of these criteria. MR-Sort stems from the ELECTRE TRI method [104], which is known as a method that enables the expression of uncertainty on comparisons between alternatives and to a certain extent, avoids compensation effects. Such effects are typically encountered in additive models [61, 65]. The MR-Sort method is a reduced model of the ELECTRE TRI method and has an intuitive and descriptive power - the parameters of the model make it easy to interpret decisions formulated by the model. Sobrie *et al.* [89] used this model for two applications in the area of anesthesia and intensive care. They used a dataset containing 898 patients clinical details (age, diabetic status, respiratory failure, etc). These health parameters were used as criteria in a MR-Sort model using as categories four ASA[1] scores ; each score indicates the risk level undergone by a given patient while receiving anesthesia. The ASA score is an indicator of a patient's health. Subsequently, Sobrie *et al.* used the ASA score as a criterion with two other criteria for another MR-Sort model to predict the acceptance or refusal of patients for surgery. The authors showed that the results were better than those from machine learning algorithms such as SVM, KNN and MLP [62, 3]. Moreover, the MR-Sort algorithm provides an interpretation of the recommendations. MR-Sort parameters can be expressed by simple rules as "*if (condition 1) and (condition 2) and ..., then decision 1*", which are comprehensible by the DM like physicians in [89].

[1] American Association of Anesthesiologists

In this work, we adopt the Majority Rule Sorting method (MR-Sort) [63] as the framework of choice to learn the preference features with partially monotone data. One advantage of the MR-Sort is its flexibility on the use of resolution algorithms to handle various dataset sizes : exact methods for small dataset sizes, and heuristics for large dataset sizes. Furthermore, addressing this inference problem with MR-Sort in such context is challenging since learning NCS - a model that generalizes MR-Sort - parameters have been proved as NP-hard [8].

In this thesis, we have based our work on MR-Sort as it is a promising technique to generate recommendations considering large alternative datasets. MR-Sort, however, requires the criteria preference direction to be *(i)* known and *(ii)* monotone. The main contributions are:

1. Extension of MR-Sort to deal with unknown monotone preferences directions.
2. An exact method to detect and learn both monotone and single-peaked or single-valley preferences.
3. Extension of MR-Sort to deal with single-peaked or single-valley preferences.

To provide a visual representation of the contributions of this thesis, Figure 1.1 provides an illustration of where the three main contributions are situated. The nature of our problems are of two kinds : either preference directions are known or unknown (horizontal axis) and either preferences are monotone or single-peaked.

The north west side quarter represents the state-of-the-art with problems related to the learning of MR-Sort parameters with monotone preferences and knowing the preferences directions. Resolution methods have been implemented for such problems : a Mixed Integer Programming (MIP) [63], an evolutionary algorithm (a metaheuristic) [88] and a Boolean Satisfaction formulation (SAT) [10]. The north east side quarter represents problems related to the learning of MR-Sort parameters with monotone preferences with unknown preference directions for criteria (discussed in Chapter 4). The south west and east side quarters illustrate MR-Sort learning problems dealing with single-peaked preferences with known and unknown preference directions. They are discussed in Chapter 6 and partially covered in Chapter 7.

This thesis is organized as follows.

In chapter 2, we establish the Multi-Criteria Decision Analysis (MCDA) foundations of our work by giving the main concepts and definitions used in this document, as we contextualize our work which is the study of partially monotone preferences with a majority rule model (MR-Sort) for sorting problems. Then we explore the principal MCDA methodologies. Finally, we connect the MCDA field with

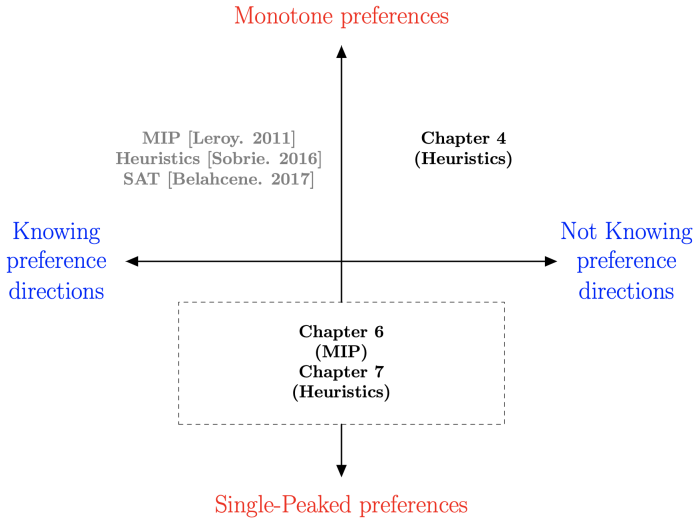


Figure 1.1: Overview illustrating the space of problems and our contributions in this thesis

Preference Learning, an emerging sub-field of Machine Learning which inspires us to tackle new problems with outranking models in particular MR-Sort.

Chapter 3 refers to the literature pertaining to our work which concerns two aspects : partially-monotone preferences and the Majority Rule sorting model, which is an outranking method. Therefore, we first explore related works on ordinal and monotone classification problems and algorithms. Subsequently we investigate existing research in the area of MR-Sort models. Finally, we describe more precisely an algorithm [92] used to learn monotone preferences with MR-Sort models (a metaheuristic algorithm which is an evolutionary algorithm) since two of the main contributions of this thesis are based on this algorithm.

In Chapter 4, we present the problem of learning MR-Sort parameters with latent preferences directions, which are preferences orders that are unknown to the decision actors. Yet the preferences are considered to be monotone. We propose two approaches for solving this problem : the first is a duplicated-based method that learns the preference directions apart from the rest of the MR-Sort parameters, the second one is a mixed-based method that takes into account both increasing and decreasing criteria preference directions in the MR-Sort resolution method. Lastly, we carried out some empirical experiments to assess the performance of our algorithms and to compare the approaches proposed.

In Chapter 5, we introduce a type of preference directions that has not yet been discussed for Non Compensatory Sorting (NCS) - a submodel of ELECTRE TRI - and MR-Sort models : single-peaked and single-valley preferences. First, we

formally describe single-peaked preferences and how they relate to MR-Sort models. Second, we establish bridges between this type of non monotone preferences and monotone preferences for MR-Sort models and illustrate them with examples.

Equipped with the characterization of single-peaked criteria in MR-Sort models, we study in Chapter 6 an exact method for learning MR-Sort models with those types of preferences, even without knowing the preference directions of criteria. We formulate a Mixed Integer Program (MIP) to solve this problem. Finally, we run both empirical and real-case experiments (ASA dataset). The results of this chapter teach us that even if the algorithm can be used in the problem of Chapter 4, the scalability of the method is limited (some instances even with four criteria do not terminate before one hour).

To circumvent this difficulty, we present in Chapter 7 a metaheuristic algorithm that is inspired by [90] - for the learning of monotone preferences - in order to learn MR-Sort models with single-peaked preferences. First, we propose two variants : one that learns simultaneously and the other successively the two specific values of single-peaked criteria. Second, we carry out some experiments with synthetic data and datasets from a well-known repository (UCI repository). The latter enables us to easily compare our proposals with existing algorithms (in particular, machine learning algorithms).

We conclude the document with Chapter 8 by summarizing our contributions. We recall the proposed algorithms, the results obtained, and the connections between them. Also, we give some relevant perspectives to our work.

Chapter 2

Background

Contents

2.1 Introduction

In this chapter, we give to the reader the prerequisites in the Multiple Criteria Decision Analysis (MCDA) field, which are useful to apprehend our contributions in the following chapters.

First, we give the definitions of the main concepts in MCDA tackled in this thesis (Section 2.2). Then, we explore the methodologies related to the field of MCDA, and particularly sorting models (Section 2.3). Finally we briefly focus on the field of Preference learning (PL) (Section 2.4) and its links with MCDA in the context of learning MR-Sort parameters with non monotone data.

2.2 Main concepts in MCDA

As the name evokes, Multiple Criteria Decision Analysis involves many criteria, and can be described as different mechanisms for making "the best" possible decision regarding a compromise between various criteria. A Multiple Criteria decision problem generally involves 2 main types of stakeholders :

- the Decision Maker (DM) : he is the central actor in a decision process, since he is the person who is confronted with a decision. As the response to his problem must be personalized, the DM has to provide at least some hints about his preferences.
- the Decision analyst (DA) : he is the guide of the DM. Not only he helps the DM in expressing appropriately his preferences, but he also helps him in making the best decision.

In some case, there are several DMs instead of one [27, 43] ; therefore the DA has some options in order to probe the opinions of the DMs as a single interlocutor : for instance getting a consensus for each preference expression, or consulting each stakeholder throughout the decision process. However, in our work, we consider the case with only one DM.

In the following section, we describe some fundamentals of MCDA. First we describe alternatives and criteria. Then we define preference information and preferences relations. After that, we explore the types of problems in MCDA. Finally, we describe aggregation and disaggregation paradigms.

2.2.1 Basic tools of the MCDA

In the MCDA field, an alternative refers to actions or options available for the DM given a decision to make. Each alternative is distinct from others and is assessed on several criteria.

A criterion represents a point of view of the alternatives. It can be seen as the judgement or the performance of an alternative focusing on a single aspect. There are generally two types of criteria regarding the measurement used : quantitative criteria (based on numerical values) or qualitative criteria (based on non-numerical values).

An alternative a is evaluated by a value a_i with $a_i \in X_i$, where X_i is called the evaluation scale of criterion i.

The set of criteria is $\mathcal{N} = \{1, ..., i, ...n\}$. An alternative a can be represented as a vector of evaluations of a on criteria, i.e. $a = (a_1, ..., a_i, ..., a_n)$. Alternatives can also be expressed without explicit evaluations, i.e. expressed with pairwise comparisons of alternatives values.

In our context, alternatives are assessed by several criteria, where each criterion corresponds to a value in the criterion's evaluation scale. We note the Cartesian product of evaluation scales as $X = \prod_{i \in \mathcal{N}} X_i$. We also denote $A = \{a^1, ..., a^j, ..., a^m\}$ the set of alternatives. Therefore A is the set of alternatives on which the DM expresses his preferences.

2.2.2 Preference information and preference relations

In order to aid the DM, the DA probes useful information to build a representation of the DM's preferences. These preferences can be expressed in one of the following ways :

- pairwise comparisons : there are four types of binary relations for comparing alternatives between themselves.
    1. The weak preference ($\succsim$) is an preorder on A (which is a reflexive and transitive binary relation).
    2. The strict preference ($\succ$) which is the asymmetric part of $\succsim$.
    3. The indifference relation ($\sim$) which is the symmetric part of $\succsim$.
    4. The incomparability relation (#).
- evaluation of alternatives : either qualitative values, quantitative values or alternatives assignments into categories.
- sets of constraints formulated in human language : for instance taking the form of conditions on criteria values, etc.

In order to achieve such a task, the DM needs to provide a set of information by means of historical preferences or on-purpose collected data that is called "reference set" (or learning set). Indeed, the elicitation process consists of the DA collecting these DM's preferences information.

2.2.3 Types of problems in MCDA

Roy [80] introduced essentially three types of decision problems that one can deal with in Multiple Criteria Decision Analysis.

1. **Choice problem** : given a list of objects (alternatives) the goal is to select an alternative or a group of alternatives that suits the best to the DM's preferences (see the first problem in Example 2.1).

2. **Sorting problem** : this problem considers a set of ordered and predefined categories. The aim is to classify alternatives into these categories with respect to the preferences of the DM (see the second problem in Example 2.1). We note $c(a)$ the category of alternative a.

3. **Ranking problem** : this problem consists in constructing an order on the alternatives such that it reflects the preferences of the DM (see the third problem in Example 2.1).

Example 2.1 *A teacher of 8 students in final year of the high school is confronted with three problems.*

The first problem is a choice problem. The teacher wants to select two skilled students in order to attend to an olympiad contest in mathematics. In order to build the team, he relies on three requirements : proficiency in Maths, but also in Language - because he thinks that language skills are necessary - and the cohesion of the two students.

The second problem is a ranking problem. As a form tutor (a teacher in charge of a class), he has to establish a ranking of the students for the end of the academic year which is expected by students as they are rewarded according to their worth. Moreover, students like comparing between themselves.

The third problem is a sorting problem. The final exam is close, the teacher did a mock exam in order to foresee the readiness of students for the exam. As he intends to plan some tutoring for needy students, he decides to break students in three categories : good students (G) - good student that need no help because they are well-prepared for the exam - , average students (A) who need a light tutoring and bad students (B) who need intensive tutoring.

The following Table 2.1 summarizes the marks of students obtained at the mock exam. We consider marks (graded between 0 and 20) attributed to the 8 students (alternatives) in three subjects (criteria) "Language", "Maths", and "History".

Our work concerns sorting problems. More precisely we consider the inverse sorting problem, where a set of alternatives sorted by the DM is given as an input of the problem. Following the aggregation and disaggregation methodologies, we can efficiently process the information in order to infer models that represents of the DM's preferences.

	Maths	*Language*	*History*	$c(a)$
a_1	11	17	9	G
a_2	18	14	7	G
a_3	8	15	10	A
a_4	6	10	13	A
a_5	9	7	20	A
a_6	15	4	10	A
a_7	5	10	9	B
a_8	9	7	5	B

Table 2.1: Assignment examples : dataset of 8 students and 3 subjects

2.2.4 Aggregation and disaggregation paradigms in MCDA

In MCDA, the aggregation of alternatives consists in representing an holistic view of these alternatives. In other words, it consists of constructing a single measure (which is a compromise of the evaluations on criteria) in order to evaluate alternatives. By extension the aggregation of preferences enables to represent a more compact expression of these preferences.

The disaggregation principle [57] is the reverse process of the aggregation. It consists of inferring a representative model of the DM from examples of the DM preferences. In others words, we aim at learning the model parameters that sketches the preferences of the DM.

In the context of MCDA, we count two types of elicitation of decision model parameters :

- **Direct elicitation** : it consists of asking the DM to express directly the model parameters of his preferences in an interactive manner.

- **Indirect elicitation**: in this case, the DM does not need to provide himself the parameters of the decision model. Instead, past preferences of the DM are considered for the elicitation process. It aims at building the most appropriate parameters of the DM decision model i.e the parameters that are compatible with DM preferences. The indirect method was developed because of the difficulties encountered with the direct elicitation (in case of direct elicitation, it is very difficult for the DM to directly provide his preferential parameters) [84, 83].

In our study, we work under the paradigm of indirect elicitation.

2.3 Methodologies in MCDA

The question of the choice of appropriate approaches draws some attention to the MCDA community. In [103], the authors notice 56 models in MCDA, and propose a selection tool to choose an appropriate method in order to solve a given problem.

They proposed a decision tree approach to guide someone who wants to choose an appropriate model for his problem.

Among the plethora of methodologies in MCDA, we distinguish three main families of methodologies [57] : additive models, outranking models and rule-based models.

Additive models are based on the aggregation of alternatives into a score. Several methods such as Utilité Additive (UTA) [56], Analytic Hierarchy Process (AHP) [82], and others use this principle.

Outranking models were introduced by Roy [80]. They are based on pairwise comparisons between alternatives. These models are more recent; we can cite PROMETHEE [21] and ELECTRE [104], which have several variants.

The development of concepts such as Rough set and Fuzzy set theories, combined with MCDA have lead to recent interesting family of models : rule-based decision models [48].

2.3.1 Additive models

There exist several additive models such as UTA [56], MACBETH [5, 6], and AHP [82].

In the following, we choose to describe UTA since a majority of models come from UTA and some existing works on the matter are related to sorting problems. We also describe the MACBETH model. As we are interested in partially monotone preferences, MACBETH has been studied in the context of monotone preferences [7, 77] and UTA with both monotone and non monotone preferences [31, 60, 35].

Utilité Additive (UTA)

The UTA (which takes its name from the french noun "UTilité Additive") method was first introduced by Jacquet-Lagrèze [56] where it was used in ranking problems. UTA in the context of MCDA originates from the theory of multi-attribute value function (also called MAVT). Commonly illustrated as a simple weighted sum, this concept aims at aggregating several view points into a single evaluation.

On each criterion i, a performance score a_i is attributed to an alternative a and used to compute a marginal value through a function called marginal value function or the utility function of the criterion i noted u_i.

Marginal values function u_i is akin to a mapping from a real scale measurement to a simple, anonymized scale which is often the unit interval.

Assuming that u_i is monotone non-decreasing, thus for a criterion i, if an alternative a is preferred over (better than) an alternative b, then a_i is greater or equal than b_i.

The UTA method uses a global utility which is a weighted sum of the criteria marginal values. Thus, the global utility function is :

$$u(a) = \sum_{i=1}^{n} w_i u_i(a) \tag{2.1}$$

Let us note that w_i is the weight of criterion i and that the global value function is a monotone function.

The method is defined on a preorder ($\succcurlyeq$) of a set of alternatives. Therefore, the UTA model makes it simple to compare alternatives between them. We have the following equivalences :

$$\begin{cases} a^j \succ a^{j'} \iff u_i(a^j) > u_i(a^{j'}), \forall a^j, a^{j'} \in A \\ a^j \sim a^{j'} \iff u_i(a^j) = u_i(a^{j'}), \forall a^j, a^{j'} \in A \end{cases} \tag{2.2}$$

The UTA method as described above presents various extensions. We can cite some : UTASTAR [87], ACUTA [16, 86], and a stochastic version of UTA in [85].

An adaptation of UTA to sorting alternatives named UTADIS has also been developed by [32].

The aim of UTADIS is to construct linear interpolations of the marginal utility values by minimizing the misclassification rate (difference between the DM assignments and the categories inferred by the model).

Considering the global utility function u, an alternative a, the marginal values u_h, $h \in \{1, ..., p\}$ (p the number of categories), the classification rule takes the following form :

$$\begin{cases} u_h \leq u(a) \leq u_{h-1} \Longrightarrow a \in C^h \\ u(a) \leq u_1 \Longrightarrow a \in C^1 \\ u(a) \geq u_{p-1} \Longrightarrow a \in C^p \end{cases} \tag{2.3}$$

MACBETH

In the MACBETH approach [5, 6], the DM is asked to provide qualitative and quantitative information concerning his preferences. On the one hand, with qualitative information, we can loosely compare the alternatives with each other. On the other hand with quantitative information, it is possible to express the intensity of differences between preferences. The principle of the method is grounded on

6 degrees of pairwise comparisons called *difference of attractiveness* : very weak, weak, moderate, strong, very strong, extreme. Each pair of alternatives compared by the DM is evaluated with one of the degrees. These degrees are reported on a scale and represents contiguous intervals. Six numeric values called *thresholds* separate these intervals from each other and are determined during the construction of the MACBETH model.

Therefore, as in the case of UTASTAR, MACBETH models are constructed with subintervals of evaluation scales. In addition, the approach aims at approximating by piecewise linear functions, the global additive function of the model. The marginal function values can be retrieved thanks to some reference points. These reference points are given by the DM and represent alternatives the DM judges absolutely satisfying, unsatisfying or neutral. In the MACBETH approach, the knowledge of only two reference points can help to build the marginal values.

The MACBETH approach is also known as an interactive approach and very suitable to real-world application, not only because the DM can possibly revise its judgments, but also the layman can easily understand the *difference of attractiveness*.

2.3.2 Outranking models

The outranking models are based on pairwise comparisons of alternatives, where alternatives are sorted in p predefined and ordered categories, i.e. $C^p \triangleright \ldots \triangleright C^2 \triangleright C^1$ (with C^p the best category and C^1 the worst category, $\triangleright$ that denotes the order between categories). In our work, we are interested by this sort of models.

ELECTRE TRI method

ELECTRE TRI was introduced by Yu [104]. It is based on the comparison of alternatives with fictitious alternatives called *profiles* which delimit the categories. More formally, we describe an evaluation scale X_i for each criterion i. Therefore the global domain of evaluations is the Cartesian product of X_i which is $X = \prod_{i \in \mathcal{N}} X_i$ with $\mathcal{N}$ the set of criteria. ELECTRE TRI considers p predefined and ordered categories, and the profiles $(b^{p-1}, ..., b^1)$ that delimits the frontiers of the categories $C^p, \ldots, C^1$.

In order to assess the comparisons between alternatives, the method refers to scores called *credibility indices*. This index is made of two components : the concordance index and the discordance index.

First, the concordance index characterizes the strength in accord with the statement to be evaluated. For instance $\mathcal{C}(a, b)$ is the concordance index for the statement "a is preferred to b". This index is an aggregation of the concordance

index of each criterion noted $\mathcal{C}_i(a,b)$. In order to introduce the expression of $\mathcal{C}_i(a,b)$ we introduce the following parameters :

- the indifference threshold, which is q_i^h, with $q_i^h \geq 0$, $h \in \{1,\ldots,p\}$, $i \in \mathcal{N}$.
- the preference threshold, which is r_i^h, with $r_i^h \geq q_i^h$, $h \in \{1,\ldots,p\}$, $i \in \mathcal{N}$.

We can express $\mathcal{C}_i(a,b)$, which is a value in [0,1] as the following :

$$\begin{cases} \text{If } a_i \leq b_i - r_i^h, \text{ then } \mathcal{C}_i(a,b) = 0, \\ \text{If } a_i \geq b_i - q_i^h, \text{ then } \mathcal{C}_i(a,b) = 1, \\ \text{Otherwise, } \mathcal{C}_i(a,b) = \frac{a_i - b_i + r_i^h}{r_i^h - q_i^h} \end{cases} \tag{2.4}$$

The concordance index reads as follows :

$$\mathcal{C}(a,b) = \sum_{i \in \mathcal{N}} \mathcal{C}_i(a,b) \tag{2.5}$$

On the other side, the discordance index (noted $\mathcal{D}(a,b)$) plays a counter effect regarding the role of concordance index. It consists of characterizing coalitions of criteria that are opposed to the preference statement. For instance $\mathcal{D}(a,b)$ which is the discordance index between a and b describes the strength of arguments against the statement "a is preferred to b". In order to compute this value, we need to calculate $\mathcal{D}_i(a,b)$ for a given criterion i. An additional parameter which is v_i^h, $h \in \{1,\ldots,p\}$, the veto threshold is used. We can compute the values of $\mathcal{D}_i(a,b)$ as follows :

$$\begin{cases} \text{If } a_i \geq b_i - r_i^h, \text{ then } \mathcal{D}_i(a,b) = 0, \\ \text{If } a_i \leq b_i - v_i^h, \text{ then } \mathcal{D}_i(a,b) = 1, \\ \text{Otherwise, } \mathcal{D}_i(a,b) = \frac{b_i - r_i^h - a_i}{v_i^h - r_i^h} \end{cases} \tag{2.6}$$

Finally we express the credibility index as the following product :

$$\sigma_i(a,b) = \mathcal{C}(a,b) \prod_{i \in \mathcal{N} : \mathcal{D}_i(a,b) > \mathcal{C}_i(a,b)} \frac{1 - \mathcal{D}_i(a,b)}{\mathcal{C}_(a,b)} \tag{2.7}$$

Thanks to the value obtained with $\sigma_i(a,b)$, it is possible to compare a and b using a given majority threshold λ. Therefore, if $\sigma_i(a,b) > \lambda$, we deduce that a is preferred to b.

In order to sort an alternative a in the appropriate category, a is compared successively with limit profiles. Therefore, two orders of pairwise comparisons are possible leading to two types of procedures : the optimistic procedure and the pessimistic procedure (which is the most commonly used). The latter procedure is the following, assuming we desire to sort an alternative a. First, we consider

the decreasing order of profiles i.e (b^{p-1}, ..., b^1). By comparing successively the alternative a with each profile of this order, we stop at the first comparison such that this alternative is preferred to the current profile, (let's say b^k). Therefore a is assigned to C^{k+1}. If such case does not happen throughout the comparisons, then the alternative a is assigned to category C^1. The optimistic rule adopts the opposite order of pairwise comparisons and the rule is different : the alternative a is assigned to a category of which a does not strictly outrank the lower limit profile of the category.

There are several variants of the ELECTRE TRI method. We can cite ELECTRE TRI-C [81], which differs from ELECTRE TRI in the fact that central profiles are used instead of limit profiles.

The Stochastic Multicriteria Acceptability Analysis (SMAA) method, initially reserved to ranking problems, has also been adapted to ELECTRE TRI giving birth to SMAA TRI [45]. The principle is to assess the sensitivity of an MCDA method through a Monte Carlo simulation based on finite spaces of arbitrarily distributed parameter values. Thus, SMAA TRI computes the category acceptability - which is the chance for an alternative to be assigned to a category given a possible set of ELECTRE TRI parameters.

The Non-Compensatory Sorting (NCS) model

The Non-compensatory Sorting (NCS) [17, 18] model is an MCDA outranking model that stems from the ELECTRE TRI method [44]. It aims at sorting alternatives into p predefined ordered categories $C^p \triangleright \ldots \triangleright C^2 \triangleright C^1$ (with C^p the best category and C^1 the worst category).

We briefly recall that $a = (a_1, \ldots, a_n) \in X = \prod_{i \in \mathcal{N}} X_i$ with X_i the evaluation scale of criterion i. The outranking relation between alternatives is defined by a preference order $\succcurlyeq_i \subset X_i \times X_i$ for each criterion $i \in \mathcal{N}$. It is defined - from the viewpoint of monotone preferences - as the following :

- i is a "*profit*" criterion if : $a_i \succcurlyeq_i a'_i$ iff $a_i \geq a'_i$, $a_i, a'_i \in X_i$,
- i is a "*cost*" criterion if : $a_i \succcurlyeq_i a'_i$ iff $a_i \leq a'_i$, $a_i, a'_i \in X_i$,

The NCS model incorporates $p-1$ limit profiles $(b^1, \ldots, b^{p-1})$ which delimit the categories between them. We have $b^h = (b^h_1, ..., b^h_n)$, $\forall h \in \{1, ..., p-1\}$. We additionally consider two fictive profiles b^0, the lower limit of category C^1 and b^p the upper limit of category C^p.

Therefore, the NCS method is the following : an alternative a belongs to the category C^h if *(i)* it is better than the lower limit of the category on a sufficiently strong subset of criteria, and *(ii)* this is not the case when comparing the alternative to the upper limit of the category.

In other words, in order to be sorted in category C^h, the set $\{\forall i \in \mathcal{N} : a_i \succcurlyeq_i b_i^{h-1}\}$ must form a "majority" whereas $\{\forall i \in \mathcal{N} : b_i^{h-1} \succcurlyeq_i a_i\}$ does not form a "majority". We describe more in details the NCS method in the two cases.

Sorting with two categories. Let us focus first on a simple case which is the sorting of alternatives into 2 categories : Good ($\mathcal{G}$) and Bad ($\mathcal{B}$). We denote $\mathcal{A}_i \subseteq X_i$ the set of approved values on criterion $i \in \mathcal{N}$.

An approved value a_i on criterion i ($a_i \in \mathcal{A}_i$) is a value that counts in the assignment of the alternative a to category $\mathcal{G}$. In order to assign alternative a to category $\mathcal{G}$, the alternative a should have approved values on a subset of criteria which forms a "majority". The set of "majorities" is also called the family of "sufficiently strong" subsets of criteria denoted by $\mathcal{F} \subseteq 2^{\mathcal{N}}$. Therefore, we write the NCS assignment rule using the previous notations :

$$x \in \mathcal{G} \quad \text{iff} \quad \{i \in \mathcal{N} : x_i \in \mathcal{A}_i\} \in \mathcal{F}, \quad \forall x \in X \tag{2.8}$$

Sorting with more than two categories. In this setting, we consider an ordered set of p categories. The sets of approved values represents $\mathcal{A}_i^h \subseteq X_i$ with $(h = 1, \ldots, p-1)$ such that $\mathcal{A}_i^1 \supseteq \mathcal{A}_i^2 \supseteq ... \supseteq \mathcal{A}_i^{p-1} \supseteq \mathcal{A}_i^p$, with $\mathcal{A}_i^1 = X_i$. Families of "sufficiently strong" subsets of criteria are also nested ; we have : $\mathcal{F}^1 \supseteq \mathcal{F}^2 \supseteq ... \supseteq \mathcal{F}^{p-1} \supseteq \mathcal{F}^p$, where $\mathcal{F}^h$ is the set of majorities pertaining to the sorting of alternatives into the category C^h. In particular, we have $\mathcal{F}^1 = \mathcal{P}(\mathcal{N})$, the set of the subsets of $\mathcal{N}$. Taking into account multiple categories, the assignment rule is the following for all $x \in X$:

$$\begin{aligned} x \in C^h \quad \text{iff} \quad \{i \in \mathcal{N} : x_i \in \mathcal{A}_i^h\} \in \mathcal{F}^h \text{ and } \{i \in \mathcal{N} : x_i \in \mathcal{A}_i^{h+1}\} \notin \mathcal{F}^{h+1}, \\ \forall h \in \{1, ..., p-1\} \end{aligned} \tag{2.9}$$

$$x \in C^p \quad \text{iff} \quad \{i \in \mathcal{N} : x_i \in \mathcal{A}_i^p\} \in \mathcal{F}^p \tag{2.10}$$

Majority Rule Sorting (MR-Sort)

The aim of the Majority Rule Sorting method (MR-Sort) is still to assign alternatives into a predefined order of categories.

MR-Sort is a special case of NCS, where the majorities are only additive.

In fact, the notion of majority is formalized using weights $(w_1, ..., w_n)$ attached to criteria (with $w_i \geq 0$, $\forall i$, and $\sum_{i \in \mathcal{N}} w_i = 1$), and a majority threshold $\lambda \in]0.5; 1]$. The subset of criteria $\mathcal{I} \subseteq \mathcal{N}$ is a majority iff $\sum_{i \in \mathcal{I}} w_i \geq \lambda$. $\mathcal{I}$ is also called a

winning coalition of criteria. Finally, the MR-Sort rule can be expressed as follows:

$$c(a) = C^h \Leftrightarrow \sum_{i:a_i \succcurlyeq_i b_i^{h-1}} w_i \geq \lambda \text{ and } \sum_{i:a_i \succcurlyeq_i b_i^{h}} w_i < \lambda \tag{2.11}$$

In other words, MR-Sort assigns alternative $a \in X$ to a category C^h, $h \in \{1, \ldots, p\}$ (denoted $c(a) = C^h$), when the set of criteria for which a_i is better than the lower profile of C^h ($a_i \succcurlyeq_i b_i^{h-1}$) forms a majority, but the set of criteria for which a_i is better than the upper profile of C^h ($a_i \succcurlyeq_i b_i^h$) is not a majority.

Example 2.2 *Let us consider the Example 2.1. Now, we want to represent the model underlying the sorting problem. We recall the table of assignment examples in Table 2.1. Let us assume in Figure 2.1 the graphical representation of the model describing the preference of the teacher. The three vertical axes represent the three criteria ; each is graduated from 0 to 20 which is the limits of the evaluation scales of criteria. Let us consider the model defined by* $b = (b_1, b_2)$ *(pictured in Figure 2.1 with blue lines) and the triplet of weights* $w = (w_1, w_2, w_3)$*, and the majority threshold* λ*. We choose to draw only four alternatives (*a_1*,*a_4*,*a_6*,*a_8*) in Figure 2.1 ; each alternative is represented by a color that represents its category (green for category G, orange for category A and red for category B).*

In Figure 2.2, we show another representation of the model based on coalitions of criteria. Yellow nodes are sufficient coalitions of criteria whereas the others are not. In other words, in order to classify an alternative in a category, this alternative must be preferred to the lower limit profile of such category on coalitions of criteria represented here with yellow nodes (i.e $\{c_1, c_2\}$*,* $\{c_1, c_3\}$ *and* $\{c_1, c_2, c_3\}$*).*

	$Maths(c_1)$	$Language(c_2)$	$History(c_3)$	$c(a)$
a_1	11	17	9	G
a_2	18	14	7	G
a_3	8	15	10	A
a_4	6	10	13	A
a_5	9	7	19	A
a_6	15	4	10	A
a_7	5	10	9	B
a_8	10	6	5	B

Table 2.2: Assignment examples : dataset of 8 students and 3 subjects

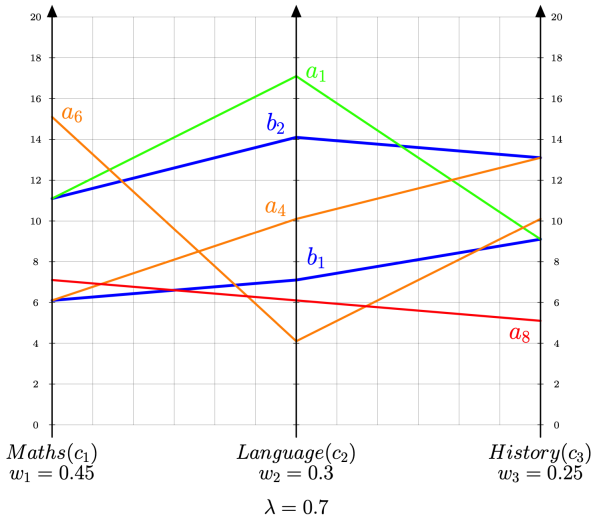


Figure 2.1: Graphical representation of profiles and alternatives

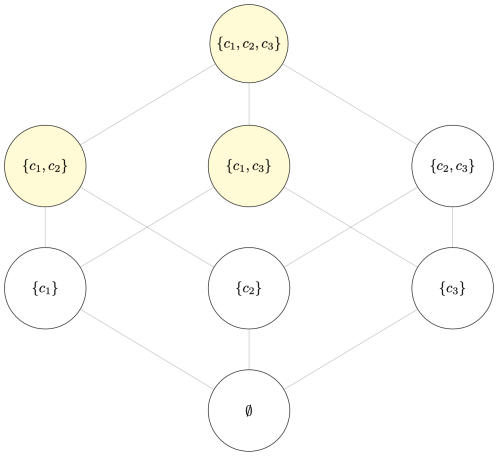


Figure 2.2: Lattice representing sufficient coalitions of criteria

2.3.3 Rule-based models

This domain deals with decision models that can be represented following the scheme of a decision rule. In this context, a decision rule can be formulated in the following manner : *If ... Then ....* A well known method using rules is the Dominance-based Rough Set approach (DRSA) [48] that is used for multiple criteria models. This approach was inspired by the field of Rough Set analysis which tends to establish some dependencies between attributes in multiple criteria sorting problems.

One advantage of this Rough Set method resides in its ability to capture some

hesitations of the DM thanks to its structure. DRSA can be summarized as an approach that separates assignments into two groups, consistent ones (with regards to the DM preferences) and inconsistent assignments (which can be considered as DM's *hesitations*).

DRSA approach

There are some analogies between terms generally used in the literature for the DRSA approach and MCDA terms - that we highlight in the following. Let us consider an *information table* which is expressed with 4 tuples $< \mathcal{A}, Q, V, f >$, where $\mathcal{A}$ is the set of objects (alternatives in MCDA), Q the set of attributes. The set of attributes is divided into two disjoint non-empty sets $\mathcal{N}$, the set of condition attributes (criteria in MCDA) and D, the set of decision variables (only one decision variable in MCDA whose values match categories). The value set of attribute $q, q \in Q$, is represented by V_q and we have $V = \cup_{q \in Q} V_q$. Here, sets of evaluations ($\forall i \in \mathcal{N}$) are similar to evaluation scales in MCDA. In the table, the alternatives are in rows, and the attributes are in columns. Finally the *information function* refers to the function f that maps the tuple of value (x, q) to a value in V_q (i.e $f(x, q) \in V_q$).

In order to compare objects, let us assume that each attribute is endowed with a preference relation $\succcurlyeq_q$ which is a complete preorder. The relation $x \succcurlyeq_q y$ means that x is at least as good as y regarding attribute q, thus we have $f(x, q) \geq f(y, q)$.

For the sake of simplicity and in order to relate to our work, we consider only one decision attribute that determine predefined and ordered classes of objects, that are $\{C^1, ...C^h, ..., C^p\}$ (which are categories in MCDA).

The DRSA approach integrates the *dominance principle* (also known as Pareto principle) [48, 49], as defined in the following :

Definition 2.1 *The principle of dominance implies that if an object x dominates y on all the attributes, then x should be classified in a class that is at least as good as the class of y.*

The concept of Rough Set enables to build sets of classes that are considered as upward and downward approximations of alternative classes. Therefore in the DRSA approach, assignment rules of alternatives can be formulated on the basis of approximation sets. Typically, each alternative x is assigned to set of contiguous categories $[C^l, ..., C^u]$, where C^l (resp. C^u) is the lower (resp. higher) category of the set. An alternative can possibly be assigned to a single category (i.e $C^l = C^u$).

Several induction algorithms can be applied in order to generate rules, and thereby alternative assignments. We can cite DOMLEM [50], MODLEM [50], and LEM2 [52].

In the next section, we talk about Preference learning (PL) which is a field that has relevant links with the MCDA field, especially in the context of learning decision models. We shortly describe the field, establish the relations between MCDA and PL, especially some motivations inspired by PL to tackle the learning of MR-Sort models with non monotone data (see Chapter 5).

2.4 Preference Learning

The field of Preference Learning (PL) is a subfield of Machine Learning [47]. It aims at learning preference models from empirical data in order to predict user preferences. Originally, PL was mostly studied for ranking tasks.

Nevertheless this interdisciplinary field has several links with other fields such as Knowledge representation [2], Data mining [101], Recommender Systems [47], and more generally Operations Research [28] and MCDA [28]. In the next subsections, we briefly describe some sorting problems in this field. Then, we show the links with MCDA and how that lines up in our work. In fact in [88], Sobrie used some PL practices in order to design an experimental setting for learning MR-Sort models.

2.4.1 Classification problems in Preference Learning

There are several types of classification problems in this field. We can cite two of them : multi-label classification problems [96] and graded multi-label classification problems [26]. The former problem deals with classification of instances into unordered labels (categories). In addition each instance can be assigned to several labels. The graded multi-label classification is a generalization of the multi-label classification where each assignment is attached to a grade, which is a degree of membership of an instance to a label. Tehrani *et al.* [42] consider the use of discrete Choquet integral (mostly used in MCDA) as an component for machine learning tasks in classification problems.

Hullermeier. et *al* [47] describe four paradigms when tackling these problems which mainly depend on the structure of the data : utility functions, preference relations, specific preference models and local aggregation of preferences.

Several other problems can be counted as Preference Learning problems such as : (multi) label ranking, instance ranking, object ranking, (graded) multi-label ranking, collaborative filtering. The field of Preference Learning (or more broadly Machine Learning) provides interesting practices to assess the performance of the learning algorithms used to solve MCDA problems particularly sorting problems.

Classification accuracy and evaluation process of learning algorithms

Let us consider a set of alternatives A assigned to some categories. A model $\mathcal{M}$ represents the ground truth, i.e $\mathcal{M}$ is an "oracle" used to assign the elements of A into their respective categories. A learned model $\mathcal{M}'$ is an approximation of $\mathcal{M}$ obtained through a learning process. The category stemming from the decision rule applied to an alternative $a, a \in A$ with the model $\mathcal{M}$ (resp. $\mathcal{M}'$) is represented by $(cat(a, \mathcal{M}))$ (resp. $(cat(a, \mathcal{M}'))$).

We describe the Classification Accuracy (CA) as the following :

$$CA = \frac{|a \in A : cat(a, \mathcal{M}) = cat(a, \mathcal{M}')|}{|A|}. \tag{2.12}$$

The classification accuracy is the rate of restoration of assignment examples from a learned model.

Unlike in PL, where the evaluation of models are based on three phases (training, validation and test phases), in our work as we deal with MCDA models, we consider training and validation phases as a single phase and called it validation phase (through misuse of language). Therefore, we distinguish two rates : CA_v which is the classification rate in validation phase and CA_g the classification rate in generalization. The difference resides in the type of dataset used to calculate the score. For CA_v, the set A is the learning set, whereas for CA_g, A represents the test set. In the Machine Learning literature, the classification error (also called the $0/1$ loss) generally denotes the rate obtained by $1 - CA$.

2.4.2 At the crossroad between MCDA and PL

We highlight some useful connections and differences between MCDA and PL in the following. First, the size of the problems in PL is generally larger than in MCDA. Nevertheless, the ability to handle large datasets comes with a computational cost. The expressivity of models in MCDA offers more tools for the interpretability of the learned models [70, 89, 9].

Second, MCDA methodology is more user-oriented than preference learning tools. The interactions between the Decision Maker (DM) and the Decision Analyst (DA) are encouraged, and the MCDA literature has tackled various problems related to interactions during the learning process [73, 33]. On the contrary, PL is more focused on the performance of learning tasks as in [38].

Finally, PL tackles a broader spectrum of problems, thanks to its capability to deal with non monotone and unknown data structures [25, 102]. Up to now, MCDA methods like NCS and MR-Sort used to treat problems assuming that the attributes of the data are monotone.

Inspired by problems encountered in PL, we intend to push the limits of these models by tackling the learning of sorting models with non monotone data (in chapters 5, 6 and 7). Moreover, as in PL [42] we investigate MR-Sort learning problems with large dataset sizes.

2.5 Conclusion

In this chapter, we introduced the main principles of MCDA. We discussed problems and models in this field. In the remaining of our work, our interest is oriented towards sorting problems in MCDA, in particular learning MR-Sort models from "large" datasets and non monotone data inspired by the Preference Learning field.

In the next chapter, we review the literature related to monotone, partially monotone, and non-monotone preferences in the context of ordinal classification. We also discuss some sorting problems with MR-Sort models.

Chapter 3

Related work

Contents

3.1 Introduction

In this chapter, we aim at reviewing the literature but particularly problems and algorithms concerning different aspects of our work.

First, we do a brief introduction on ordinal classification in the next subsection 3.1.1.

Secondly in Section 3.2, we explore existing problems and algorithms related to monotonicity on preferences in Multicriteria Decision Analysis (MCDA), Preference Learning and related fields.

The last part, Section 3.3 is dedicated to the related work around MR-Sort; we show the different problems and approaches based on models close to MR-Sort. We also review different works done in the context of the inference of MR-Sort models and present the algorithm on which some of our contributions are based.

3.1.1 Ordinal classification

In decision problems, classification is the generic term used to refer to the assignment of alternatives into predefined categories (not necessarily ordered).

However, sorting - mostly used in the MCDA community - generally denotes the assignment of alternatives in predefined and ordered categories. The term used beyond the field of MCDA is ordinal classification (also known as ordinal regression), which describes the task of assigning objects into ordered classes (which are equivalent to categories in MCDA).

The question of ordinal classification has interested the artificial intelligence and expert systems communities since their early days [66, 4].

There exists different taxonomies for ordinal classification problems regarding several features [54, 55, 93] : the type of the problem, the nature of the input data, the type of model and the type of algorithm used to solve the problem. Inevitably, all these characteristics are intimately connected. On the one hand, the problem defines the types of data handled, the outputs expected by the problem, and therefore the specific metrics in order to evaluate the solutions of the problem. On the other hand, the type of model chosen in order to solve the problem implies the use of restrained resolution methods and techniques.

In the next section, we deal with the specificity of monotone and non-monotone classifications. We talk about ordinal classification problems in a large sense including not only the fields of MCDA and Preference Learning but also other related fields.

3.2 Monotone, partially monotone and non-monotone preferences

With monotone preferences, the improvement of a performance of an alternative does not contribute to a worse assignment of this alternative into the categories ; this is not always the case with other types of monotonicities.

In this section, we explore the literature related to monotone, partially monotone and non-monotone preferences in MCDA. We briefly review some works in data mining and some supervised learning fields.

In their review works, [24, 54, 55] outlined the connections between ordinal and monotone classification. Indeed, monotone classification is a special case of ordinal classification and it has been a growing body of research on this topic.

Monotone classification relates to the monotone dependency between the values of attributes and classes : either the increase of attributes values contribute to an assignment in the same or an upper class (monotonically increasing relationship) or contribute to assignment in the same or a lower class (monotonically decreasing relationship). We call a *preference direction*, that relationship (monotonicity) with regards to a given attribute and the classification : either increasing or decreasing. Formally, we introduce the following definition regarding multicriteria sorting problems [92]:

Definition 3.1 *A sorting function is monotone if an alternative x cannot be assigned to a less preferred category than an alternative y whenever x is at least as good as y on all attributes. In other words, given $\succcurlyeq_i$ the preference order on attribute i, for all $x, y \in A$, with $x_i \succcurlyeq_i y_i$, $\forall i \in \mathcal{N}$, we have $c(x) \rhd c(y)$ or $c(x) = c(y)$.*

Therefore monotone preferences are characterized by alternatives assigned by such a monotone sorting function.

Unfortunately, it is not obvious to straightforwardly describe non-monotone preferences. They relate to a spectrum of kind of preferences, from partially monotone preferences to totally non-monotone preferences. In [76], partial monotonicity is described as a monotonicity restrained to a subset of $\mathcal{N}$.

Definition 3.2 *Given, $B \subseteq \mathcal{N}$, and two alternatives x and y, the partial ordering $\succcurlyeq_B$ is defined as $x \succcurlyeq_B y \Leftrightarrow x_i \succcurlyeq_i y_i$, $\forall i \in B$. Partial monotonicity constraints are defined as : if the alternative x is not worse than y regarding B (or $x \succcurlyeq_B y$), then $c(x) \rhd c(y)$ or $c(x) = c(y)$.*

In partially monotone classification, only attributes of the subset B have monotone relationships with classes.

In MCDA, the question of order is important. The evaluation scales X_i, $(i \in \mathcal{N})$ are set of values, which are performances that are necessarily ordered. Therefore, criteria are considered as ordinal attributes.

In [14], monotonicity is tackled in terms of ordinal attributes and non-ordinal attributes. They distinguish five types of classification problems depending on the presence or absence of monotonicity relationships between attribute values and assignments classes. Two of the five problems are ordinal problems and deal with ordinal attributes. They can be summarized in the following two groups :

- *(i)* monotone classification : (which includes both Definition 3.1 and Definition 3.2) where at least one attribute is ordinal, classes are ordered, and there is a monotone relationship between values of ordinal attributes and classes.

- *(ii)* non-monotone classification : (which also includes Definition 3.2) where at least one attribute is ordinal, classes are ordered, but there is no monotone relationship between values of ordinal attributes and classes.

In our work, we are interested in both cases, but in particular in the latter case. As mentioned in [14], for *(ii)*, although no global monotone relationship can be found between values of ordinal attributes and classes in this case, this type of classification can suggest local monotonicity relationships. In the following we formally describe the single-peaked principle and conclude that ordinal attributes endowed with this principle express two different local monotonicity relationships.

The single-peaked principle

Single-peaked preferences were introduced first by Black in 1948 [12, 13]. He used this type of preference in economy to describe an idiosyncratic behaviour of committee members desiring to adopt a resolution in front of several motions. He illustrated this through a single-peaked curve which is a curve that changes its direction at most once, from up to down.

Nowadays, the notion is still abundantly studied in several domains, mainly in Social Choice field. We consider the following definition according to Escoffier *et al.* [41] (we only use these notations for only this part to recall the definition of the single-peakedness in it original field, which is Social Choice):

Let us consider a set of voters $V = \{1, \ldots, m\}$, and a set of candidates $X = \{x_1, \ldots, x_n\}$ with $n \geq 3$. A preference relation (noted $\succ$) on X is a linear order on X. The peak of the preference relation $\succ$ is the candidate $x^\star = \text{peak}(\succ)$ such that $x^\star \succ x$ for all $x \in X \setminus \{x^\star\}$.

Definition 3.3 *A preference relation $\succ$ is considered as single-peaked with respect to a given axis if and only if for all $x_i, x_j \in X$ such that x_i and x_j are on the same*

side of the peak $x^\star$ of $\succ$, one has $x_i \succ x_j$ if and only if x_i is closer to the peak than x_j, that is, if $x^\star > x_i > x_j$ or $x_j > x_i > x^\star$, [41].

Therefore, single-peaked preferences as mentioned in [14] can be seen as a combination of two successive local monotonicities : an increasing order up to a certain peak, and from that peak a decreasing order. In Chapter 5, we elaborate more about single-peaked preferences and how we define single-peaked criteria for MR-Sort models.

3.2.1 Monotonicity in Multiple Criteria Decision Analysis

We briefly discuss the first works on monotonicity in MCDA based on additive models which are most often related to ranking problems.

Early in 1995, Despotis and Zopounidis [31] were the first to consider single peaked value functions with an additive piece-wise linear model for ranking problems. Indeed, they positioned their method as an extension of the UTA method following the principle developed in [59] for decomposition of non-monotone preferences into small attributes ranges where the monotonicity is guaranteed.

Inferring UTA parameters with the consideration of this type of non-monotonicity (single-peaked) is really an asset when knowing the most preferred criteria value of the DM.

Following on from this method, some proposed more elaborated methods such as UTA-NM [60] to allow utility functions with multiple peaked without any beforehand details on the shape of these functions, or a preprocessing heuristic [39] to transform nominal and cardinal attributes inputs into criteria, yielding better results for additive ranking problems. Nevertheless, one of the limit of [60] is their inability to deal with large real-world problems; [39] also failed to capture single-peaked features of cardinal attributes because of their linear regression approach.

In the rest of this section, we investigate the literature related to classification tasks, based on additive, outranking and rule-based models.

Additive models for Multicriteria Sorting problems

In [22], the authors studied non-monotone utility functions with quadratic utility functions for an additive model. They applied their model to a bank credit loan use case. For this application, they characterized the criteria "age" as a quadratic utility function. Indeed considering their use case, young and old people are less inclined to benefit from loans than the rest of the age classes. Therefore the DM preferences decreases when this attribute value gets closer to the endpoints of the

evaluation scale. This type of criteria equates to a single-peaked criterion in our contribution (chapter 5).

Liu *et al.* [64] proposed an approach using a regularization framework - which is known as a tool for addressing trade-offs between the model complexity and the generalization performance (overfitting) [99] - in order to learn the parameters of an additive sorting model. Their disaggregation process uses piecewise linear functions that is beneficial in order to shape precisely the true marginal value function, including monotone and non-monotone from a reference set of alternatives. This approach which learns has the advantage of detecting non-monotone criteria over the algorithm described in [36] for learning UTADIS models used for similar problems.

Guo *et al.* [53] proposed a progressive preference elicitation for multicriteria sorting using a utility model with non-monotone (including multiple-peaked) attributes. They were interested in circumventing the difficulty of learning a representative model given a set of reference alternatives. Therefore inspired by the works from Robust Ordinal Regression (ROR) field [28, 51], they considered the 2 known cases of assignments : possible and necessary assignments. Their algorithm essentially resolves inconsistencies and assigns non-reference alternatives in an iterative and progressive process.

Some recent works of Kadzinski *et al.* [58] addressed the inference of specific parameters of a designed additive sorting model from assignment examples that include diverse forms of marginal values function on non strictly monotone criteria.

They covered 10 types of shapes of monotonicity such as monotonically increasing/decreasing and non-monotone (single peaked, single caved) marginal value functions. The formulated model of Kadzinski et al. takes advantage of subinterval-based construction of marginal functions during the learning process to better fit the marginal functions shapes. Based on their method that uses subinterval-based construction of marginal functions, they were able to retrieve the model parameters (thresholds and functions values) through a Mixed integer linear programming technique. As they inferred shapes of functions that characterizes criteria with their model, we also are interested in learning the preference directions of criteria of MR-Sort models (chapters 4, 6 and 7).

All in all, additive models were predominantly used throughout the past studies. Our studied model (MR-Sort) is rather in the scope of outranking methods. In this area, the preference directions on criteria was always assumed to be part of prior knowledge. In our first contribution (Chapter 4), we focus on inferring from assignment examples, MR-Sort models in the context of partial information on preference directions of criteria (i.e. we assume that preference directions are monotone, but we ignore the direction of such monotonicities).

Outranking models for Multicriteria Sorting problems

As the most known outranking model, ELECTRE TRI has been studied by Mousseau and Słowiński [71]. They proposed a MIP-based algorithm for the inference of ELECTRE TRI models. Due to high effort on the computation of all the ELECTRE TRI parameters in [71], Mousseau *et al.* [70] developed a Linear Program (LP) for learning weights and thresholds with fixed profiles. Conversely, Ngo The and Mousseau [73] proposed a MIP to learn profiles with fixed weights and threshold and presented a global procedure to learn all the parameters.

Doumpos *et al.* in [35] talked about a differential evolutionary algorithm for learning ELECTRE TRI models from assignment examples considering sorting problems. Their method learns all the model parameters and can deal with large dataset sizes, which is an improvement of previous proposals based on MIP algorithm.

To the best of our knowledge, the work proposed for the inferring of outranking models from assignment examples always assumed monotone preference directions of the criteria. Preference directions are also assumed to be known in advance (either increasing or decreasing). In this thesis, we consider a simplified version of ELECTRE TRI and extend this inference problem to unknown preference directions and single-peaked preferences (chapters 4, 6 and 7). We review separately related works of the two outranking models (NCS and MR-Sort) in section 3.3.1.

Dominance-based Rough Set Approach for Multicriteria sorting problems

Among rule-based models and as a MCDA model, Dominance-based Rough Set Approach (DRSA) is not disconnected from the subject of monotonicity of preferences for sorting problems [15, 50].

In [19], the author suggested three ways to deal with the learning of non monotone data, given monotone classification functions :

- enforcing the learning of non monotone data with monotone functions by minimizing the classification error (which is common in Machine Learning),
- transforming non monotone data into monotone data as a preprocessing phase. In [20], Brabant *et al.* proposed the use of fuzzy (Sugeno) integrals to deal with monotone classification problems using the framework of rule-based models. They suggest a monotone relabeling method - which consists of modifying alternatives classes - as a preprocess to deal with non monotone data.
- translating non-monotone data as uncertainty in the data (which is the scope of approaches such as DRSA).

In [14], the authors deal with the induction of decision rules using the DRSA paradigm (introduced in Chapter 2). As we introduce at the beginning of the section 3.2, [14] presented five types of classification problems (including those described as problems *(i)* and *(ii)* in 3.2) depending on the presence or absence of monotonicity relationships between attribute values and assignment classes and present methods to solve them.

They presented a non-invasive transformation (i.e. no interference in the discovering of relationships) to induce monotonicity relationships between condition and decision variables, without knowing a priori the type of monotonicity. They propose to clone non-ordinal condition attributes which results in two attributes - where one represents a positive monotonicity relation and the other a negative monotonicity relation - then apply a DRSA method, which induces decision rules.

Thus, these rules are used to deduce relationships between condition attributes and the decision attribute (globally non monotone with local monotonicities, globally positively monotone or globally negatively monotone).

In the context of MR-Sort, the classification rule is always monotone, while the values order of some criteria may reveal single-peaked preferences, which amounts to two local monotonicity relationships (first an increasing preference relation up to a most preferred point, and after a decreasing preference relation).

In the end, in both contexts (DRSA and MR-Sort), the interpretation of the relationships between such condition variables (single-peaked criteria) and decision variables (categories) is similar. Compared to Brabant *et al.* [20] (previously mentionned), in one of our contributions we transform non-monotone criteria to monotone criteria, except our transformation process is part of the learning approach itself. In Chapter 5 and 6 we leverage the transformation of single-peaked criteria to monotone criteria - which is formulated as a mathematical program - to learn MR-Sort parameters.

3.2.2 Monotonicity in data mining and supervised learning fields

In data mining, the aim of ordinal classification is to assign patterns to classes (each pattern being a tuple of values).

Following the type of data considered (monotone or non monotone), the resolution methods are abundant in the literature. Cano *et al.* [24] reported the evolution of the growing research proposals in the realm of monotone classification.

In [24, 54, 55, 93], they give an overview of ordinal and monotone classification concerning models, methodologies, algorithms, and metrics used. In particular, [24] presented a taxonomy of algorithms for monotone classification. According to their non-exhaustive hierarchy of algorithms, the stakes of the problem we are

dealing with is quite close to the scope of monotone classifier and classification rules algorithms.

The authors also gathered the metrics generally used to evaluate the performances of algorithms as well as datasets involved in the literature for these type of problems. In our work, we use some of them such as UCI and KEEL [1] repositories in order to compare our algorithms with others.

Decision trees

Early in 1999, Potharst and Bioch [78] tackled the learning problem of binary decision trees from monotone data while [79] extend the latter proposal by presenting an algorithm that can handle possibly non-monotone data. Pei and Hu [76] tackled ordinal classification with partially monotone decision trees. They use rank-inconsistent measurements to differentiate attributes from criteria (criteria being usually defined as attributes that are monotone with regard to the decision). Thanks to several entropy measures, they establish some measurements that discriminate not only monotonically increasing (or decreasing) relationships between attributes and the decision, but also determine true criteria from regular attributes.

In Chapter 4 we investigate a similar question that is the inference of criteria preference directions (gain or cost criteria) of MR-Sort models from a priori monotone data.

Regression problems

Tehrani *et al.* [42] presented a choquistic regression based on a gradient-based optimization technique to infer the parameters of the model (i.e. the Choquet Integral parameters). They were inspired by principles behind logistic regression which always requires monotone attributes.

In machine learning, ordinal classification problems have mostly been used to translate into regression problems, whose common drawback is the imprecision when corresponding numerical scale and ordinal one. In addition, most traditional machine learning algorithms have been stamped as "black box" models - despite the emerging of the issue of interpretability in the Machine Learning community - , whereas ours (MR-Sort) has been proved to be easily interpretable [89].

3.3 Problems and algorithms for learning MR-Sort models

In this section, we focus on the outranking method MR-Sort. We have already described in details the MR-Sort model in the previous chapter.

First and briefly, we recall some facts related to MR-Sort and it associated inverse learning problem. Then, we review the existing work related to the other problems related to MR-Sort models, as well as proposed learning algorithms. Finally, we describe the algorithm which is a base for some of our contributions.

Briefly, the aim of the Majority Rule Sorting method (MR-Sort) is to assign the alternatives into a predefined order of categories $C^p \rhd \ldots \rhd C^1$ (C^p and C^1 are respectively the best and worst category). The MR-Sort parameters are the following : weights ($w_i, \forall i \in \mathcal{N}$), majority level ($\lambda$), and limit profiles ($\langle b \rangle$). In order to set appropriate values for these parameters, we consider the *Inv-MR-Sort* problem, which consists in searching for the MR-Sort parameters that match the best the set of assignment examples provided by the DM.

3.3.1 MR-Sort related works

The MR-Sort method is a sub-case of the Non-Compensatory Sorting (NCS) method. NCS was axiomatized in [17, 18]. Compared to the MR-Sort, it generalizes the set of winning coalitions of criteria taking into account non additive majorities.

Thus, individual weights are extended to capacities of subsets of criteria, then allowing more characterizations of the interactions between criteria. NCS itself stems from the ELECTRE TRI sorting method in its pessimistic version. Previous works such as [71] proposed a nonlinear integer programming (NLIP) formulation to learn the parameters for the ELECTRE TRI sorting method (see also [44, 37]).

Later on, Leroy *et al.* [63] used a mixed-integer linear program (MILP) to solve the *Inv-MR-Sort* problem. These two techniques (NLIP and MILP) only allow for solving instances of small size due to computational difficulty.

Meyer and Olteanu studied MR-Sort models by proposing an approach based on an LP formulation and a simulated annealing algorithm to infer MR-Sort parameters under imprecise and missing evaluations [68]. They also established MR-Sort variants based on dominance, dictator and veto principles and proposed exact approaches (MILP) to solve them [67].

Nefla *et al.* [72] presented an interactive elicitation based on the learning of MR-Sort parameters knowing beforehand the profiles values. They used a max-margin optimization technique for the elicitation of the parameters of an MR-Sort model. Assuming that the profiles values are known, their goal was to estimate the remaining MR-Sort parameters, and determine uncertainties associated with DM's questions during the elicitation process. For this purpose, they use a shared non-negative margin as a decision variable that is maximized in the objective function of the optimization phase using a LP. The noise is captured by slack variables. At each iteration, the DM is asked to sort the unassigned alternatives.

Based on an interactive framework, Ozpeynirci *et al.* [74] developed an interactive method for learning a different Inv-MR-Sort problem consisting of determining

a set of actions that improves objects scores and promotes better alternatives assignments while minimizing the cost of such actions. In [75], the same authors consider MR-Sort for assigning alternatives into sorted categories with size restrictions [75].

In [98], they propose an extension of MR-Sort taking into account multiple contexts. This model called MR-Sort-C aggregates the alternatives assignments of multiple MR-Sort "sub-models" depending to their context into a unique MR-Sort. This newly hierarchical model have the potential to enrich the interpretability of those kinds of models. A very similar rationale is also proposed in [97], where the context consists of different time-series.

Recently, efficient Boolean Satisfiability formulations for learning NCS models from data have been proposed [94, 10, 8]. These SAT/MaxSAT formulations makes it possible to handle larger datasets. Belahcene proved in [8] that the (*Inverse-NCS*) problem - which is the inference problem of NCS parameters from assignment examples - is an NP-hard problem even with two categories.

Recently, an evolutionary population-based heuristic has been proposed to solve Inv-MR-Sort, see [90, 88]. This heuristic is indeed computationally limited ; the execution time is dependant on a setting defined beforehand. Considering our first research question - which is to learn MR-Sort models where the preference order on some criteria is unknown - we need to learn additional information in comparison to the standard MR-Sort learning problem. As a metaheuristic, the Sobrie's algorithm [90] is flexible and naturally offers the possibility to introduce this additional setting for addressing our new problem (chapters 4 and 7).

Before detailing our methods in next chapters, we provide a description of this metaheuristic, since two contributions are directly inspired by this algorithm.

3.3.2 Description of the existing metaheuristic

The heuristic proposed in [88, 90] is an evolutionary population-based algorithm. It learns an MR-Sort model that best matches a learning set composed of assignment examples.

As an evolutionary algorithm, each individual (denoted by $(\langle b \rangle, w, \lambda)$) is an MR-Sort model. The first population generation is initialized with N_{mod} models (Algorithm 1, line 2). Random weights and thresholds are generated for each model while their profiles are generated following a specific heuristic (described in [90]) that takes into account the representativeness of alternatives in the categories.

After the initialization step, models in the population are subject to two successive improvement steps (Algorithm 1, line 5 and 6) in their ability to correctly restore the assignment examples. These two steps are repeated for as many iterations as needed to reach the ideal model within a prefixed number of iterations N_o (Algorithm 1, line 3). An ideal model in a population - which is not always retrieved since the heuristic is an approximate method - is the model that restores

entirely the assignment examples. In that case, the fitness score (which corresponds to classification accuracy (CA)) equals 1. The fitness is computed at the end of each iteration (Algorithm 1, line 8). Then, the population renewal is carried out by replacing the worst half models - based on their fitness - in the population by new randomly generated models (Algorithm 1, line 9).

Algorithm 1: Inv-MR-Sort heuristic, [90]

Input: L: learning set
Output: model $(\langle b \rangle, w, \lambda)$ that best match L in the population

1 $it \leftarrow 1$
2 Initialize POP, a population of N_{mod} models
3 **while** *($it \leq N_o$) and (no model in POP fully restores L)* **do**
4 **foreach** *model* $(\langle b \rangle, w, \lambda) \in POP$ **do**
5 Optimize weights w and threshold λ using LP
6 Improve profiles $\langle b \rangle$ heuristically N_{it} times
7 **end**
8 Compute the fitness of models
9 Renew the $\lfloor N_{mod}/2 \rfloor$ worst models in POP
10 $it \leftarrow it + 1$
11 **end**
12 **return** $(\langle b \rangle, w, \lambda)$ that best match L in POP

The first improvement step (Algorithm 1, line 5) of an individual relates to the optimization of weights w and the majority threshold λ whereas profiles are fixed. This part is solved with a Linear Programming (LP) algorithm. In this program, the constraints account for the characterization of w and λ as MR-Sort parameters, and for the compatibility of these parameters to assignments examples. The weight $w_i, \forall i \in \mathcal{N}$ is bounded below by 0 and above by 1. The objective function accounts for the correctness of parameters according to assignments examples through the minimization of the sum of slack variables (which expresses inaccuracies).

The second improvement step (Algorithm 1, line 6) is dedicated to the adjustment of profiles given optimized values of w and λ obtained at the previous step. It uses a well-elaborated heuristic for choosing appropriate profiles values b_i^h ($i \in \mathcal{N}$, $h \in \{1, \ldots, p\}$) that optimize the restoration of assignment examples.

It associates both an iterative process and a randomized selection of candidates values for profiles [90]. The number of iteration (N_{it}) of this process is known a priori. The search range of profiles values $b_i^h, \forall i \in \mathcal{N}$, $h \in \{1, \ldots, p\}$ is X_i.

These two components used alternately have proved their effectiveness in the restoration of assignment examples, taking into account of artificial instances. For

instance, on only the first component, experiments in [88] shows that CA_g exceeds 98% for an MR-Sort model with 10 criteria, 3 categories and only 200 alternatives in the learning set and 10000 alternatives in the test set (randomly generated instances without noise). With the same settings, the CA_g reaches to 95% based only on the execution of the second component. The reader can refer to [90, 88] for more details on those experiments.

3.4 Conclusion

In this chapter, we reviewed the literature related to monotone , partially monotone and non monotone classification. Then, we gave an overview of existing problems and algorithms for sorting problems in MCDA as well as classification on fields such as Preference Learning, Data mining and some supervised learning domains.

Towards the end of our chapter, we made an inventory of some approaches developed to tackle MR-Sort-related problems. In particular, we introduced the metaheuristic proposed by [90] to infer MR-Sort parameters from monotone data. This algorithm paves the way to our following chapter on the learning of MR-Sort models with unknown criteria preference directions.

Chapter 4

Learning MR-Sort models with unknown preference directions

Contents

4.1 Introduction

In Multiple Criteria Decision Analysis (MCDA), the disaggregation process consists in reconstructing the model features that characterize the Decision Maker (DM) preferences. It has been applied with MR-Sort models in order to learn the standard MR-Sort parameters (profiles values, criteria weights and majority threshold), assuming that the preference directions of criteria were given [63, 90, 88]. The preference directions refer to the preference order introduced in the Chapter 2 (Section 2.3.2).

In this chapter, we investigate the learning problem that pertains to the case where the preference directions are not known in advance. However, the Decision Analyst (DA) (with regards to the context, and having discussed with the DM) presumes to deal with monotone preferences (gain or cost criterion, see Section 2.3.2).

We explore mainly two approaches, both based on the metaheuristic algorithm proposed in [90] in order to learn MR-Sort models in the case where preference directions are unknown. The first approach (duplicated-based approach discussed in Section 4.3) consists in two distinct steps (determining preference directions, then retrieving the standard MR-Sort parameters); this work was also proposed in [69]. The second approach (mixed-based approach in Section 4.4) consists of evolving models with both gain and cost criteria in the population of models during the learning process. We end our chapter with some experiments and concluding remarks in Section 4.5.

First of all, we introduce in the following example, the research question of this chapter.

An illustrative example

A company wants to launch an advertisement campaign whose goal is to send targeted emails to clients interested by their new product : a Windows tablet. The marketing team first aims at building a recommendation tool in order to sort clients into two groups : those who are supposedly interested in buying the new product (I), and those who are not interested (N). To achieve this task, the team wants to use historical purchases of clients in terms of revenues per type of products in order to build a decision model. Thus, they consider the following :

- the *Windows PC* turnover (in €) noted c_1
- the *Pack Office* turnover (in €) noted c_2
- the *Linux PC* turnover (in €) noted c_3

	c_1 (↑)	c_2 (↑)	c_3 (↓)	c_4 (?)	$c(a)$
a_1	500 k€	20 k€	300 k€	150 k€	N
a_2	200 k€	10 k€	350 k€	130 k€	N
a_3	800 k€	90 k€	150 k€	100 k€	I
a_4	600 k€	50 k€	300 k€	100 k€	I
a_5	900 k€	70 k€	250 k€	100 k€	I

Table 4.1: Assignment examples: dataset of 5 clients and 4 criteria

- the *Dual boot PC* turnover (in €) noted c_4

Given that the company wants to commercialize a Windows device, the *Windows PC* and *Pack Office* criteria are gain criteria i.e. the more the client buys Windows PCs and Pack Office, the more he/she is inclined to purchase Windows devices. On the opposite the criterion *Linux PC* is a cost criterion, i.e the more a client buys a Linux PC, the less he/she is inclined to purchase Windows devices. In order to leverage all the data at disposal, the manager of the team suggests to also consider clients' revenues on Dual Boot PC, even if he does not know whether it is a gain or a cost criterion regarding to the recommendation task. We present in the Table 4.1, a sample of five clients (a_1 to a_5) that have been assigned to the two categories : I and N, with $I \rhd N$, where $\rhd$ indicates the order between categories (in our case I is the best category and N the worst one).

In the MR-Sort method, Windows PC and Pack office purchases indicate an interest in a Windows environment, hence should be maximized whereas purchases of Linux PCs indicates disinterest for Windows product and should be minimized. The purchase of Dual boot PC does not induce *a priori* any particular optimization direction regarding the interest for Windows environment. Considering this dataset (Table 4.1), our research question consists in determining a method to retrieve the preference direction of criteria like c_4 together with the other parameters (weights, limit threshold and limit profiles) of the MR-Sort decision model.

4.2 Basic notations and reminder

Let recall $\mathcal{N}$ the set of criteria. The evaluation scale X_i, for $i \in \mathcal{N}$ represents the set of possible performances on the criterion i. The preference direction d_i describes how the preference relation $\succcurlyeq_i$ on criterion i relates to the evaluations on X_i. Criterion i has an increasing (decreasing, resp.) preference direction, noted d_i^+ (d_i^-, resp.), when criterion i is a gain (cost, resp.) criterion. The vector of criteria preference directions is noted : $d = (d_1, ..., d_n)$; d can be considered as an additional

parameter of the MR-Sort model, whose original parameters are : $\langle b \rangle = \langle b^1, ..., d^p \rangle$ the evaluation profile, w the criteria weight vector, and λ the majority threshold.

The *Inv-MR-Sort* problem consists of retrieving the parameters of the MR-Sort model parameters that best match assignment examples. In this chapter, we are interested in extending the *Inv-MR-Sort* problem to a broader problem that also comprises the learning of the preference directions of criteria. We note $\mathcal{Q}$ the set of unknown preference directions criteria ($\mathcal{Q} \subseteq \mathcal{N}$ and $|\mathcal{Q}| = q$). Therefore we denote $IMS_{q|n}$, the *Inv-MR-Sort* problem that aims at learning q preference directions over n criteria in the model ($q \leq n$). In the following, we consider the $IMS_{q|n}$ problem, that aims at inferring the tuple of parameters $(\langle b \rangle, w, \lambda, \{d_i : \forall i \in \mathcal{Q}\})$.

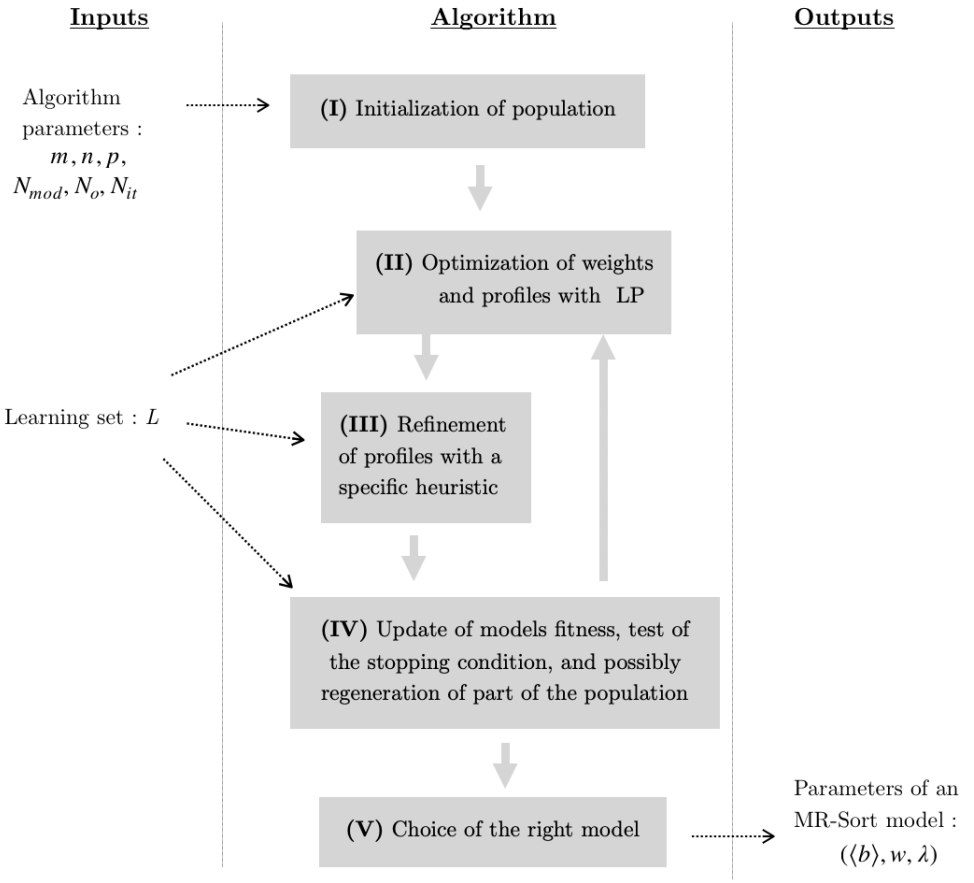


Figure 4.1: Learning process of the Sobrie's metaheuristic

Remarks

We sometimes refer to unknown preference directions as latent preference

directions since the preference directions - that are inherent to the criteria - exist but are not revealed yet. In the rest of the chapter for simplicity and without loss of generality, when the preference direction is known, without indication, we assume that the criterion is a gain criterion.

Finally, for the sake of readability of our proposals, we recall the procedure of the original metaheuristic for learning MR-Sort models from monotone data [90] (see Figure 4.1). In a nutshell, the algorithm is an evolutionary method which aims at evolving a population of models. The improvement phase of parameters relies on two successive optimization phases : update of weights and the majority threshold, and the refinement of profile values. The reader can refer to the Section 3.3.2 for an in-depth presentation of the algorithm.

We summarize in Figure 4.1 the main components of the metaheuristic as well as inputs/outputs of the algorithm.

4.3 The duplication-based approach

4.3.1 Motivations and guiding principles for the learning of preference directions

Let us consider an MR-Sort learning problem in which all criteria directions are known except for criterion i. We can use the procedure of Figure 4.1 to learn an MR-Sort model $(\langle b \rangle, w, \lambda)$ from a dataset, hypothesizing incorrectly the preference direction for criterion i (i.e. i is considered as a gain criterion while in the ground truth, i is a cost criterion). In this case, the behaviour of the procedure (Figure 4.1) is singular; it strives to best restore the learning set at the risk of *inhibiting* criterion i.

Let $h, h \in \{1, ..., p-1\}$ the index of the category C^h. Thus, b_i^h denotes the profile value that separates C^{h+1} from C^h on criterion i. We consider a reference set of alternatives $A^\star$, and we denote by a_i the evaluation of an alternative a on criterion i. In this context, models that inhibit criterion i are those for which $w_i = 0$, or those for which $b_i^h > Max_{a \in A^*}\{a_i\}$, $\forall h$ or $b_i^h < Min_{a \in A^*}\{a_i\}$, $\forall h$.

The rendered values $(\langle b \rangle, w, \lambda)$ of the procedure (Figure 4.1) can be informative regarding two aspects. For instance, a very small weight w_i could result in a criterion i that cannot be part of any minimal sufficient coalition of criteria. Also, the profile $\langle b \rangle$ could be such that b_i^h values are close to the endpoints of the scale X_i, meaning that b_i^h is not discriminating enough. When these behaviours occur, it could be interpreted as an inhibition of i due to the fact that i was assigned to a wrong preference direction.

With these observations, we propose in the following an algorithm that learns the MR-Sort parameters in two steps.

4.3.2 The duplicated-based algorithm

In this section, we describe the duplicated-based algorithm which is a two-stages method based on the duplication of criteria to solve the $IMS_{q|n}$ problem (the inverse MR-Sort problem with q latent criteria preference directions or q unknown preferences directions). The two stages correspond to *(i)* learning the unknown preference directions ($\{d_i : \forall i \in \mathcal{Q}\}$), and then *(ii)* learning the other parameters ($\langle b \rangle, w, \lambda$).

Schema of the approach based on duplicated criteria

We describe in Figure 4.2 the main components of the approach as well as inputs/outputs of the algorithm. We enumerate 6 inputs : the size of the learning set m, the number of criteria n, the number of categories p, the number of models in the population N_{mod}, the number of iterations of the outer loop N_o and the number of iterations of the inner loop of the metaheuristic N_{it}.

The differences with the Sobrie's heuristic [90] are mentioned in red.

The first stage of the approach (Figure 4.2) consists of three steps. First, the initial problem is modified to obtain an $IMS_{0|n+q}$ problem in which the q latent criteria are duplicated (**Step I**). Then, the procedure of Figure 4.1 is executed on $IMS_{0|n+q}$ (**Step II**). This leads to a problem that takes into account two types of preference directions (profit and cost criterion) for the q latent criteria. At the end of the first stage, we induce the plausible preference directions of the q latent criteria (**Step III**). In the second stage, the procedure of Figure 4.1 is performed with the q preference directions that have been fixed (**Step IV**). Finally we choose the right model which will give the remaining parameters of the learned model (**Step V**). Let us now present the details of each step.

4.3.3 First stage

The first stage of the proposed approach performs consecutively three steps:

- the transformation of $IMS_{q|n}$ to an intermediate problem ($IMS_{0|n+q}$) obtained through the duplication of criteria with unknown preference directions (see Figure 4.2, **Step I**),
- the resolution of $IMS_{0|n+q}$ with the procedure of Figure 4.1 (see Figure 4.2, **Step II**),
- the deduction of the q preference directions of the initial problem from the output of $IMS_{0|n+q}$ (see Figure 4.2, **Step III**)

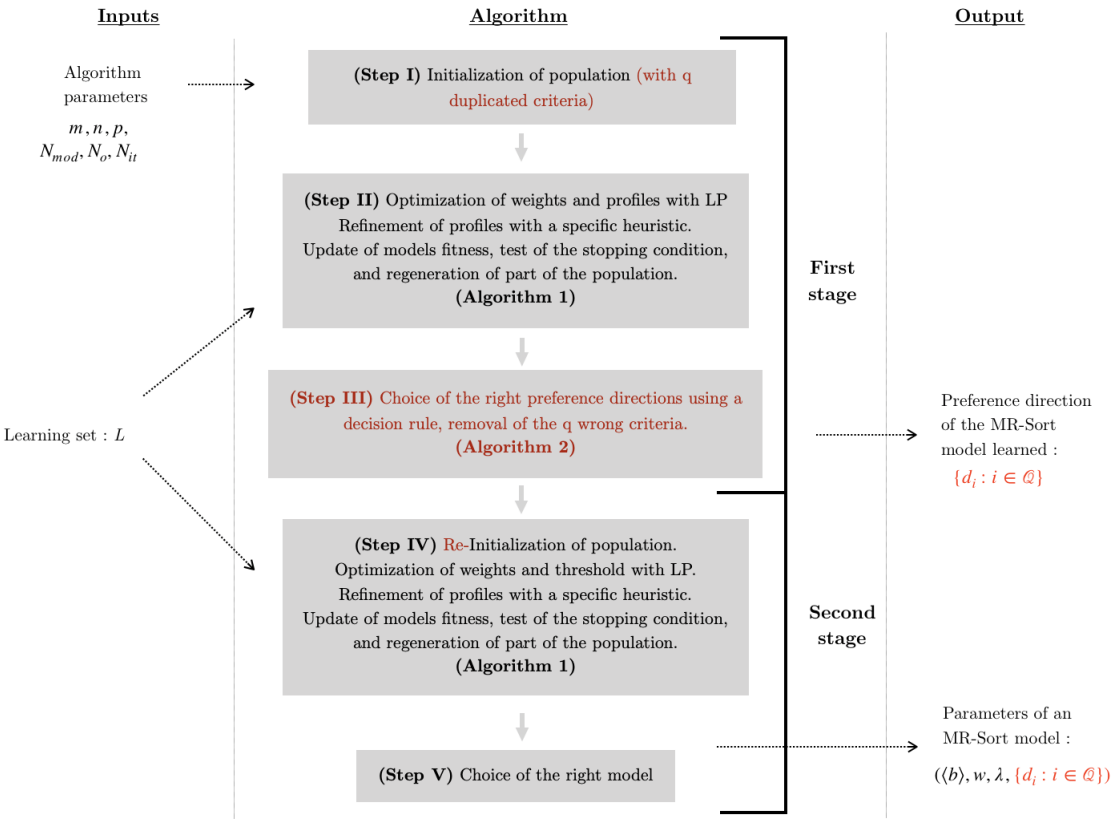


Figure 4.2: Learning process of the approach based on the duplication of criteria

Step I : From $IMS_{q|n}$ to $IMS_{0|n+q}$:

The intuition behind the duplication criteria is to explore the combinations of preference directions over criteria with unknown preference directions. Then, it is possible to foster the learned model to distinguish between criteria with wrong preference directions from those with true preference directions.

The duplication principle is the following. Considering $IMS_{q|n}$, it consists in duplicating the subset $\mathcal{Q}, Q \subset \mathcal{N}$ into a similar set $\mathcal{Q}'$ except that preference directions of elements of $\mathcal{Q}'$ and the ones in $\mathcal{Q}$ are opposite. More precisely, without loss of generality, we choose to assign to elements of $\mathcal{Q}$ the increasing preference direction ($d_i = 1$, i.e $i, i \in \mathcal{Q}$ is considered as a gain criterion), and to the elements of $\mathcal{Q}'$ the decreasing preference direction ($d_i = -1$, i.e , $i \in \mathcal{Q}$ is considered as a cost criterion). The duplication also consists in the copy of the sub-matrix of the performance table underlying $\mathcal{Q}$ over the alternatives towards the one that matches with $\mathcal{Q}'$.

Thus, we associate to each criterion $i \in \mathcal{Q}$ to the criterion $\tau(i) \in \mathcal{Q}'$ that shares the same performances values over the alternatives : they both account for the same initial criterion whose preference direction is unknown. Let us assume that $\mathcal{Q} = \{1, ..., q\}$ and $\mathcal{Q}' = \{n+1, ..., n+q\}$, considering $\mathcal{N} \cap \mathcal{Q}' = \{1, ..., q, ..., n, n+1, ..., n+q\}$. There is a correspondence between $\mathcal{Q}$ and $\mathcal{Q}'$ since $\tau(i) \in \mathcal{Q}'$ is the copy of $i \in \mathcal{Q}$. We call $\mathcal{AC}$ the set of couples of criteria $(i, \tau(i))$ of this sort. This modified problem can be read as $IMS_{0|n+q}$, since there is no more unknown preference directions and the total number of criteria rises to $n+q$. Thus, through the duplication of i to $\tau(i)$, we foster the learned model to inhibit the criterion that carries a wrong preference direction (i or $\tau(i)$), while making the other criterion more influential. For the sake of simplicity, we name $i' = \tau(i)$ the duplicate of i in the remaining of the document.

Step II : Resolution of $IMS_{0|n+q}$:

At this step, we solve $IMS_{0|n+q}$ using the procedure described in Figure 4.1. The problem comprises $n+q$ criteria which implies the learning of $n+q$ weights, profiles $\langle b \rangle$ of dimension $n+q$, as well as a threshold λ. These temporary outputs are only used to induce the q desired preference directions in the next step.

Step III : Deduction of q preference directions and the construction of $IMS_{0|n}$:

After the resolution of $IMS_{0|n+q}$, we now interpret the preference directions of the q latent criteria.

Our reasoning is the following. Considering the resulted parameters $(\langle b \rangle, w, \lambda)$ of $IMS_{0|n+q}$ obtained in the previous step :

1. $\forall\ (i, i') \in \mathcal{AC}$, if $w_i = 0$ and $w_{i'} \neq 0$ then we derive that the correct criterion is the cost criterion i', since i is inhibited in the model ($w_i = 0$).

2. $\forall\ (i, i') \in \mathcal{AC}$, if $w_i \neq 0$ and $w_{i'} = 0$ then we derive that the correct criterion is the profit criterion i, since i' is inhibited in the model ($w_{i'} = 0$).

3. $\forall\ (i, i') \in \mathcal{AC}$, if $w_i \neq 0$ and $w_{i'} \neq 0$, thus we cannot immediately derive the correct preference direction on the basis of weight values. We ground our analysis on the position of profiles $\langle b \rangle$ on criteria i and i'. As mentioned above, profiles on criterion i (or i') close to the endpoints of the scale X_i (or $X_{i'}$) indicates that criterion i (or i') is almost *"inhibited"*. Therefore, we select the preference direction corresponding to criterion i or i' as the one for which the profile is the further away from the endpoints of the scales X_i and $X_{i'}$.

   In order to measure how close a profile can be regarding to the endpoints of the evaluation scale X_i, we introduce $X_i^\star \subseteq X_i$ which is the evaluation scale induced by the reference set $A^\star$. We partition $X_i^\star$ into p disjoint sets i.e. $X_i^\star = \bigcup_{h \in \{1,...,p\}} X_i^{\star h}$. For each set, with h the index of the category C^{h+1}, we have : $X_i^{\star h} = \{x_i \in X_i^\star : b_i^{h-1} \leq x_i < b_i^h\}$.

   We then introduce a measure $\mu_i \in [0, 1]$ - which evaluates the proportion of the largest set among $\{X_i^{\star 1}, ..., X_i^{\star p}\}$ - with the following formula :

$$\mu_i = \frac{Max\{|X_i^{\star h}| : \forall h \in \{1, ..., p\}\}}{|X_i^\star|} \tag{4.1}$$

   Our intuition is that a high value of μ_i is an indicator of an unbalanced distribution of $X_i^{\star 1}, ..., X_i^{\star p}$. At the extreme case, the largest set tends to cover $X_i^\star$ which results in profile values b_i^h that stack at the endpoints of $X_i^\star$. As we conclude previously, this is a sign that i is *"inhibited"*. Let us define a decision threshold $\mu, \mu \in [0, 1]$. We consider three configurations, regarding how μ_i and $\mu_{i'}$ compare to μ (μ_i, resp. $\mu_{i'}$ stands for the measure computed in Equation 4.1 considering criteria i resp. i'):

   - If $\mu_i > \mu$ and $\mu_{i'} \leq \mu$ then we derive that the right criterion is the cost criterion i'. This case translates a situation where the criterion i is less discriminating than i'. In fact, if b_i^h values are extreme, then i has a low discriminating power. Therefore, the more extreme μ is, the more strict is this rule. In our tests, we fix $\mu = 0.9$.
   - If $\mu_{i'} > \mu$ and $\mu_i \leq \mu$ then we derive that the right criterion is the profit criterion i. In this case, the criterion i' is less discriminating than i.

- Otherwise, we call an *ad hoc* heuristic (see Algorithm 2) considering the problem $IMS_{1|1}$ (which involves only one criterion whose preference direction is unknown) with the following inputs : $(i, i') \in \mathcal{AC}$, L, X_i and expect as an output the right preference direction of i.

4. $\forall\ (i, i') \in \mathcal{AC}$, if $w_i = 0$ and $w_{i'} = 0$, we refer to an *ad hoc* heuristic (see Algorithm 2) considering the problem $IMS_{1|1}$ with the following inputs : $(i, i') \in \mathcal{AC}$, L, X_i and expect as an output the right preference direction of i.

Remark

One can consider in the case where $w_i = 0$ and $w_{i'} = 0$, $\forall\ (i, i') \in \mathcal{AC}$ removing both i and i'. Learned models in such a case do not involve i and i' in the classification rule. However, it is not accurate to remove them since the computed weights depend on the considered learning set. Having $w_i = 0$ and $w_{i'} = 0$ as outputs of the Sobrie heuristic does not necessary mean that the ground truth model possesses such weight values.

The *ad hoc* heuristic:

We describe a heuristic that aims at estimating the preference direction of a criterion whose preference direction is a priori unknown. Let us consider the original problem $IMS_{q|n}$ and its restriction to one of the unknown preference direction criteria, called $IMS_{1|1}$ (with $\mathcal{AC} = \{(i, i')\}$). The resolution of this new problem takes into account the evaluation performance of the unknown preference direction criterion and the learning set.

Solving $IMS_{1|1}$ is less complex than solving $IMS_{q|n}$ since we have only one criterion in the model that plays a dictatorship role. In fact, the resolution of $IMS_{1|1}$ amounts to heuristically compute the profiles of the model $\langle b \rangle = b^h$, $\forall h \in \{1, ..., p\}$ that optimize the restoration of assignments examples (see Algorithm 2).

In the heuristic (Algorithm 2), the set B is kept updated by new constructed values profiles as it will contain the learned profiles values $\langle b \rangle$. In order to build B, we first initialize B with b^0 and b^p which are respectively the minimum value of X_i and the maximum value of X_i (lines 1-3 of Algorithm 2). Then, for each profile b^h - randomly chosen among those that are not constructed yet (line 4 of Algorithm 2) - we compute the interval $[b^y, b^z]$ that can be seen as the search area. b^h should belong to $[b^y, b^z]$, which is the narrowest interval whose interval bounds are known at the moment of the process (see lines 5-6 of Algorithm 2 for the construction of the interval). B is the set of profiles known or already constructed at the current state of the process. We introduce two functions : $e_{max}(x)$ and $e_{min}(x)$. Let $x \in X_i$

such that x be a potential candidate value for b^h. The function $e_{max}()$ (resp. $e_{min}()$) associates to x, the number of alternatives in accordance with the choice of x as b^h in case i is a gain (resp. cost) criterion. In order words, assuming that i is a gain, $\forall a \in A^\star$ and $cat(a)$ the category of a, $e_{max}(x)$ is the number of alternatives such that :

- the evaluation (a_i) is included in $[b^y, x]$ and the assignment category $cat(a)$ belongs to the set of categories induced by $[b^y, x]$ i.e $cat(a) \in \{C^{y+1}, \ldots, C^h\}$
- the evaluation (a_i) is included in $[x, b^z]$ and the assignment category $cat(a)$ is in the set of categories induced by $[x, b^z]$ i.e. $cat(a) \in \{C^{h+1}, \ldots, C^z\}$

An analogue reasoning can be done with $e_{min}(x)$ considering the case where i is a cost criterion. Therefore, $max(e_{max})$ (resp. $max(e_{min})$) indicates the maximum number of correctly assigned alternatives involving profile b^h when i is a gain criterion (resp. cost criterion). $argmax(e_{max})$ (resp. $argmax(e_{min})$) is the optimal position for b^h when i is a gain criterion (resp. cost criterion) (lines 11-12 of Algorithm 2).

Once profiles are heuristically constructed, we are interested in the classification accuracy obtained when considering both possible preference directions for this criterion with unknown preference direction (lines 16-17 of Algorithm 2). We note them CA^i and respectively $CA^{i'}$, the classification accuracy when the unknown preference is considered as a gain criterion i and respectively the one when considered as a cost criterion i'.

We advocate that the correct criterion (i or i') corresponds to the one whose classification accuracy (CA^i or $CA^{i'}$) is the greatest (lines 18-19 of Algorithm 2) since the classification supposedly is more favorable for one case rather than the other.

4.3.4 Second stage

Once the q preference directions have been determined, we can reduce $IMS_{0|n+q}$ to $IMS_{0|n}$ by preserving the right q criteria previously derived and by removing their associated criteria. Next, we solve the standard $IMS_{0|n}$ problem with the procedure described in Figure 4.1 (see Figure 4.2, **IV**).

The resulting parameters of this last process give the remaining part of the solution to the initial problem $IMS_{q|n}$. In this way, we are able to retrieve the four parameters $(w, b, \lambda, \{d_i : \forall i \in \mathcal{Q}\})$ of the *Inv-MR-Sort* problem with latent preference direction criteria.

Algorithm 2: Heuristic for selecting the right preference direction given $(i, i') \in \mathcal{AC}$ and $IMS_{1|1}$

Input: $IMS_{1|1}$, $A^\star$, $\{C^1, ..., C^p\}$, $(i, i') \in \mathcal{AC}$, L, X_i the evaluation scale of i

Output: d_i or $d_{i'}$: the right criterion preference direction in $IMS_{1|1}$

$b^0 = min\{\forall x \in X_i\}$;
$b^p = max\{\forall x \in X_i\}$;
$B = \{b^0, b^p\}$;
foreach *profile* $b^h \in \{b^1, ..., b^{p-1}\}$ *randomly chosen and* $b^h \notin B$ **do**
  $b^y = max\{b \in B; b < b^h\}$;
  $b^z = min\{b \in B; b > b^h\}$;
  foreach $x \in X$ *and* $x \in [b^y, b^z]$ **do**
    $e_{max}(x) = |\{\forall a \in A^\star; (cat(a) \in \{C^{y+1}, ..., C^h\}$ and $a_i \in [b^y, x])$ or $(cat(a) \in \{C^{h+1}, ..., C^z\}$ and $a_i \in [x, b^z])\}|$;
    $e_{min}(x) = |\{\forall a \in A^\star; (cat(a) \in \{C^{h+1}, ..., C^y\}$ and $a_i \in [b^y, x])$ or $(cat(a) \in \{C^{z+1}, ..., C^h\}$ and $a_i \in [x, b^z])\}|$;
  end
  if $max(e_{min}) > max(e_{max})$ **then** $b^h \leftarrow \underset{x}{\text{argmax}}\ e_{min}$;
  else $b^h \leftarrow \underset{x}{\text{argmax}}\ e_{max}$;
  $B \leftarrow B \cup \{b^h\}$;
end
$\langle b \rangle \leftarrow B$;
$CA^i = \frac{\sum_{h=1}^{p-1} |\{\forall a \in A^\star; a \in X^{\star h} \text{ and } cat(a) = C^h\}|}{|A^\star|}$;
$CA^{i'} = \frac{\sum_{h=1}^{p-1} |\{\forall a \in A^\star; a \in X^{\star h} \text{ and } cat(a) = C^{p-h+1}\}|}{|A^\star|}$;
if $CA^i > CA^{i'}$ **then return** d_i ;
else return $d_{i'}$;

4.4 The mixed-based algorithm

4.4.1 Motivation

We present in this section another contribution for learning MR-Sort models with latent preference directions from assignment examples. This method is inspired by some facts regarding the original metaheuristic [90]. In the original metaheuristic, the addressed models were homogeneous from the point of view of preference directions of criteria. The models in population were endowed with the same preference directions for all criteria, that were gain criteria. However, it was

possible to express a cost criterion for instance by converting the performance values of a gain criterion to their opposite values - which therefore reverses the preference order of that criterion.

As we are dealing with the *Inv-MR-Sort* problem with latent criteria preference directions, it makes sense to relax the assumption of the homogeneity of the population. This is the fundammental idea behind our second approach.

Our proposal consists in enabling models with increasing preference directions alongside models with decreasing preference directions in the same population to evolve during the resolution process. We recall that this variation of preference direction occurs only on latent criteria preference directions. The remaining criteria are not subject to this variation since their preference directions are already known.

With this approach, the number of combinations of preference directions over $\mathcal{Q}$ - that could be considered in the population - grows exponentially with the increase of $|\mathcal{Q}|$. Therefore, it is essential to master this phenomenon, and our goal consists of fostering the evolution of models that possess correct preference directions.

Nevertheless, the constant objective is the learning of models whose restoration rate of the learning set (CA_v) is the greatest.

In summary, we simultaneously perform 2 goals : the retrieval of the q correct preference directions, and the parameters of the MR-Sort model that best restore the learning set.

4.4.2 Definitions and overview on the approach

Our proposal is based on the adaptation of the metaheuristic [90] considering three aspects : *heterogeneous models*, the *use of additional parameters*, and a *modified metaheuristic* oriented towards the resolution of our specific problem (which is to simultaneously handle gain and cost criteria, in the model).

We summarize in Figure 4.3 the main components of the approach as well as inputs/outputs of the algorithm. The differences with the Sobrie's metaheuristic are mentioned in red.

The resolution of the $IMS_{q|n}$ problem requires 5 steps (see Figure 4.3):

- **Step I: Population Initialization.** We initialize the population of models according to a renewal rate and/or a renewal coefficient (that are described below).

- **Step II: Update of models (individuals) parameters (weights).** During this step, we apply one of the core strategies of this approach. It consists of penalizing criteria in $\mathcal{Q}$ by narrowing the scale of their weight value throughout the optimization phase.

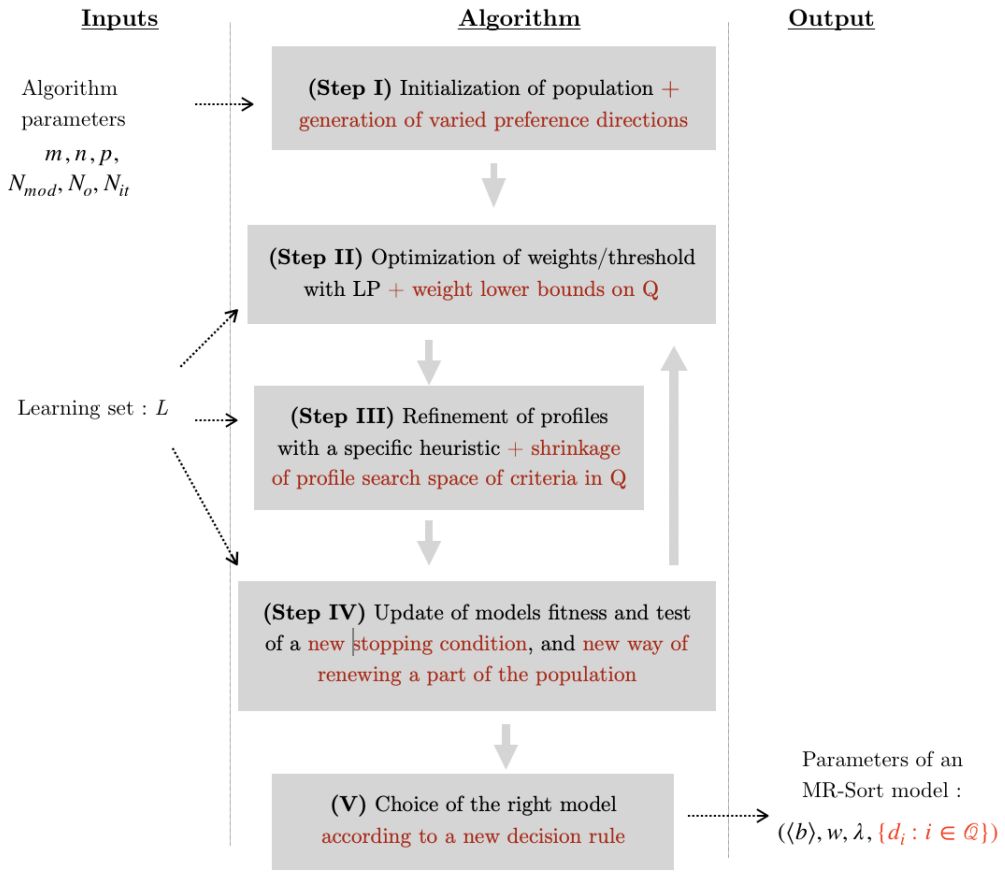


Figure 4.3: Learning process of the approach with mixed models in the population

- **Step III: Update of models (individuals) parameters (profiles).** We apply the other main strategy of this approach. It consists of penalizing criteria in $\mathcal{Q}$ by narrowing the scale of their profile.
- **Step IV: Renewal of the population.** Reaching this point, we examine the satisfaction of the stopping condition. Either the condition is fulfilled then we move to the next step (**Step V**); otherwise, we renew the population and return to **Step II**, thus restarting an other iteration of the metaheuristic.
- **Step V: Final step.** At this phase, we conclude this algorithm by selecting the appropriate model with the help of a decision rule.

In order to configure and control the evolutionary process of the novel approach, we need to define 5 parameters - in addition to the accustomed parameters N_{mod}, N_o, N_{it} described in [88] :

- the *renewal rate* (noted rr): this rate refers to the proportion of renewal of the population based on the fitness score of models (CA). For example, $rr = 0.5$ means that the renewal pertains to half of the bad models (in terms of their fitness) in the population. The settings of the original metaheuristic [88] equate to the case where this parameter is fixed to 0.5.

- the *renewal coefficient* σ: this parameter refines the renewal rate rr throughout the process of the evolutionary algorithm. This parameter is dependant of rr and the quality of the models in the population expressed by $\overline{CA}$, which is the average fitness score of the models in the population.

  Here, the rationale of σ is to renew models more frequently when the average fitness score of models is low and therefore boost a significant renewal of the population.

  From another viewpoint, σ is a metaparameter of rr ; it controls the variation of rr. The formula of the renewal coefficient is the following :

  $rr = \frac{1-\overline{CA}}{\sigma}$.

  With a constant $\overline{CA}$ and for small (resp. high) values of σ, rr rises (resp. falls) which increases (resp. decreases) the number of renewed models in the next generation.

- the *distribution of the preference directions* $\pi_i = [\pi_i^+, \pi_i^-]$ of the criterion i, $i \in \mathcal{Q}$: where π_i^+ (respectively π_i^-) is the probability of assigning a gain criterion (respectively cost criterion) to i, given a model in the population. We have $\pi_i^+ + \pi_i^- = 1$, $\forall i \in \mathcal{Q}$.

- the *maximum weight lower bound* w_* of criteria with latent preference directions : this value is the minimum value that can take w_i, $i \in \mathcal{Q}$.

- the *maximum order* φ *(*$0 < \varphi < 50$*) of the interquantile of* X_i, $i \in \mathcal{Q}$: this parameter defines a new domain of variation of profiles : the interquantile range of order φ (i.e the interval between the value of the φ^{th} quantile and the value of the $(100 - \varphi)^{th}$ quantile of the interval of ordered values of X_i).

The details of the steps of our approach are described in what follows.

4.4.3 Initialization of the population (Step I)

First, we generate N_{mod} models. For each model, we generate random profiles values b_i^h, $h \in \{1, ..., p\}$, $i \notin \mathcal{Q}$ within [0,1] using the same heuristic described in [90]. We randomly draw b_i^h, $i \in \mathcal{Q}$, $h \in \{1, ..., p\}$, within the interquantile range of

order φ of X_i (which is fixed to X_i at the initialization phase) such that $b_i^h < b_i^{h+1}$, $i \in \mathcal{Q}$, $h \in \{0, ..., p-1\}$.

At the beginning of the process, we do not have any hint regarding the preference directions of criteria in $\mathcal{Q}$, namely the values of π_i, $i \in \mathcal{Q}$. Therefore, it appears fair to assign to each criterion i, $i \in \mathcal{Q}$, a gain or a cost criterion with an equal chance to be chosen (i.e $\pi_i = [0.5, 0.5]$). More precisely, we randomly choose a real in [0;1[; if this real is greater than 0.5, then i is a gain criterion (d_i^+), otherwise i is a cost criterion (d_i^-).

At this point, the population is generated and ready for the evolution process.

4.4.4 Update of models parameters

In the following, we describe how the models parameters are built. On the one hand, we optimize weights and thresholds with a Linear Program (LP), and on the other hand we adjust profiles with a dedicated heuristic.

We choose to examine the key factors (weights and profiles) that influence the choice of the preference direction of models while minimizing the loss of efficiency and effectiveness of the computation of the parameters. Our contribution is divided into two successive stages: first, during the optimization of weights and then during the readjustment of profiles.

Optimization of weights (Step II)

The weight intensity of a criterion determines its influence in the restoration of assignment examples, indirectly the restoration of preference direction.

Therefore, we choose to constrain the construction of weights. The rationale is to prevent the inhibition of criterion i, $i \in \mathcal{Q}$ by forcing $w_i > 0$ in order to make i influential. Therefore, we consider the LP formulation with the constraints pertaining to the optimization of weights as in [90]. We also add to these constraints, for each criterion i, a constraint that shrinks the definition domain of w_i to [w_*,1], w_* being the lower bound of w_i (with $0 \leq w_* < 1$). The constraint is the following :

$$w_i \geq w_*, \forall i \in \mathcal{Q} \tag{4.2}$$

This equation aims at penalizing models that do not have the correct preference directions in the population. We observe through experiments that for such models, imposing a significant weight to i (i.e. a significant value for w_*), mostly deteriorates their CA score. Thanks to this strategy, we intend to keep good models (models in which criteria in $\mathcal{Q}$ possess the correct preference directions) in the population, while pushing out the bad models (models in which criteria in $\mathcal{Q}$ possess the wrong preference directions).

In order to ensure this intention throughout the process, we define a function f_w dependent on the number of iterations of the outer loop (N_o):

$$\begin{array}{rccc} f_w: & [1, N_o] & \longrightarrow & [0, w_*] \\ & it & \longrightarrow & \dfrac{w_*}{N_o - 1}(N_o - it) \end{array} \tag{4.3}$$

with it the current iteration, N_o the maximum number of iterations, and w_* ($w_* \in [0; 1[$) the predefined lower bound weight.

We assign the value $f_w(it)$ at each iteration to the lower bound of w_i, $i \in \mathcal{Q}$ in the LP formulation (i.e. at the iteration it, $w_i \in [f_w(it), 1]$). $f_w(it)$ is decreasing throughout iterations, but imposes a strong constraint on the weights of criteria in $\mathcal{Q}$ at the first iterations. Thus, we relax this constraint progressively with the decrease of $f_w(it)$. Hopefully after restoring the correct preference directions, we do not hinder the algorithm to reach the optimal CA at the end of the learning process.

Otherwise, the same constraint could continuously penalize some good models. Finally, we obtain $f_w(it) = 0$ at the last iteration (when $it = \mathcal{N}_o$), which corresponds to the lower bound of w_i in the original setting [90, 88].

Readjustment of profiles (Step III)

A specific heuristic was developed in [90] for this purpose. This heuristic is well-designed for choosing appropriate profiles values b_i^h that optimize the restoration of assignment examples. We ground our approach in constraining the construction of those profiles.

The principle is to forcefully restrain first, and then progressively enlarge the scope of profiles values throughout the iterations of the outer loop. In the same spirit as in the previous part (optimization of weights), we operate a restriction by shrinking the evaluation scale X_i, $i \in \mathcal{Q}$ to the interval defined by interquantile ranges on X_i throughout the iterations. We define a function that illustrates the refinement operated throughout the process :

$$\begin{array}{rccc} f_\varphi: & [1, N_o] & \longrightarrow & [0, \varphi] \\ & it & \longrightarrow & \dfrac{\varphi}{N_o - 1}(N_o - it) \end{array} \tag{4.4}$$

with it the current iteration, N_o the maximum number of iterations, φ the maximum interquantile order (predefined) and f_φ a function that updates the order of the interquantile range of X_i at each iteration.

The predefined value φ accounts for the order of the interquantile range on X_i. At the first iteration, $f_\varphi(it) = \varphi$. Profiles are constructed similarly as in [90] -

namely in a random manner and by fostering discriminating values - except that they are chosen inside the interquantile range instead of X_i.

Throughout the refinement process, the interquantile range of order $f_\varphi(it)$ stretches incrementally. We aim at forcing the profile b_i^h, $\forall i \in \mathcal{Q}$ to be discriminant enough by restraining the interval at the first iterations.

In such a situation, the preference direction of i plays a crucial role since it could degrade the restoration rate of the learning set of models on which d_i is wrongly assigned. Therefore, we expect these models (with d_i wrongly assigned) to be removed from the population. Thus, we would contribute to preserve good models in the population. At the last iteration, $f_\varphi(it) = 0$, then the construction space of b_i^h amounts to X_i, which is faithful to the original setting [90, 88].

4.4.5 Renewal of the population (Step IV)

At this step, we reach the end of an iteration (of the outer loop) of the metaheuristic [90]. First, we evaluate models by calculating their fitness. We use the same fitness score as described in [90] : CA which is the restoration rate of the learning set. Then, we verify the satisfaction of the stopping condition : there are 2 cases :

- *the stopping condition is fulfilled* : in the initial algorithm ([90]), this state occurs either at the last iteration (the N_o^{th} iteration) or whenever an optimal model (with $CA = 100\%$) is found. In our approach, we end the process only at the last iteration, even if an optimal model has been found before. This decision enables to increase the confidence in the restoration of preference directions which globally increases incrementally.

  After this step, we reach the final step.

- *the stopping condition is not fulfilled* : therefore, the current iteration is not the last, so we renew the population. The objective of this renewal is not only to increase the probability of obtaining better models (in terms of fitness) but also to get rid of wrong models (models where preference directions are wrongly assigned to criteria in $\mathcal{Q}$).

  Given the renewal coefficient σ (predefined), we calculate the rate of renewal rr (see the definition in Section 4.4.2). Here, the course of action differs from the original heuristic that constantly consists in renewing half of the population (i.e. with rr a priori fixed to 0.5).

  Thereafter, we proceed to the renewal of $rr * N_{mod}$ models in the population through a renewal procedure. This procedure consists in generating randomly new models by assigning preference directions according to the distribution on the preference directions of the remaining models in the population. In

other words, the new distribution of preference directions π_i^+ (respectively π_i^-), $\forall i \in \mathcal{Q}$, equals the proportion of remaining models for which i has an increasing (resp. decreasing) preference direction. After the update of the new population, the process restarts with a new iteration of the metaheuristic (outer loop).

4.4.6 Final step (Step V)

This step consists in retrieving the learned model by using a decision rule. Our goal is to choose a model that optimizes the CA rate and retrieves the right preference directions of criteria in $\mathcal{Q}$ all at once.

Therefore, we return the first learned model that is the best one in terms of CA rate (with the appropriate parameters: weights, threshold and profiles), and its corresponding preference directions on criteria in $\mathcal{Q}$.

All the strategies operated in **step II**, and **step III** (contributing to keep in the population models having potentially accurate preference directions) reasonably justify the choice of the best model in terms of CA as the one that carries the right preference directions.

Hence, this model gives the solution of the $IMS_{q|n}$ problem : $(w, b, \lambda, \{d_i : \forall i \in \mathcal{Q}\})$. This ends our second approach for the $IMS_{q|n}$ problem.

In conclusion, the two last sections were dedicated to our contributions for the resolution of the *Inv-MR-Sort* problem with two different methods. The first one (approach based on duplication of criteria) consists of the duplication of criteria into their opposite preference directions in order to select the right preference directions and finally solve the problem with known preference directions ([90]). The second (approach based on the mixed preference directions in the population) enables to handle both gain and cost criteria types in the models population of the initial metaheuristic ([90]). Both approaches allow to learn the preference directions of all the criteria in the model. The duplicated-based approach necessitates two runs of the initial metaheuristic [90] in order to determine first the preference directions, and then the other parameters. The mixed-based approach learns simultaneously the preference directions and the other parameters. In the next section, we examine the performances of both methods with empirical results.

4.5 Experimentations and results

In this section, we present some numerical results to analyze the behaviour of the two proposed algorithms. Thanks to the experiments, we aim at answering the following questions:

- Regarding the computing time, how do the algorithms cope with large datasets ?
- What is the ability of the algorithms to restore an existing dataset when criteria preference directions are latent ?
- How many assignment examples should the learning set contain so that learned models accurately classify new alternatives ?
- How do the algorithms cope with noisy datasets ?

4.5.1 Experimental protocol

We run our experiments on a machine endowed with Ubuntu 18.04.4 LTS (64 bits) with an Intel(R) Xeon(R) Gold 6248 CPU @ 2.5GHz and 376 GB of RAM.

Our tests are characterized by two specific random datasets generated in accordance with a pre-constructed MR-Sort model : datasets with noise and noise-free datasets. The noise introduced in the datasets consists of alternatives falsely assigned to wrong categories. Therefore, datasets with noise could be assimilated to real preferences data because real data are subject to flaws. The workflow of the experiments is illustrated below (see Figure 4.4).

At the beginning, the triplet (n, q, p) defines the first set of inputs of the process, such that n is the number of criteria, q the number of criteria with unknown preference directions and p the number of categories. These entries are used to construct an MR-Sort model thanks to the *Generator of MR-Sort models* (see Figure 4.4).

The generation of models works as follows. We uniformly generate $p-1$ random values in [0,1] for the profiles values of each criterion and order them as the following : $b_i^1 \leq b_i^2 ... \leq b_i^{p-1}, \forall i \in \mathcal{N}$. In order to assign values to weights, we first draw $|\mathcal{N}| - 1$ numbers in [0,1]. These numbers added to 0 and 1 are ranked in ascending order. The difference between each successive pairs in the ranking forms the values of criteria weights. We randomly generate λ in]0.5,1[. Therefore we have created a model $\mathcal{M}^0$ with $(\langle b \rangle, w, \lambda)$, that is the ground truth. With the ground truth, we can judge the learning abilities of our algorithms on a reliable basis.

Then, we create a set of randomly generated alternatives $A^\star$; each generated alternative is a tuple of n values uniformly drawn from [0.05;0.95]. We classify such alternatives according to $\mathcal{M}^0$ in such a manner to obtain a balanced distribution of alternatives over categories. The set of the obtained assignments is called the learning set L. Similarly, we also generate directly a set of randomly generated alternatives A^{tests} (generally larger than $A^\star$) and classify alternatives according to $\mathcal{M}^0$: the resulted assignments represent the test set. We are then able to induce noisy learning set using L. The process is the following. Considering $L = (A^\star, f)$,

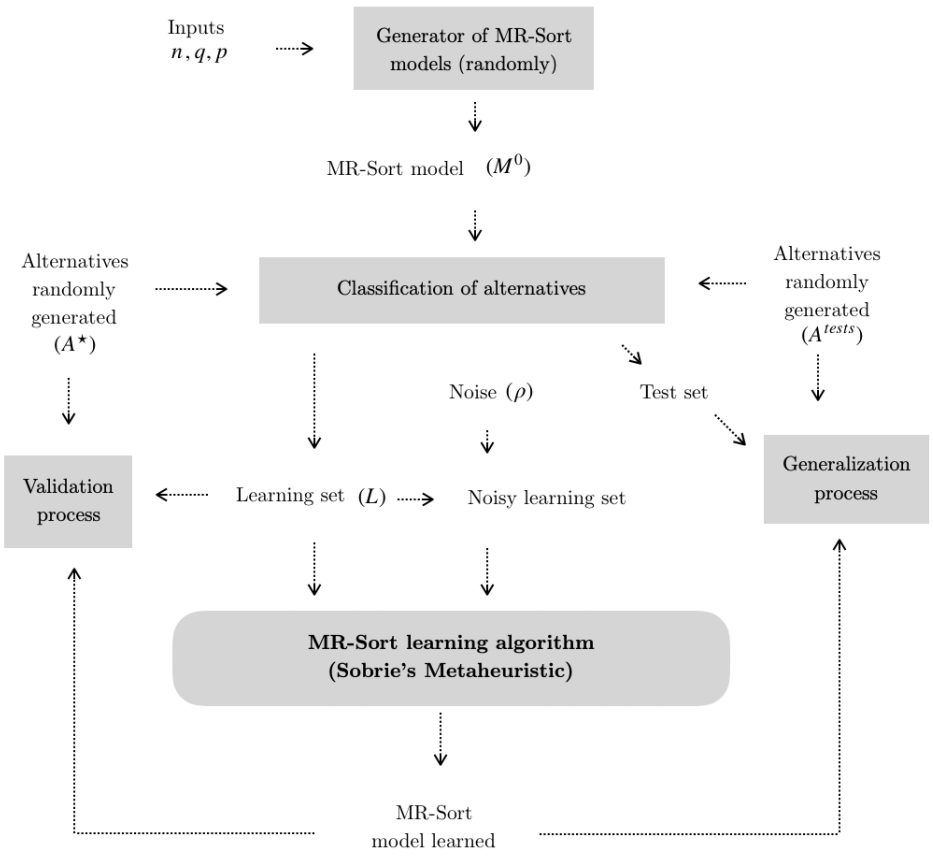


Figure 4.4: Experimental workflow

$f : A^\star \longrightarrow C$ (with C the set of categories) the learning set generated from $\mathcal{M}^0$, we randomly choose $\rho \times |A^*|$ examples of the learning set that will be subject to error. For these alternatives, we change their assignment to an adjacent category.

Once $\mathcal{M}^0$ and L are well defined, we choose not to disclose the preference directions of q out of n criteria in order to formulate the problem $IMS_{q|n}$. Without loss of generality, we consider all n criteria as gain criteria and q criteria out of n whose preference directions are considered as hidden. Therefore, each criterion of the q criteria is potentially a gain or a cost criterion. Then, the learning set is used as input for the learning algorithm. The algorithm output an MR-Sort model learned that is used both as input for the validation and the generalization process.

The validation process aims at confronting the learning set $L = (A^\star, C)$ and the assignments obtained by classifying $A^\star$ with the new learned model. Thus, the classification accuracy in validation (CA_v) is computed : this is the rate of the correctly assigned alternatives by the new learned model compared to the assignments obtained with $\mathcal{M}^0$ on $A^\star$. The generalization process enables to

compare the assignments obtained on A^{tests} (the test set) with $\mathcal{M}^0$ and those obtained with the new learned model. Thus, we compute the classification accuracy in generalization that is the rate of the correctly assigned alternatives by the new learned model compared to the assignments obtained with $\mathcal{M}^0$ on A^{tests}.

The experimental parameters and the values taken into account for generating a dataset are : the number of criteria $|\mathcal{N}| \in \{5, 7, 9, 11\}$, the number of latent criteria $q \in \{1, 2, \ldots, |\mathcal{N}|\}$, the number of categories $p \in \{2, 3, 4, 5\}$, the learning set size $|A^*|$ in $\{100, 250, 500, 750, 1000, 1250, 1500\}$, and noise rate ρ in $\{0, 0.05, 0.1, 0.15, 0.2, 0.25\}$. The test set size is 10000 alternatives.

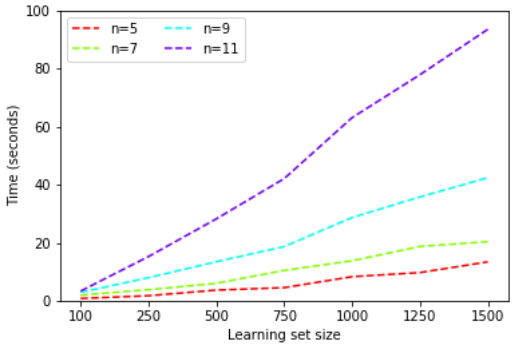
The value ρ represents the percentage of alternatives whose assignment are subject to errors. In order to build learning sets with noise, we assign the erroneous alternatives to an adjacent category of the true category. For instance, considering three categories, if the true category of the alternative a is C^2, the erroneous category can be C^1 or C^3. If the true category of a is C^3, the erroneous category is C^2.

We execute the two approaches as described in this chapter for the resolution of the $IMS_{q|n}$. M^0 represents a ground truth on which the learned models (with the algorithms) are validated. We expect a good rendering of solutions since M^0 comes from a real MR-Sort model, at least with noise-free datasets. In our experiments, we use 100 samples of MR-Sort $\mathcal{M}^0$ in order to obtain mean values as performances of learned models in terms of computation time, restoration rate of the learning set (CA_v), restoration rate of the test set (CA_g), and restoration rate of the preference directions. We express the latter rate with two metrics in order to appreciate in detail the restoration of preference directions : PDR_{all} which is the rate of restoring all the preference directions at once, while PDR_{one} is the proportion of restoring one preference direction on average.

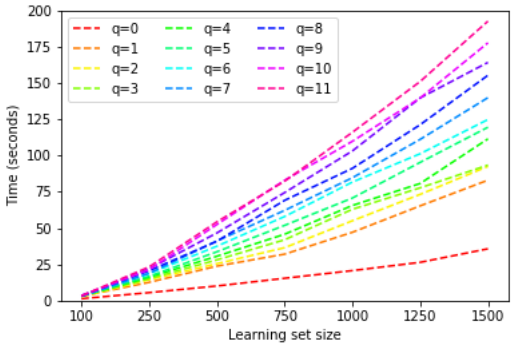
4.5.2 Experimental study for the duplicated-based approach

Execution time and memory requirements

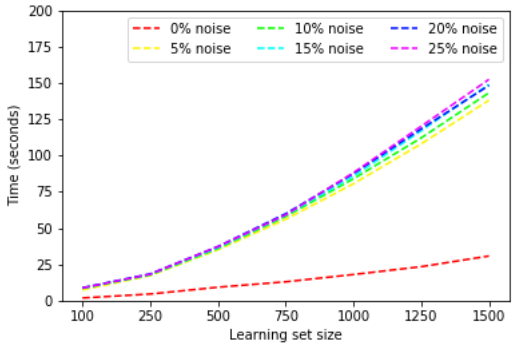
The execution time of this approach increases in function of the increase of the 5 parameters : the learning set size, the number of criteria n, the number of latent criteria q, the number of categories p and the level of noise ρ. Figure 4.5a shows a linear trend of the evolution of the time when the number of criteria varies in the considered problems. The time is affordable ; it takes less than 100s to solve $IMS_{3|11}$ with 2 categories. When the number of latent criteria varies (see Figure 4.5b) with different learning set sizes (from 250 to 1500), we also observe a linear shape of the evolution of execution time. The problem $IMS_{11|11}$ involving 2 categories requires almost 200s of execution time.



(a) Impact on the number of criteria : MR-Sort model with 3 latent criteria, 2 categories and noise-free learning sets per number of criteria (n) and learning set size



(b) Impact on the number of latent criteria : MR-Sort model with 11 criteria, 2 categories and noise-free learning sets per number of latent criteria (q) and learning set size



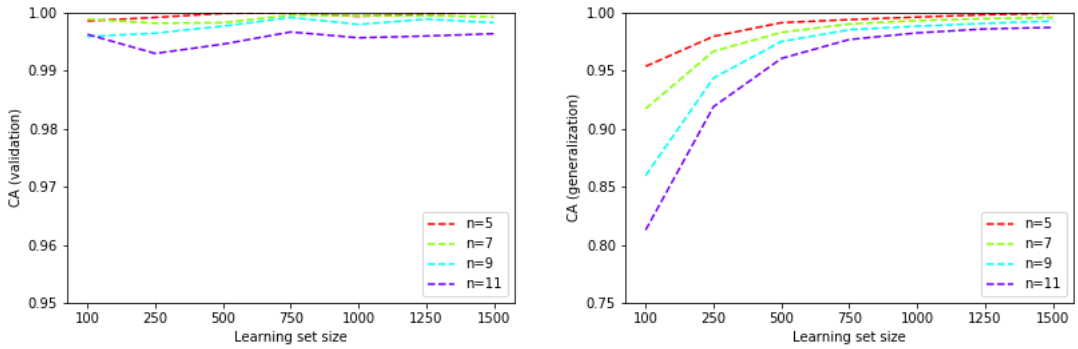
(c) Impact on the noise percentage : MR-Sort model with 7 latent criteria, 7 criteria and 2 categories per noise percentage in the learning set and learning set size

Figure 4.5: Results of the first approach regarding the execution time influenced by the problem parameters

The impact of the noise on the execution time is noticeable. On the one hand, the presence of noise considerably increases the execution time (see Figure 4.5c). The gap between the two situations is widening all the more with large learning set sizes (the gap rises from 10s with 250 alternatives to 125s with 1500 alternatives in the learning set). On the other hand, the increase of the noise is not a disadvantage regarding the execution time since the Figure 4.5c shows very similar behaviours as soon as the noise percentage is over 5%.

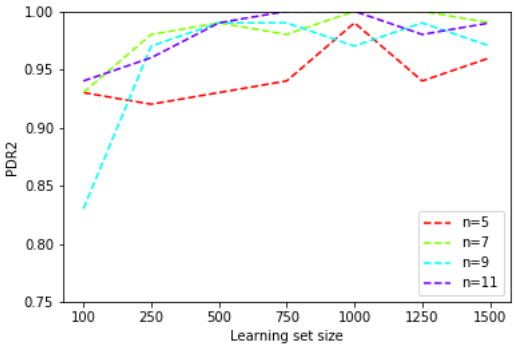
To conclude this analysis, let us notice that the execution time on a laptop rises to 545 seconds when considering the problem $IMS_{1|11}$, with 5 categories and 1500 alternatives in the learning set without noise. However, the required memory space is less than 50MB.

Varying the number of criteria (n)



(a) Classification accuracy (CA) of the learning set (validation)

(b) Classification accuracy (CA) of the test set (generalization)



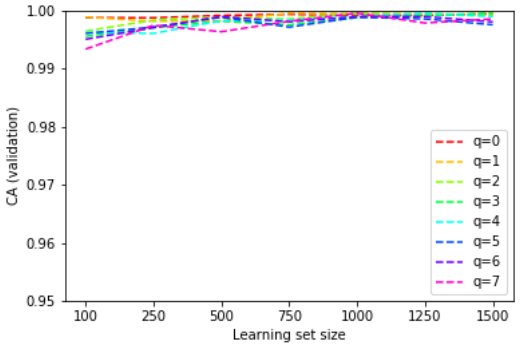
(c) Preference direction restoration (PDR_{one})

Figure 4.6: Results of the first approach for problems involving one latent criterion, 2 categories and noise-free learning sets per number of criteria (n) and learning set size

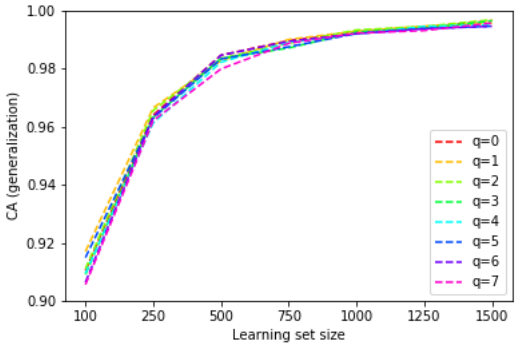
First, the duplicated-based approach was executed for a series of problems involving different number of criteria with 1 latent criterion, 2 categories and considering noise-free datasets. These executions were performed on increasing learning set sizes. The classification accuracy of the learning set is excellent

(between 0.99 and 1) regardless the number of criteria in the problem (see 4.6a). The ability of the algorithm to restore assignment examples decreases with the increase of the number of criteria in the problem (see Figure 4.6b). However the classification accuracy surges to 95% with 500 alternatives in the learning set and then converges towards 1, considering the problem $IMS_{1|11}$. The behaviour of the algorithm is less obvious regarding the preference direction rate (PDR_{one}) (see Figure 4.6c). Nevertheless, the PDR_{one} exceeds 0.9 with at least 250 alternatives in the learning set.

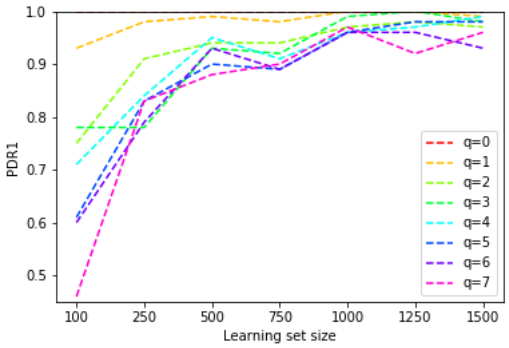
Varying the number of criteria with latent preference directions (q)



(a) Classification accuracy (CA) of the learning set (validation)



(b) Classification accuracy (CA) of the test set (generalization)



(c) Preference direction restoration (PDR_{all})

Figure 4.7: Results of the first approach for problems involving 7 criteria, 2 categories and noise-free learning sets per number of latent criteria q and learning set size

Second, we test the first approach on instances involving 7 criteria, 2 categories, with noise-free and increasing datasets sizes. We vary the number of latent criteria. The classification accuracy of the learning set is flawless (ranges between 99% and 100%) regardless the number of latent criteria and the learning set size (see Figure 4.7a). The classification accuracy in generalization converges towards 1 regardless the number of latent criteria (see Figure 4.7b). The algorithm behaves exactly in the same manner independently of the number of latent criteria. In order words, the increase of unknown preference directions does not bring additional difficulties in restoring of assignments. With only 250 alternatives in the learning set, 96% of new assignments are restored. The PDR_{all} increases with the size of the learning set and converges differently depending on the number of latent criteria, towards 1 (see Figure 4.7c). Globally, the algorithm behaves in general better with less latent criteria. With 750 alternatives in the learning set, the algorithm performs with a PDR_{all} more than 0.85 considering 7 latent criteria. This is by far better than the random probability of retrieving the preference directions of 7 latent criteria that is $1/2^7$ (≈ 0.008).

Varying the number of categories p

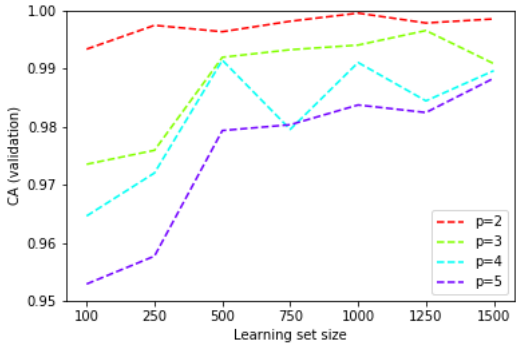
Third, we test the first approach on instances with 7 criteria, 7 latent criteria, with noise-free and increasing datasets sizes, and different number of categories. The CA of the learning set is satisfactory since it improves with the increase of the learning set size and reaches at least 0.98 with at least 500 assignment examples independently of the number of categories (see Figure 4.8a). The classification accuracy (in generalization) decreases progressively according to the increase of the number of categories in the problem (Figure 4.8b). Despite this, the CA_g still increases when the learning size set becomes greater and greater. With 5 categories, and 500 assignment examples in the learning set, the algorithm is able to restore more than 90% of new assignment examples.

The preference direction restoration (PDR_{all}) degrades moderately with more than 2 categories, but increases with the size of the learning set (Figure 4.8c). The PDR_{all} reaches at least 0.8 with more than 500 alternatives in the learning set regardless the number of categories.

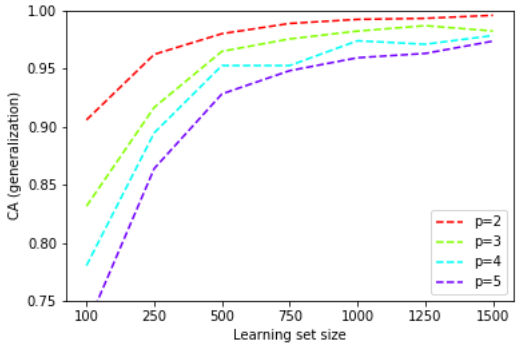
Considering noisy learning sets ρ

The fourth test of the first approach concerns the case with 7 criteria, 7 latent criteria, 2 categories and considering the presence of noise (ρ) in the learning set.

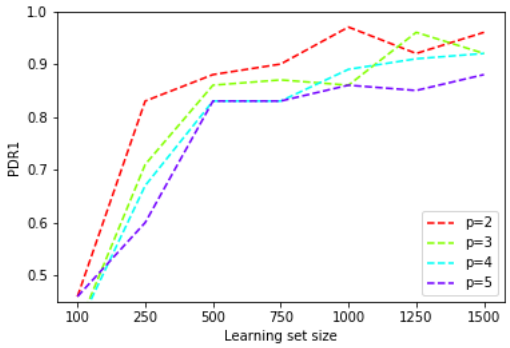
Figures 4.9a and 4.9b teach us the ability of our approach to restore more than $100^*(1-\rho)\%$ of new assignments examples, which means that the algorithm is



(a) Classification accuracy (CA) of the learning set (validation)



(b) Classification accuracy (CA) of the test set (generalization)



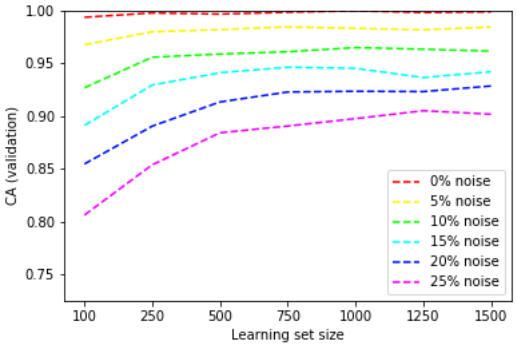
(c) Preference direction restoration (PDR_{all})

Figure 4.8: Results of the first approach for problems involving 7 criteria, 7 latent criteria and noise-free learning sets per number of categories p and learning set size

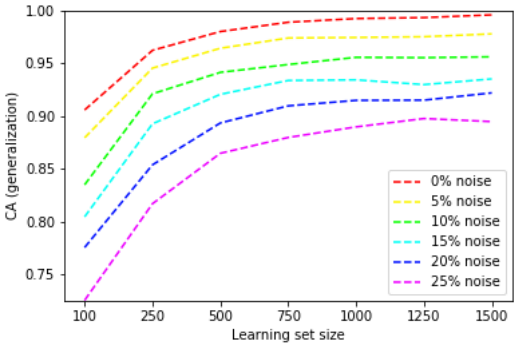
fairly robust. As an example, the CA_v attained 85% with 25% of noise when the learning set size is 500, which means that the algorithm is able to find the true assignments of some wrongly assigned alternatives (at least 10% of the learning set).

The CA of the learning set still increases proportionally to the presence of error when the learning set size increases. In addition, the CA (in generalization) preserves a similar proportionality.

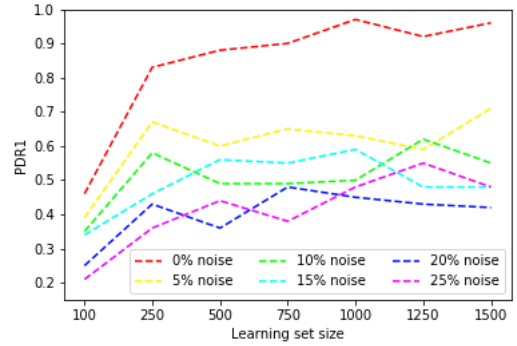
Figures 4.9c and 4.9d show that the impact of more noise ($\rho > 0$) is not so strong on the ability of the algorithm to restore preference directions. The PDR_{one} increases with difficulty along with the learning set size once in the presence of noise.



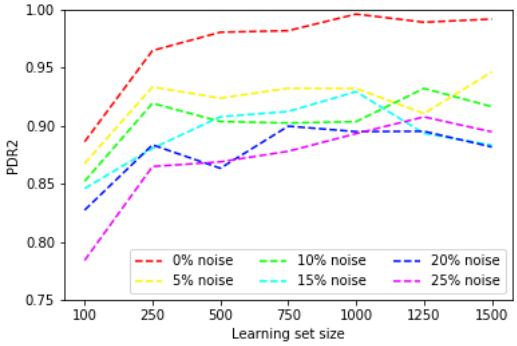
(a) Classification Accuracy (CA) of the test learning (validation)



(b) Classification Accuracy (CA) of the test set (generalization)



(c) Preference direction restoration (PDR_{all})



(d) Preference direction restoration (PDR_{one})

Figure 4.9: Results of the first approach for a problem involving 7 criteria, 7 latent criteria, 2 categories per noise percentage in the learning set and learning set size

4.5.3 Results of the mixed-based algorithm

Execution time

Here again, the execution time of the approach increases in function of 5 parameters : the learning set size, the number of criteria n, the number of latent criteria q, the number of categories p and the level of noise. As an example, Figure 4.10 shows us one of the highest execution time recorded in our tests. It concerns the learning problem $IMS_{5|7}$ with $p = 5$ involving a noise-free learning set of 1500 entries : the execution rises to 957s.

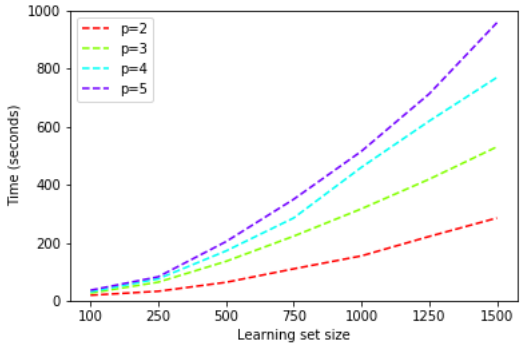


Figure 4.10: Time execution of the second algorithm for a problem involving 7 criteria, 5 latent criteria, and noise-free learning sets per number of categories and learning set size

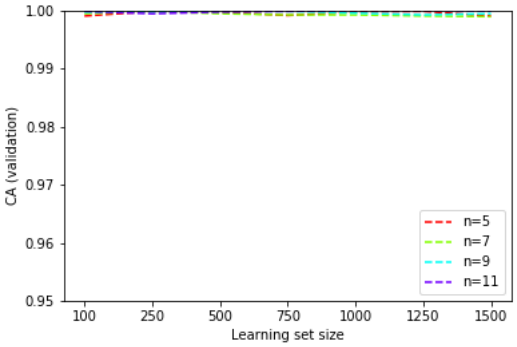
Varying the number of criteria (n)

First, we executed this approach for a series of problems involving different criteria with 1 latent criterion, 2 categories and considering noise-free datasets and multiple criteria ($n \in \{5, 7, 9, 11\}$).

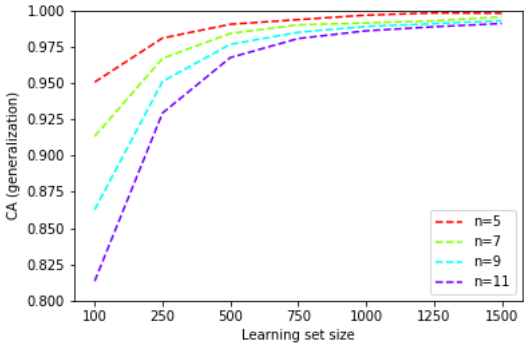
We note in Figure 4.11a a perfect restoration of the learning set (CA=1) regardless the number of criteria considered in the problem. However, for small learning set size (for example 100 alternatives) we cannot hasten to deduce a similar case in generalization (see Figure 4.11b). As a matter of fact, regardless the number of criteria n, the lowest CA score is recorded with small learning sets size (with $n = 11$ and 100 alternatives in the learning set, $CA_v \approx 0.81$). As the assignment examples in the learning set increase, this score converges towards 1, and the less criteria in the problem, the better the CA is.

The results of Figure 4.11c show a general upward trend of the preference direction rate (PDR_{one}) (in particular for $n = 11$). Indeed with at least 750 alternatives in the learning set, the preference direction is perfectly restored (PDR_{one}=1) in $IMS_{1|11}$. Let us notice that PDR_{one} is similar in this case to PDR_{all} since there is only one latent criteria considered here. The algorithm performs globally better with more than 5 criteria and with at least 250 assignment examples in the learning set. It seems that few criteria in the model (considering this configuration) implies a less constrained model, which is subject to more uncertainty on preference directions.

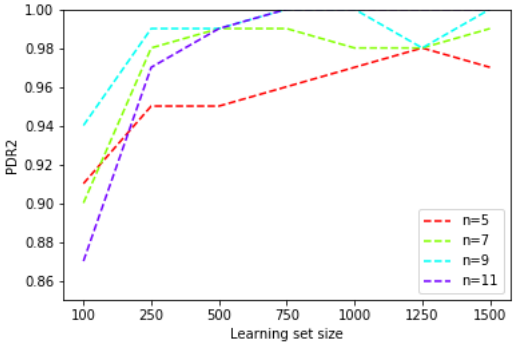
Varying the number of criteria with latent preference directions (q)



(a) Classification accuracy (CA) of the learning set (validation)



(b) Classification accuracy (CA) of the test set (generalization)

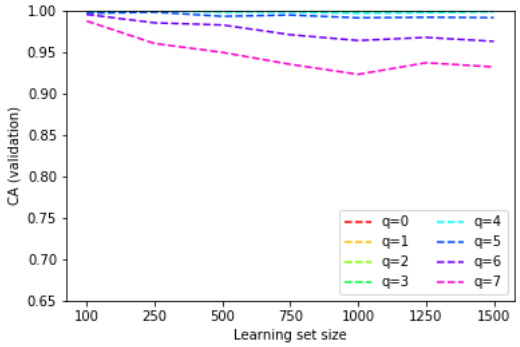


(c) Preference direction restoration (PDR_{one})

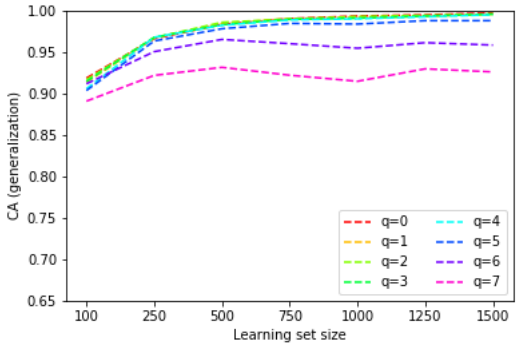
Figure 4.11: Results of the first approach for problems involving 1 latent criterion, 2 categories and noise-free learning sets per number of criteria (n) and learning set size

Our second series of test concerns instances of 7 criteria, 2 categories, with noise-free datasets and varied the number of latent criteria, tested on increasing learning set sizes. The classification accuracy of the learning set clearly degrades when the number of latent criteria is greater then 6 ($q \geq 6$) while the remaining ($q < 6$) stays constant (between 0.99 and 1) when the learning set size increases (Figure 4.12a). Interestingly, this tendency is not followed in the generalization case (see Figure 4.12b), where we do not observe such degradation of the CA score for $q \geq 6$, but rather a constant trend (≈ 0.95 for $q = 6$ and ≈ 0.9 for $q = 7$). The remaining increases slightly and converges towards 1.

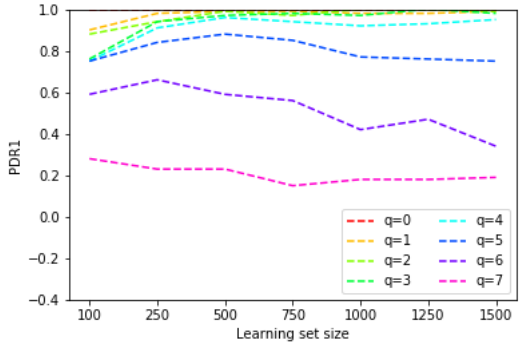
Concerning the preference direction rate (PDR_{all}) (see Figure 4.12c), the algorithm behaves with some difficulties when $q \geq 5$: the PDR_{all} approximates 0.8



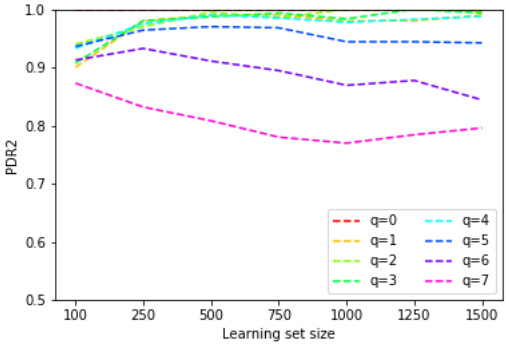
(a) Classification accuracy (CA) of the learning set (validation)



(b) Classification accuracy (CA) of the test set (generalization)



(c) Preference direction restoration (PDR_{all})



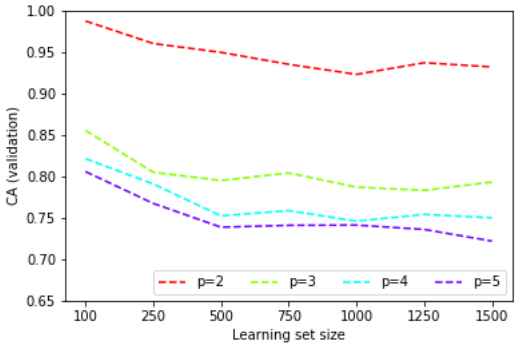
(d) Preference direction restoration (PDR_{one})

Figure 4.12: Results of the first approach for problems involving 7 criteria, 2 categories and noise-free learning sets per number of latent criteria q and learning set size

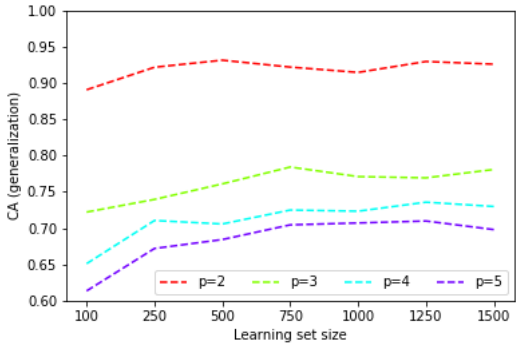
on average for $q = 5$; it declines from 0.6 to less than 0.4 for $q = 6$ and stabilizes around 0.2 for $q = 7$. In contrast, for $q \leq 5$ PDR_{all} the preference direction restoration is more satisfactory (globally above 0.9 with at least 250 assignment examples in the learning set).

The PDR_{one} reads as an optimistic view on the preference direction rate. As a matter of fact, the result in Figure 4.12d is quite different from the previous Figure particularly for $q = 7$ where PDR_{one} approximates 0.8. It indicates that the algorithm restores on average more than 5 preference directions out of 7 (since $0.8 > \frac{5}{7}$) on average and succeeds only in 20% ($PDR_{all} = 0.2$) of cases to restore all the 7 preference directions of the latent criteria.

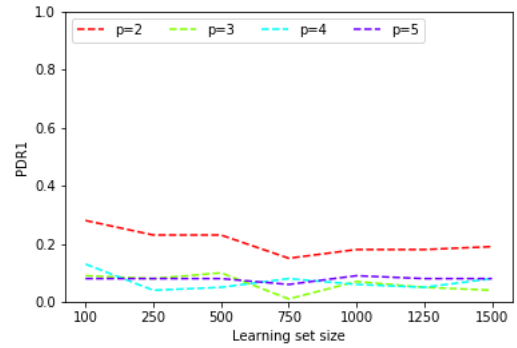
Varying the number of categories



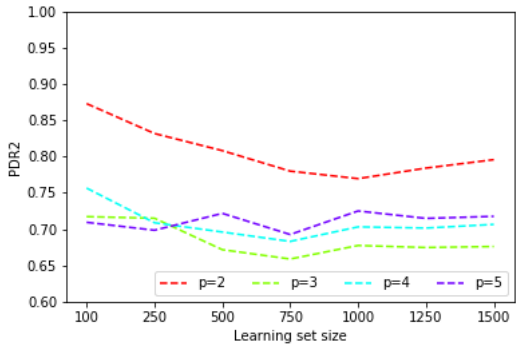
(a) Classification accuracy (CA) of the learning set (validation)



(b) Classification accuracy (CA) of the test set (generalization)



(c) Preference direction restoration (PDR_{all})



(d) Preference direction restoration (PDR_{one})

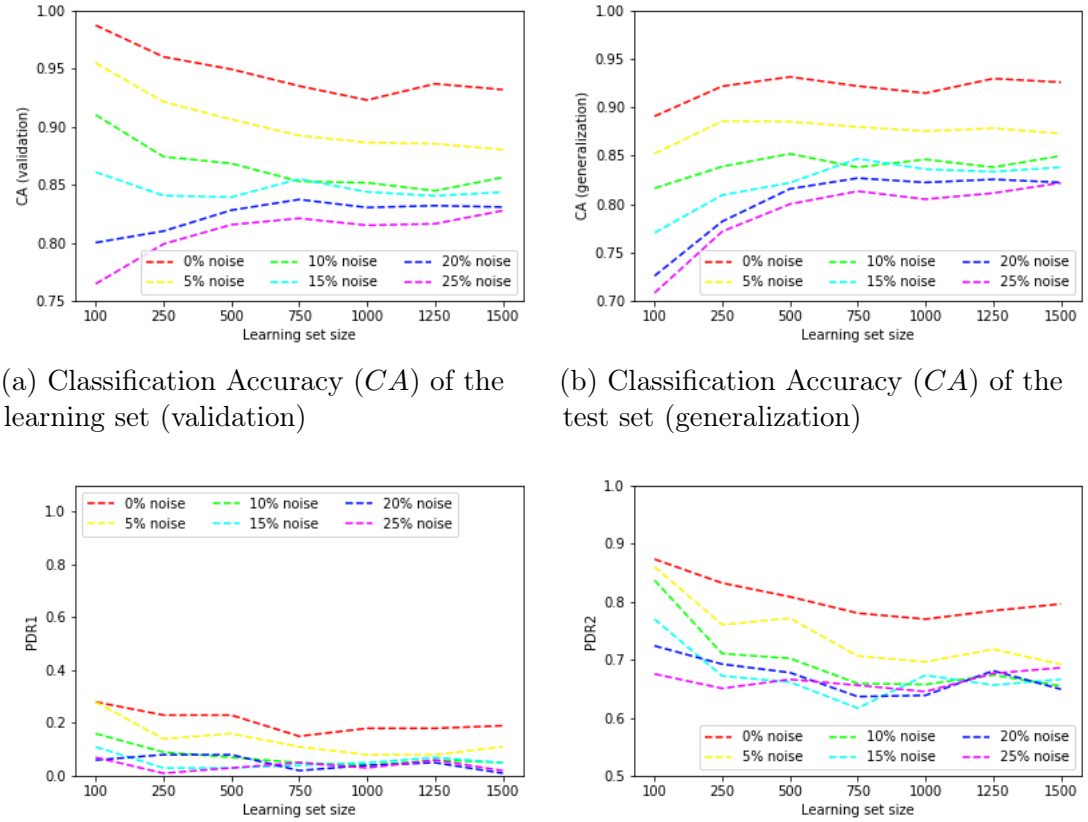
Figure 4.13: Results of the first approach for problems involving 7 criteria, 7 latent criteria and noise-free learning sets per number of categories p and learning set size

Third, we tested the second approach on instances of 7 criteria, 7 latent criteria, with noise-free datasets and varied number of categories. The classification accuracy of the learning set (see Figure 4.13a) decreases moderately with the increase of the learning set size. There is a noticeable performance gap of 0.15 between the case with 2 categories ($p = 2$), which averages 0.95 and the three other cases ($p \in \{3, 4, 5\}$) which are close (their CA_v score are below 0.8 on average). This gap is preserved even in generalization (see Figure 4.13b). However the CA in generalization weakly improves as the learning set size increases.

Regarding Figure 4.13c, the PDR_{all} is very low and globally averages 0.2 for $p = 2$ and worsen with $p > 2$ (the PDR_{all} is globally below 0.1). The algorithm

behave better regarding the PDR_{one} rate (around 0.85 and 0.65, see Figure 4.13d).

Considering noisy learning sets



(a) Classification Accuracy (CA) of the learning set (validation)

(b) Classification Accuracy (CA) of the test set (generalization)

(c) Preference direction restoration (PDR_{all})

(d) Preference direction restoration (PDR_{one})

Figure 4.14: Results of the first approach for a problem involving 7 criteria, 7 latent criteria, 2 categories per noise percentage of the learning set and learning set size

The fourth test of the second approach concerns the case with 7 criteria, 7 latent criteria, 2 categories and considering the presence of the noise (represented by the percentage of erroneous alternatives ρ) in the learning set. Figure 4.14a shows us slight decreases of the CA (of the learning set) when $\rho < 0.15$ while slight increases when $\rho > 0.15$. Globally the CA score ranges between 0.95 and 0.8 with at least 250 alternatives in the learning set. In generalization, the CA slowly rises with the increase of the learning set size and lowers with the increase of noise in the learning set (Figure 4.14b). With 500 alternatives in the learning

set, the classification accuracy is quite satisfactory even with 25% of noise with $CA \geq 0.8$. With more $\rho \geq 0.10$, the influence of the noise becomes minor for large dataset sizes. This is emphasized with the PDR_{one} (see Figure 4.14d) where the restoration of preference directions are close to each other for $\rho \geq 0.10$. Alike the bad performance in PDR_{all} obtained by this algorithm for multiple categories, the situation is quite similar in the presence of noise (see Figure 4.14c). The preference direction rate roughly ranges between 0 and 0.2.

4.5.4 Comparing the two approaches

Useful insights have been found through the experiments of the two approaches.

Execution time

First, the comparison of the two approaches gives the advantage to the first approach for the execution time. Indeed, the first approach only requires $\approx$ 55s while the second method requires $\approx$ 150s in order to solve the problem $IMS_{5|7}$ with 2 categories, and 1000 noise-free alternatives in the learning set. This significant difference resides in the difference in the settings of the two algorithms parameters. Indeed, the second approach takes into account 50 models in the population instead of 10 for the first approach. In addition, the number of iterations N_o is forcefully attained in the second method while in the first method, as soon as a model with optimal restoration of the learning set was found, the iterations stopped.

Although the first method involves two runs of the Sobrie's metaheuristic [90], the computational time is fairly affordable.

Concerning this method, the worst-case scenario considered in our tests (which is the learning of $IMS_{7|7}$ with $|A^\star| = 1500$ and 4 categories considering 25% of noise in the learning set) indicates that the algorithm runs less than 15 minutes and the memory space needed is less than 50MB (using a machine endowed with 2,3 GHz Intel Core i5 and 8Go of RAM). In any case, the execution time strongly depends on the number of iterations of the outer loop of the algorithm (N_o).

Classification Accuracy and preference direction restoration when varying the number of criteria, latent criteria and categories

First, globally the first method is better than the second method. It seems counter-intuitive with respect to the results (comparing for instance Figure 4.6a against Figure 4.11a, Figure 4.6c against Figure 4.11c). However, not only the advantage of the second method over the first method is minor, but also when q is great in the considered problem, the advantage goes for the first method.

Concerning the variation of the number of latent criteria, the advantage goes to the first method. It is apparent that the first method improves the CA_v/CA_g and PDR_{one}/PDR_{all} regardless of the number of latent criteria while the second method degrades the CA and the PDR_{one}/PDR_{all} when the number of latent criteria become large.

Regarding the variation of number of the categories, the first approach clearly behaves better than the second in terms of classification accuracy (with at least 1000 alternatives in the learning set, the minimum CA score of the learning set exceeds 0.95 in the first method, while the maximum score never exceeds 0.95 in the second method). In addition, in terms of preference direction rate (PDR_{all}), while the first method reaches 0.8 with at least 500 alternatives, the second method never exceeds 0.3.

Considering the first method, the algorithm succeeds in retrieving preference directions as well as the other MR-Sort parameters of our problem, with noise-free learning sets and 2 categories. Indeed, the classification accuracy of the learning set is excellent ($\approx 99\%$) regardless of the number of latent criteria and the number of criteria considered in our tests ($n \in \{5, 7, 9, 11\}$). In generalization, the behaviour of the algorithm is similar. As a matter of fact, the restoration of new assignments is as good as the restoration of the Sobrie's problem (which is $IMS_{0|n}$) independently of the number of latent criteria and the learning set size. The approach also succeeds in restoring preference directions since the restoration converges quite quickly towards 1 (with $|A^\star| = 1500$). Moreover, additional results show that PDR_{one} - which considers the restoration of one preference direction on average - reaches 95% for $IMS_{q|7}$, with $0 \leq q \leq 7$ with only 250 assignment examples in the learning set.

Classification Accuracy and preference direction restoration with noisy learning datasets

The ability to restore assignments examples as well as latent preference directions is greater with the first method than with the second method. The CA (both in validation and generalization) as well as the PDR_{one}/PDR_{all} increase regardless the noise percentage in the first method and therefore outperforms the second method, on which these indicators often decrease.

Regarding the first method and as expected, in the presence of noisy data in the learning set, the classification accuracy in generalization is reduced in function of the increase of the noise introduced in the learning set. This is all the more linked with the difficulty to learn the preference direction as soon as we deal with noisy data in comparison to the case without noise (true for both approaches). Nevertheless, the first approach takes advantage of the increase of the learning set

size since it still succeeds at restoring more assignments (CA in generalization) with more assignment examples in the learning set.

4.6 Conclusion

We have considered the MR-Sort model in the case where the direction of preference of criteria is unknown. We have proposed a solution which extends the heuristic method introduced by Sobrie [90] to this case. For a given learning set, our algorithm estimates the unknown preference directions as well as the parameters of the MR-Sort model.

Our contribution is in line with [90, 92], which concerns the learning MR-Sort models from monotone data. Indeed, we took advantage of this work and bring some adaptations in order to deal with our problem. We proposed two approaches for the resolution of the problem. Although each of them have advantages and shortcomings, the first method appears to be the most effective. The experiments results of our approaches on noisy data justifies its suitability for real-world learning problems, where sometimes gain and cost criteria are insufficient to describe precisely the type of preference order. In our next chapter, we deal with the learning of MR-Sort models with single peaked preferences.

Chapter 5

The MR-Sort model with single-peaked preferences

Contents

5.1 Introduction and motivation

In this chapter, we investigate a new preference feature for the learning of partially monotone data : single-peaked preferences. Single-peaked preferences have not been considered yet in the previous works concerning outranking models in particular MR-Sort models. In the following, we illustrate through an example how single-peaked is expressed. The Section 5.2 describes formally the characterization of single-peaked preferences through the lens of the Non Compensatory Sorting (NCS), and therefore with MR-Sort since MR-Sort is a sub-case of NCS. Finally, in the Section 5.3, we give some connections between single-peaked preferences and monotone preferences.

Motivating example : Renting a new apartment

Bob got a new job in a new city and is looking for renting an apartment there. The real estate agent is determined to help him in his choice ; so he presents to Bob 6 proposals (a_1 to a_6). It is not convenient for him to visit all the houses, so he must select the most relevant proposals for a visit. Therefore, Bob and the real estate agent agree to sort the proposals into two groups : interesting ones that will be granted a visit (**I**), and rejected proposals (**R**). The real estate agent probes Bob for discerning the criteria upon which the sorting must be done. Four criteria are considered:

- the price (€) of the apartment noted c_1. The criterion c_1 is a cost criterion ($\downarrow$) i.e. for Bob, the lower the price the better is his preferences,
- the surface (m^2) of the house noted c_2. The criterion c_2 is a gain criterion ($\uparrow$) i.e. for Bob, the greater the surface, the better is his preferences,
- the energy class noted c_3 (expressed by 7 letters from A the best category to G the worst one). The criterion c_3 is a cost criterion ($\downarrow$) considering the following order the the evaluation scale : $A < B < C < D < E < F < G$.
- the distance (in km) between his future house and his future workplace, noted c_4.

Regarding the last criterion, for some reason, Bob does not want to live too close to his working place ; neither does he want to live far from his workplace.

Therefore there exist an ideal distance for which, a lesser or a greater distance would decrease Bob's preference. We call this type of criterion (c_4) a *single-peaked criterion* ($\nearrow\!\!\nwarrow$) regarding the ideal distance, which represents the *peak* - his most preferred value in this criterion. Here is the performance table :

c_i	c_1 (€)	c_2 (m^2)	c_3	c_4 (km)	**cat(a)**
a_1	500	40	F	7 km	**I**
a_2	700	20	C	5 km	**I**
a_3	900	30	D	12 km	**I**
a_4	1000	50	B	3 km	**R**
a_5	600	40	E	10 km	**R**
a_6	800	60	A	15 km	**R**

Table 5.1: Performance table of flat proposals

The real estate, who is a budding decision analyst, uses the majority sorting rule (MR-Sort) to model the preferences of Bob. Therefore, he scribbled the following table. More precisely, he determined the profile values (b_1,b_2,b_3,b_4), the weight values (w_1,w_2,w_3,w_4) and the threshold majority λ.

c_i	c_1 (€)	c_2 (m^2)	c_3	c_4 (km)	**cat(a)**
a_1	500	40	F	7 km	**I**
a_2	700	20	C	5 km	**I**
a_3	900	30	D	12 km	**I**
a_4	1000	50	B	3 km	**R**
a_5	800	40	E	10 km	**R**
a_6	600	20	A	15 km	**R**
d_i	$\downarrow$	$\uparrow$	$\downarrow$	$\nearrow\nwarrow$	
b_i	700	30	D	[5-12]	
w_i	0.3	0.2	0.2	0.3	
$\lambda = 0.8$					

Table 5.2: The MR-Sort model parameters of the real estate agent for Bob's decision problem

The model assumed by the real estate agent is an example among many that describes the preferences of Bob. In this modelling, he determined the profile b_4 as the interval $[5km - 12km]$, which is compatible with Bob's preferences. He finally thinks that the Bob's ideal distance between his home and his workplace ranges between 5 and 12 km. Therefore, values comprised inside this interval, contribute positively to his preferences and are accountable for the sorting of proposals in the MR-Sort rule. We describe more formally how single-peaked preferences are formalized in NCS and MR-Sort models in the Section 5.2.

Remark

The assumption of monotonicity made on some criteria could even be relaxed under some situations. We could assume that the DM (the buyer) ignores the environment where he is looking for a house. Therefore, he is not sure that a lower price would guarantee a pleasing environment; neither a greater surface would be an ideal choice since a small one makes it easy to clean. In this case, it could be more accurate to consider the price and surface criteria as single-peaked criteria. Therefore, for the buyer, a higher price is prohibitive, and a lower price is suspicious. In the same manner, he is neither comfortable with a small surface nor a gigantic house surface.

Constructing the preferences in this way enables us to consider hidden features (here, living environment and household labors) that could not be perceived otherwise, especially when the buyer does not have any additional information.

Single-peaked preferences have been largely studied in the field of Decision Theory, and in particular voting systems. In the next section, we ground our formalization development of single-peaked preferences regarding our context, on the definition of single-peaked given by Escoffier *et al.* [41] (see Chapter3).

5.2 Characterization of single-peaked preferences for MR-Sort models

5.2.1 Rewriting MR-Sort with approved sets

As we mention before, MR-Sort is a variant of NCS, where the majorities are necessarily additive. Indeed, we can use NCS notations, particularly the notion of criteria subsets families ($\mathcal{F}$) to formulate the MR-Sort rule.

In order to achieve this task, let us notice that the families of sufficient coalitions of criteria become all equal $\mathcal{F}^2 = ... = \mathcal{F}^p = \mathcal{F}$ as soon as we deal with MR-Sort. The majority is now defined through the sum of weights attached to criteria, and a threshold $\lambda \in [0, 1]$. We have $\mathcal{F} = \{F \subseteq \mathcal{N} : \sum_{i \in F} w_i \geq \lambda\}$, with $w_i \geq 0$, $\sum_i w_i = 1$.

In addition, as the finite set of possible values on criterion i, $X_i = [min_i, max_i] \subset \mathbb{R}$, the order on $\mathbb{R}$ induces a complete preorder $\succcurlyeq_i$ on X_i. Hence, the sets of approved values on criterion i, $\mathcal{A}_i^h \subseteq X_i$ $(i \in \mathcal{N}, h = 2...p)$ are defined by $\succcurlyeq_i$ and $b_i^h \in X_i$ the minimal approved value in X_i at level h: $\mathcal{A}_i^h = \{x_i \in X_i : x_i \succcurlyeq_i b_i^h\}$. In this way, $b^h = (b_1^h, \ldots, b_n^h)$ is interpreted as the frontier between categories C^{h-1} and C^h; $b^1 = (min_1, ..., min_n)$ and $b^{p+1} = (max_1, ..., max_n)$ are the lower frontier of C^1 and the upper frontier of C^p. The assignment rule is defined bellow, for all $x \in X$, where $\mathcal{A}_i^1 = X_i$, $\mathcal{A}_i^{p+1} = \emptyset$, $\mathcal{F}^1 = \mathcal{P}(\mathcal{N})$, and $\mathcal{F}^{p+1} = \emptyset$.

$$x \in C^h \quad \text{iff} \quad \{i \in \mathcal{N} : x_i \in \mathcal{A}_i^h\} \in \mathcal{F}^h \text{ and } \{i \in \mathcal{N} : x_i \in \mathcal{A}_i^{h+1}\} \notin \mathcal{F}^{h+1} \quad h \in \{2, \ldots, p\} \tag{5.1}$$

$$x \in C^p, \quad \text{iff} \quad \{i \in \mathcal{N} : x_i \in \mathcal{A}_i^p\} \in \mathcal{F}^p \tag{5.2}$$

This formula holds because criteria are either a *gain* or a *cost* criterion (i.e monotone criteria).

Considering the preorder $\geq_i$ that induces the order in $\mathbb{R}$, and as a result of the monotonicity, $\forall h \in \{1, ..., p\}$, we have :

- when i is a gain criterion :
  - $x_i \in \mathcal{A}_i^h$ and $x_i' \geq_i x_i \Rightarrow x_i' \in \mathcal{A}_i^h$,
  - $x_i \notin \mathcal{A}_i^h$ and $x_i \geq_i x_i' \Rightarrow x_i' \notin \mathcal{A}_i^h$.

  Therefore $\mathcal{A}_i^h$ is specified by $b_i^h \in X_i$: $\mathcal{A}_i^h = \{x_i \in X_i : x_i \geq b_i^h\}$.
- when i is a cost criterion :
  - $x_i \in \mathcal{A}_i^h$ and $x_i \geq_i x_i' \Rightarrow x_i' \in \mathcal{A}_i^h$,
  - $x_i \notin \mathcal{A}_i^h$ and $x_i' \geq_i x_i \Rightarrow x_i' \notin \mathcal{A}_i^h$.

Therefore $\mathcal{A}_i^h$ is specified by $b_i \in X_i$: $\mathcal{A}_i^h = \{x_i \in X_i : x_i \leq b_i^h\}$. We study hereafter the MR-Sort rule in the case of single-peaked preferences [12].

5.2.2 Single-peaked and single-valley preferences

In this part, we define single-peaked preferences and describe its formulations using the set of approved values for MR-Sort models.

Let us note more specifically X_i as the finite set of possible values of i that is covered by $[min_i, max_i] \subset \mathbb{R}$, with min_i (resp. max_i) the minimum (resp. maximum) of X_i. Before defining single-peaked and single-valley preferences, we consider the following assumptions :

- $X_i \subset \mathbb{R}$;
- the preorder relation $\succcurlyeq_i$ (the preorder $\geq_i$) that induces the order on $\mathbb{R}$.

Definition 5.1 *A preference is single-peaked with respect to $\geq_i$ iff there exist $p_i \in X_i$ such that:*

- $x_i \leq y_i \leq p_i \Rightarrow p_i \succcurlyeq_i y_i \succcurlyeq_i x_i$, *and*

- $p_i \leq x_i \leq y_i \Rightarrow p_i \succcurlyeq_i x_i \succcurlyeq_i y_i$.

Definition 5.2 *A preference is single-valley with respect to $\geq$ iff there exist $p_i \in X_i$ such that:*

- $x_i \leq y_i \leq p_i \Rightarrow p_i \succcurlyeq_i x_i \succcurlyeq_i y_i$, *and*
- $p_i \leq x_i \leq y_i \Rightarrow p_i \succcurlyeq_i y_i \succcurlyeq_i x_i$.

Considering these definitions, on the left of the peak p_i we can interpret single-peaked preferences as a gain criterion to be maximized, or a single-valley preferences as a cost criterion to be minimized. Similarly, on the right of the peak p_i, we can interpret single-peaked preferences as a cost criterion to be minimized, or single-valley preferences as a gain criterion to be maximized.

Interestingly, single-peaked and single-valley preferences are equivalent to gain and cost criteria in some situations. In fact, when $p_i = max_i$, single-peaked preferences match with gain criteria whereas single-valley preferences match with cost criteria. Conversely, when $p_i = min_i$, single-peaked preferences match with cost criteria whereas single-valley preferences match with gain criteria.

In order to use MR-Sort with single-peaked preferences, we need to consider how the approved values sets are constructed.

Case with 2 categories

Regarding the case of 2 categories, we only consider the set of approved values $\mathcal{A}_i^1$ in order to sort alternatives into categories. In this case, if i is a single-peaked criterion with the peak p_i and $x_i, x_i' \in X_i$, we deduce the following assertions :

$$\left\{\begin{array}{l} x_i \in \mathcal{A}_i^1 \text{ and } p_i \geq_i x_i' \geq_i x_i \Rightarrow x_i' \in \mathcal{A}_i^1 \\ x_i \in \mathcal{A}_i^1 \text{ and } x_i \geq_i x_i' \geq_i p_i \Rightarrow x_i' \in \mathcal{A}_i^1 \\ x_i \notin \mathcal{A}_i^1 \text{ and } p_i \geq_i x_i \geq_i x_i' \Rightarrow x_i' \notin \mathcal{A}_i^1 \\ x_i \notin \mathcal{A}_i^1 \text{ and } x_i' \geq_i x_i \geq_i p_i \Rightarrow x_i' \notin \mathcal{A}_i^1 \end{array}\right. \tag{5.3}$$

We note that the approved values (i.e the elements of $\mathcal{A}_i^1$) are spread around the peak p_i. Thus, we deduce that $\mathcal{A}_i^1$ can be expressed with two thresholds $\overline{b}_i^1, \underline{b}_i^1 \in X_i$ such that $\overline{b}_i^1 \geq_i p_i \geq_i \underline{b}_i^1$. Therefore, $\mathcal{A}_i^1$ is covered by an interval (of minimum range) of approved values. This interval is $[\underline{b}_i^1, \overline{b}_i^1]$ since $p_i \in \mathcal{A}_i^1$.

Similarly, considering i a single-valley criterion, the peak p_i (here, p_i is downwards) and $x, x_i' \in X_i$, we have :

$$\left\{\begin{array}{l} x_i \in \mathcal{A}_i^1 \text{ and } p_i \geq_i x_i \geq_i x_i' \Rightarrow x_i' \in \mathcal{A}_i^1 \\ x_i \in \mathcal{A}_i^1 \text{ and } x_i' \geq_i x_i \geq_i p_i \Rightarrow x_i' \in \mathcal{A}_i^1 \\ x_i \notin \mathcal{A}_i^1 \text{ and } p_i \geq_i x_i' \geq_i x_i \Rightarrow x_i' \notin \mathcal{A}_i^1 \\ x_i \notin \mathcal{A}_i^1 \text{ and } x_i \geq_i x_i' \geq_i p_i \Rightarrow x_i' \notin \mathcal{A}_i^1 \end{array}\right. \tag{5.4}$$

In this case, the same reasoning applies here. The approved set $\mathcal{A}_i^1$ is determined by two thresholds $\overline{b}_i^1, \underline{b}_i^1 \in X_i$ and $\overline{b}_i^1 \geq_i p_i \geq_i \underline{b}_i^1$. Therefore, $\mathcal{A}_i^1$ is covered by $[min_i; \underline{b}_i^1] \cup [\overline{b}_i^1; max_i]$ since $p_i \notin \mathcal{A}_i^1$.

Case with more than 2 categories

Let $i \in \mathcal{N}$ be a single-peaked or a single-valley criterion. With several categories ($p > 2$), the approved sets are represented by couples of intervals that are interleaved between each other throughout the categories.

Then in the case where i is a single-peaked criterion with the peak p_i, we can note that :

$$\begin{cases} x_i \in \mathcal{A}_i^h \text{ and } p_i \geq_i x_i' \geq_i x_i \Rightarrow x_i' \in \mathcal{A}_i^h \\ x_i \in \mathcal{A}_i^h \text{ and } x_i \geq_i x_i' \geq_i p_i \Rightarrow x_i' \in \mathcal{A}_i^h \\ x_i \notin \mathcal{A}_i^{\geq h} \text{ and } p_i \geq_i x_i \geq_i x_i' \Rightarrow x_i' \notin \mathcal{A}_i^h \\ x_i \notin \mathcal{A}_i^{\geq h} \text{ and } x_i' \geq_i x_i \geq_i p_i \Rightarrow x_i' \notin \mathcal{A}_i^h \end{cases} \tag{5.5}$$

We deduce that the approved sets are the following (for i a single-peaked criterion):

- $\mathcal{A}_i^p$ is covered by the interval $[\underline{b}_i^{p-1}, \overline{b}_i^{p-1}]$.
- $\forall h \in \{2, ..., p-1\}$, $\mathcal{A}_i^h$ is covered by intervals $[\underline{b}_i^{h-1}, \underline{b}_i^h]$ and $[\overline{b}_i^h, \overline{b}_i^{h-1}]$
- $\mathcal{A}_i^1$ is covered by $[min_i, \underline{b}_i^1] \cup [\overline{b}_i^1, max_i]$.

Conversely, in the case where i is a single-valley criterion with the hollow or the peak (by abuse of language) p_i , we can note that :

$$\begin{cases} x_i \in \mathcal{A}_i^h \text{ and } p_i \geq_i x_i \geq_i x_i' \Rightarrow x_i' \in \mathcal{A}_i^h \\ x_i \in \mathcal{A}_i^h \text{ and } x_i' \geq_i x_i \geq_i p_i \Rightarrow x_i' \in \mathcal{A}_i^h \\ x_i \notin \mathcal{A}_i^{\geq h} \text{ and } p_i \geq_i x_i' \geq_i x_i \Rightarrow x_i' \notin \mathcal{A}_i^h \\ x_i \notin \mathcal{A}_i^{\geq h} \text{ and } x_i \geq_i x_i' \geq_i p_i \Rightarrow x_i' \notin \mathcal{A}_i^h \end{cases} \tag{5.6}$$

We deduce that the approved sets are the following (for i a single-valley criterion):

- $\mathcal{A}_i^p$ is covered by intervals $[min_i, \underline{b}_i^{p-1}]$ and $[\overline{b}_i^{p-1}, max_i]$.
- $\forall h \in \{2, ..., p-1\}$, $\mathcal{A}_i^h$ is covered by intervals $[\underline{b}_i^h, \underline{b}_i^{h-1}]$ and $[\overline{b}_i^{h-1}, \overline{b}_i^h]$
- $\mathcal{A}_i^1$ is covered by $[\underline{b}_i^1, \overline{b}_i^1]$.

Having weights $w_1, ..., w_n$ attached to criteria (with $w_i \geq 0,\ \forall i$, and $\sum_{i \in \mathcal{N}} w_i = 1$), and a majority threshold $\lambda \in [0.5; 1]$, we can express more generally the MR-Sort rule as follows :

$$\forall h \in \{1, ..., p-1\}, c(a) = h \quad \Leftrightarrow \quad \sum_{i \in \mathcal{N}: a_i \in \mathcal{A}_i^h} w_i \geq \lambda \quad \text{and} \quad \sum_{i \in \mathcal{N}: a_i \in \mathcal{A}_i^{h+1}} w_i < \lambda \tag{5.7}$$

$$c(a) = p, \quad \Leftrightarrow \quad \sum_{i \in \mathcal{N}: a_i \in \mathcal{A}_i^{p-1}} w_i \geq \lambda \tag{5.8}$$

In the next section, we explore the bridges between single-peaked preferences and monotone preferences which enables us to understand more the structure of these preferences as we are interested in learning single-peaked preferences.

In the remaining of this chapter, for the sake of readability, we replace $\leq_i$ (resp. $\geq_i$) with $\leq$ (resp. $\geq$) regarding the comparison between evaluations criteria.

5.3 Relations between single-peaked preferences and monotone preferences

In the previous section, we reformulated an MR-Sort rule that is agnostic of the type of criteria i.e the rule encompassed all types of criteria (gain, cost, single-peaked and single-valley criteria). Despite the differences between single-peaked/single valley preferences and monotone criteria, we can convert a single-peaked/single-valley criterion into a monotone criterion at the cost of some re-encodings.

In this section, we show a procedure to transform a single-peaked/single-valley criterion into a monotone criterion. First, we detail the steps and illustrate this transformation with a simple example considering 2 categories. Then we give some insights on how we can apply the transformation with more than 2 categories.

5.3.1 Transformation of a single-peaked criterion to a monotone criterion with 2 categories

Our procedure consists in converting a single-peaked (resp. single-valley) criterion i into a cost (resp. gain) criterion i'.

Let i a single-peaked criterion (w.l.o.g) and $[\underline{b}_i, \overline{b}_i]$ its profile interval ; since there are only 2 categories, it is the unique profile of criterion i. We denote the middle of the interval $[\underline{b}_i, \overline{b}_i]$ by $b_i^{\perp} = \frac{\underline{b}_i + \overline{b}_i}{2}$.

We set a re-encoding function that calculates for a given criterion evaluation, the absolute value of its distance with the middle of the interval. Therefore this function (ϕ_i) is defined as follows :

$$\begin{array}{rccl} \phi_i : & X_i & \longrightarrow & X_i \\ & x_i & \longrightarrow & |x_i - b_i^{\perp}| \end{array} \tag{5.9}$$

Hence, the new evaluation value $x_{i'} = |x_i - b_i^{\perp}|$. The construction of the cost criterion i' consists of calculating, a new value profile, defining a new criterion evaluation scale, as well as a set of approved values :

1. The profile of the cost criterion i' is the half distance interval $[\underline{b}_i, \overline{b}_i]$ of the single-peaked criterion. Therefore $b_{i'} = \frac{\overline{b}_i - \underline{b}_i}{2}$.

2. The evaluation scale is : $X_{i'} = \{\phi_i(x_i) : x_i \in X_i\}$.

3. Therefore, the approved set of the cost criterion is $\mathcal{A}_{i'} = \{x_i \in X_i : \phi_i(x_i) \leq \frac{\overline{b}_i - \underline{b}_i}{2}\}$. In other words, $\mathcal{A}_{i'} = \{x_{i'} \in X_i : x_{i'} \leq b_{i'}\}$, which corresponds to the definition of the approved set a cost criterion.

Analogously, given a single-valley criterion i and its approved set $\mathcal{A}_i$ (covered by the interval $[min_i; \underline{b}_i] \cup [\overline{b}_i, max_i]$), we can convert i into i', a gain criterion. Using the same definitions of $b_i^{\perp}$ and ϕ_i, we can deduce the following transformations:

1. The profile of the gain criterion i' is the half distance interval $[\underline{b}_i, \overline{b}_i]$ of the single-valley criterion. Therefore $b_{i'} = \frac{\overline{b}_i - \underline{b}_i}{2}$.

2. The evaluation scale is : $X_{i'} = \{\phi_i(x_i) : x_i \in X_i\}$.

3. The approved set of the gain criterion is $\mathcal{A}_{i'} = \{x_i \in X_i : \phi_i(x_i) \geq \frac{\overline{b}_i - \underline{b}_i}{2}\}$. In other words, $\mathcal{A}_{i'} = \{x_{i'} \in X_i : x_{i'} \geq b_{i'}\}$, which corresponds to the definition of the approved set a gain criterion.

In the following example, we perform a transformation of a single-peaked criterion to a cost criterion.

Example 5.1 *Let consider the following MR-Sort model with 2 monotone criteria and one single-peaked criterion, and a set of 5 alternatives. These alternatives are sorted into two categories : C^2 the best category and C^1 the worst one. We intend to transform the single-peaked criterion c_3 into a cost criterion $c_{3'}$.*

c_i	c_1	c_2	c_3	$\mathbf{c(a)}$
a_1	0.2	0.1	0.3	C^1
a_2	0.4	0.6	0.6	C^1
a_3	0.7	0.7	0.4	C^2
a_4	0.8	0.9	0.5	C^2
a_5	0.9	0.2	0.8	C^2
d_i	$\uparrow$	$\downarrow$	$\nearrow\nwarrow$	
b_i	0.5	0.5	[0.4,0.6]	
w_i	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	
$\lambda = \frac{2}{3}$				

Table 5.3: Performance table with the parameters of the MR-Sort model. $\lambda = \frac{2}{3}$

Regarding our transformation procedure, we construct $c_{3'}$ as follows :

- $b_3^{\perp} = \frac{0.6+0.4}{2} = 0.5$
- $\forall x_3 \in X_3$, $x_{3'} = \phi_3(x_3) = |x_3 - 0.5|$
- *The profile of the cost criterion $c_{3'}$ is* $b_{3'} = \frac{0.6-0.4}{2} = 0.1$
- *The evaluation scale is :* $X_{3'} = \{\phi_3(x_3) : x_3 \in X_3\}$
- *Therefore, the approved set of the cost criterion is* $\mathcal{A}_{3'} = \{x_{3'} \in X_{3'} : b_{3'} \leq 0.1\}$.

c_i	c_1	c_2	c'_3	$\mathbf{c(a)}$
a_1	0.2	0.1	0.2	C^1
a_2	0.4	0.6	0.1	C^1
a_3	0.7	0.7	0.1	C^2
a_4	0.8	0.9	0	C^2
a_5	0.9	0.2	0.3	C^2
d_i	$\uparrow$	$\downarrow$	$\downarrow$	
b_i	0.5	0.5	0.1	
w_i	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	
$\lambda = \frac{2}{3}$				

Table 5.4: Performance table with the parameters of the MR-Sort model with c_3 converted in $c_{3'}$. $\lambda = \frac{2}{3}$

We note that the transformation is consistent with regards to the assignments of the 5 alternatives.

We highlight that whenever we evaluate a new alternative with the new model (obtained after the transformation of single-peaked/valley criteria into monotone criteria), the appropriate criteria values of this alternative must also be re-encoded (using functions ϕ_i).

In the next part, we investigate to extend our analysis - of the conversion of single-peaked criteria into monotone criteria - to the case with more than 2 categories.

5.3.2 Transformation of single-peaked preference to monotone preferences with more than 2 categories

In this subsection, we investigate the transformation of single-peaked/single-valley criteria in the presence of more than two categories. In this case, more than one interval profile defines the profile of the considered criterion.

Example 5.2 *Given, this example of MR-Sort model with 3 categories, we attempt to convert the criterion c_3 into a cost criterion $c_{3'}$. We can base our reasoning on the same principle of transformation made in the previous part, in order to re-write c_3.*

c_i	c_1	c_2	c_3	**c(a)**
a_1	0.1	0.1	0.2	C^1
a_2	0.5	0.6	0.6	C^2
a_3	0.9	0.6	0.6	C^2
a_4	0.7	0.8	0.4	C^3
a_5	0.8	0.2	0.7	C^3
d_i	$\uparrow$	$\downarrow$	$\nearrow\nwarrow$	
b_i^1	0.5	0.7	[0.3,0.8]	
b_i^2	0.7	0.5	[0.4,0.5]	
w_i	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	
$\lambda = \frac{2}{3}$				

Table 5.5: Initial performance table with the parameters of the MR-Sort. $\lambda = \frac{2}{3}$

With 3 categories in the MR-Sort model, we have two different single-peaked intervals. Therefore we can investigate 2 cases in order to apply the transformation procedure depending on how is built ϕ_3:

c_i	c_1	c_2	$c_{3'}$	**c(a)**
a_1	0.1	0.1	0.35	C^1
a_2	0.5	0.6	0.05	C^2
a_3	0.9	0.6	0.05	C^3
a_4	0.7	0.8	0.15	C^2
a_5	0.8	0.2	0.15	C^3
d_i	$\uparrow$	$\downarrow$	$\downarrow$	
b_i^1	0.5	0.7	0.25	
b_i^2	0.7	0.5	0.05	
w_i	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	
$\lambda = \frac{2}{3}$				

(a) Case 1

c_i	c_1	c_2	$c_{3'}$	**c(a)**
a_1	0.1	0.1	0.25	C^2
a_2	0.5	0.6	0.15	C^2
a_3	0.9	0.6	0.15	C^2
a_4	0.7	0.8	0.05	C^3
a_5	0.8	0.2	0.25	C^3
d_i	$\uparrow$	$\downarrow$	$\downarrow$	
b_i^1	0.5	0.7	0.25	
b_i^2	0.7	0.5	0.05	
w_i	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$	
$\lambda = \frac{2}{3}$				

(b) Case 2

Table 5.6: Performance table, assignments and parameters of the 2 resulted MR-Sort models : (a) model using the formulation of ϕ_3 with $b^{\perp 1}$, (b) model using the formulation of ϕ_3 with $b^{\perp 2}$. $\lambda = \frac{2}{3}$

Depending on the 2 cases, we compute the profiles of the cost criterion $c_{3'}$:
Case 1: *$\phi_3(x_3) = |x_3 - b_3^{\perp 1}|$ (ϕ_3 is defined with $b_3^{\perp 1}$)*

- $b_3^{\perp 1} = \frac{0.8+0.3}{2} = 0.55$
- $\forall x_3 \in X_3$, $\phi_3(x_3) = |x_3 - 0.55|$
- $b_{3'}^1 = \frac{0.8-0.3}{2} = 0.25$
- $b_{3'}^2 = \frac{0.6-0.4}{2} = 0.05$

Case 2: *$\phi_3(x_3) = |x_3 - b_3^{\perp 2}|$ (ϕ_3 is defined with $b_3^{\perp 2}$)*

- $b_3^{\perp 2} = \frac{0.5+0.4}{2} = 0.45$
- $\forall x_3 \in X_3$, $\phi_3(x_3) = |x_3 - 0.45|$
- $b_{3'}^1 = \frac{0.8-0.3}{2} = 0.25$
- $b_{3'}^2 = \frac{0.5-0.4}{2} = 0.05$

In both scenarios, the resulting MR-Sort models are not equivalent to the initial MR-Sort model since the assignments are not consistent with the initial assignments. Therefore our transformation procedure cannot be applied when considering at least 3 categories in the model.

In fact, with more than two categories, this transformation may not work since the middle of intervals are not necessarily the same (as in our previous example). It results in two distinct middles of intervals that leads to two potential transformed models that are not equivalent to the initial one.

In conclusion, as a rule of thumb, for more than 2 categories, we must first of all ensure that profiles intervals are centred around the same middle value. If it is the case, we can apply our transformation procedure detailed previously. If not, we must make more changes. One course of action is to set the middle of the most embedded profile interval, (e.g. $b_i^{\perp 1}$, the middle of $[\underline{b}_i^1, \overline{b}_i^1]$ if i is a single-valley criterion) as the middle on which all others profiles intervals must be centred around. This implies to elaborate a transformation application in order to transform the other intervals such that $b_i^{\perp h}$ becomes the middle of those intervals.

5.4 Conclusion

In this chapter, we introduced the single-peaked preferences for MR-Sort models. The construction of these preferences makes it possible to be expressed in the MR-Sort settings. We also investigate the transformation of single-peaked (resp. single-valley) criteria into cost (resp. gain) criteria (in the case of two categories). In the next two chapters, we leverage these results in order to learn MR-Sort models taking into account single-peaked (single-valley) criteria.

Chapter 6

An exact approach for the resolution of the Inverse MR-Sort problem with single-peaked preferences

Contents

6.1 Introduction and reminder

In Chapter 5, we defined single-peaked criteria as criteria which preference order is twofold : the preferences increase before the most preferred value (the peak) and decrease after. We also characterized single-peaked preferences and elaborated how it can be handled as criteria in MR-Sort models.

In this chapter, we present an approach based on a Mixed Integer Programming (MIP) for the resolution of the Inverse MR-Sort problem with single-peaked preferences. First, in this section, we recall some notions of our context. Second, we detail our method which uses single-peaked expressions established in Chapter 5 in order to embrace as well as possible the decision maker (DM) preferences. In Section 6.3 and Section 6.4, we run some experiments. Our algorithm was challenged both with customized datasets and real-case data. We discuss about advantages and disadvantages of the approach.

6.1.1 Single-peaked preferences and the Inverse MR-Sort-SP problem

We aim at learning the parameters of an MR-Sort model with potentially single-peaked criteria from assignment examples. Let X, the cardinal product of evaluation scales $X_i, i \in \mathcal{N}$, that are finite sets. We denote an assignment example by a couple $(a, c(a))$, with the alternative $a \in A^\star \subset X$ and its category $c(a) \in \{C^1, \ldots, C^p\}$, the set of categories. Our learning process consists of the resolution of a MIP program based on L, the set of assignment examples (the learning set). Therefore we call the new Inverse MR-Sort problem, *Inv-MR-Sort-SP* problem since we take into account single-peaked/single-valley criteria. This problem takes a learning set L as input, and calculates the optimal tuple of MR-Sort parameters (in terms of the maximization of the classification accuracy). The parameters are : the set of preference directions, the set of criteria weights, the majority threshold, and the profile vector.

In this problem, we assume not knowing in advance the type of preferences of criteria involved in the learning process. In addition as said previously, we take into consideration single-peaked and single-valley criteria. We denote by $\mathcal{S}$ the set of single-peaked and single-valley criteria, and $s, s = |\mathcal{S}| \leq n$ the number of single peaked and single-valley criteria. We also denote by $\mathcal{Q}$ the set of criteria with unknown preference directions, and $q, q = |\mathcal{Q}| \leq n$ the cardinal of this set. We note $IMSS_{q|n}$ the *Inv-MR-Sort-SP* problem with q, the number of criteria with unknown preferences directions, and n the number of criteria which possibly contains some single-peaked/single-valley criteria.

6.2 The MIP formulation

In the present section, we formulate the *Inv-MR-Sort-SP* problem as a MIP. The aim of this section is to describe and explain this formulation. First we detail some general setting of our approach by considering two categories. Before presenting the complete mathematical program, we explain the different variables and constraints pertaining to the *Inv-MR-Sort-SP* problem. Then we formulate the problem and interpret the results given by the MIP. Finally, we give some insights on the *Inv-MR-Sort-SP* problem with more than two categories.

We recall in the following the three main parameters that we expect to learn with this approach:

- the preference direction of each criterion d_i (cost ($\downarrow$), gain ($\uparrow$), single-peaked ($\nearrow\nwarrow$), or single-valley ($\searrow\swarrow$) criterion), $i \in \mathcal{N}$
- the weights of criteria w_i, $i \in \mathcal{N}$ and the majority threshold λ,
- the limit profiles values b_i^h for cost or gain criteria, and the interval $[\underline{b}_i^h, \overline{b}_i^h]$ for single-peaked or single-valley criteria, $i \in \mathcal{N}$, $h \in \{1, \ldots, p\}$.

The input of our algorithm is the learning set L which represents a set of assignment examples. We consider two categories : C^2 and C^1, with $C^2 \rhd C^1$ ($\rhd$ describes the order between categories, here C^2 is better than C^1). Let us note by $A^\star$ the set of reference alternatives and a partition of this set into 2 subsets $A^\star = A^{\star 1} \cup A^{\star 2}$. Let j be the index of alternatives, with $A^\star = \{a^1, ..., a^j, ..., a^{|A^\star|}\}$. Therefore, we have : $A^{\star 1} = \{a^j \in A^\star : c(a^j) = C^1\}$ and $A^{\star 2} = \{a^j \in A^\star : c(a^j) = C^2\}$. We call $J^\star$, $J^{\star 1}$, and $J^{\star 2}$ the indices j of alternatives contained in $A^\star$, $A^{\star 1}$, and $A^{\star 2}$, respectively. We recall X_i the set of possible evaluations of i that is included in $[min_i; max_i]$, with min_i (resp. max_i) the minimum (resp. maximum) of X_i.

In this MIP formulation, we propose an implementation that is restrained to the expression of only two types of preferences : single-peaked or single-valley criteria. We opt for this configuration to avoid redundancy. As we showed in Chapter 5, we can express a single-peaked criteria as a gain criterion if the peak is equal to max_i or as a cost criterion if the peak is equal to min_i. Similarly, we can derive both a gain and a cost criterion from a single-valley criterion : if the peak is "strictly lower" than min_i, then we have a gain criterion ; if the peak is "strictly higher" than max_i, we have a cost criterion. In these cases, we can consider the peak as a fictitious one since X_i is bounded by $[min_i; max_i]$.

Therefore, our rationale is the following :

1. Given the *Inv-MR-Sort-SP* problem, we assume all criteria as either single-peaked or single-valley,

2. We formulate the MIP by fixing all the criteria as either single-peaked or single-valley, and solve it.

3. Finally, once the resolution of the MIP, we attempt to retrieve the true preference directions of criteria.

As previously mentioned, for simplicity reasons, we illustrate our mathematical formulation with two categories ($p = 2$); thus $\mathcal{A}_i$ is $\mathcal{A}_i^2$ and we omit the category index on the profile notation. At the end of the section, we describe the supplementary requirements in the case of more than two categories.

In the following, first we express decision variables and constraints, then we expose the complete formulation and lastly we derive the learned parameters.

6.2.1 Variables and constraints related to approved sets and profiles

We consider i a single-peaked criterion. Its corresponding set of approved values noted $\mathcal{A}_i$ is covered by the interval $[\underline{b}_i, \overline{b}_i]$ (see Chapter 5.2). We recall X_i the evaluation scale of i, with $i \in \mathcal{N} = \{1, \ldots, n\}$. Let denote by $b_i^{\perp} = \frac{\overline{b}_i + \underline{b}_i}{2}$, the middle of the interval of approved values which is a variable of the MIP. We note $a^j \in A^{\star}$, an alternative in the reference set and its evaluation on criterion i, a_i^j. Therefore, a_i^j is part of the approved set (i.e, $a_i^j \in \mathcal{A}_i$) if $a_i^j \in [\underline{b}_i, \overline{b}_i]$.

Let us recall some facts and expressions deduced in Chapter 5.

We have noted that : $\phi_i(a_i^j) = |a_i^j - b_i^{\perp}| \leq \frac{\overline{b}_i - \underline{b}_i}{2}$ if $a_i^j \in [\underline{b}_i, \overline{b}_i]$. It can be read as the following : the value a_i^j belongs to $[\underline{b}_i, \overline{b}_i]$ if the distance of the value a_i^j from the middle of the interval $[\underline{b}_i, \overline{b}_i]$ does not exceed half the range of the interval.

This condition enables us to write the definition of the approved set $\mathcal{A}_i$ as $\{x_i \in X_i : |x_i - b_i^{\perp}| \leq \frac{\overline{b}_i - \underline{b}_i}{2}\}$, which captures the variables we desire to retrieve, i.e $\overline{b}_i$, $\underline{b}_i$ and $b_i^{\perp}$.

In the following, we introduce the decision variables and the constraints pertaining to approved sets and profiles.

Decision variables

We describe the following decision variables of the MIP :

- $\alpha_i^j, \;\; i \in \mathcal{N}, \; j \in J^{\star}$: in order to test whether the alternative evaluation a_i^j belongs to $\mathcal{A}_i$, we introduce $\alpha_i^j = a_i^j - b_i^{\perp}$ such that $a_i^j \in \mathcal{A}_i \Leftrightarrow |\alpha_i^j| \leq \frac{\overline{b}_i - \underline{b}_i}{2}$. By introducing this new notation, we re-encode criterion i as a cost criterion i'. In particular, $|\alpha_i^j|$ is the evaluation of this cost criterion, and reads as the

distance between a_i^j and $b_i^\perp$; the limit profile of this criterion is $b_{i'} = \frac{\overline{b}_i - \underline{b}_i}{2}$, which is half the interval $[\underline{b}_i, \overline{b}_i]$.

- $b_i^\perp, i \in \mathcal{N}$: which is the middle of the interval $[\underline{b}_i, \overline{b}_i]$,
- $b_i, i \in \mathcal{N}$: the limit profile of the transformed criterion i (which is a cost criterion). Thus, we can write $\mathcal{A}_i$ as $\mathcal{A}_i = \{x_i \in X_i : |x_i - b_i^\perp| \leq b_i\}$.
- α_i^{j+} and α_i^{j-}, $i \in \mathcal{N}$, $j \in J^\star$: the two positive variables such that $\alpha_i^j = \alpha_i^{j+} - \alpha_i^{j-}$, which enable to linearize the use of absolute value of α_i^j in the MIP.
- β_i^j, $i \in \mathcal{N}$, $j \in J^\star$: a binary variable that helps to properly restrain the domains of α_i^{j+} and α_i^{j-}.
- $\delta_{ij} \in \{0, 1\}$, $i \in \mathcal{N}$, $j \in J^\star$: a binary variable expressing the membership of evaluation a_i^j in the approved set $\mathcal{A}_i$ ($\delta_{ij} = 1 \Leftrightarrow a_i^j \in \mathcal{A}_i$).
- σ_i, $i \in \mathcal{N}$: a binary variable which indicates whether the criterion i is a single-peaked ($\sigma_i = 1$) or single-valley criterion ($\sigma_i = 0$).

Next, we describe and explain the MIP constraints based on these decision variables.

Constraints

In the MIP formulation, we need to characterize the constraints into linear expressions. In order to achieve that for the expression $|\alpha_i^j| = |a_i^j - b_i^\perp|$, we need the variables α_i^{j+} , α_i^{j-} and binary variables β_i^j verifying the following constraints (6.1a)-(6.1c) :

$$\alpha_i^j = a_i^j - b_i^\perp = \alpha_i^{j+} - \alpha_i^{j-} \tag{6.1a}$$

$$0 \leq \alpha_i^{j+} \leq \beta_i^j M \tag{6.1b}$$

$$0 \leq \alpha_i^{j-} \leq (1 - \beta_i^j) M \tag{6.1c}$$

M is an arbitrary large positive value. The constraints (6.1b) and (6.1c) ensure that at least one variable among α_i^{j+} and α_i^{j-} is null. Thus, $|\alpha_i^j| = \alpha_i^{j+} + \alpha_i^{j-}$ remains positive as an absolute value.

In the following we describe constraints pertaining to the nature of criteria. For that purpose, we use the variable δ_{ij} that plays the same role as the one defined in [63]. It enables to determine the relative position of the alternative evaluation a_i^j compared to b_i. This is a useful indication for tuning the appropriate value of b_i in order to remain consistent with the assignment of the alternative a^j.

We distinguish two cases depending on the nature of the preference direction of i :

- **If i is a single-peaked criterion :**

  In this case, we convert the single-peaked criterion into a cost criterion (with $|\alpha_i^j|$, the new evaluation of alternative a^j on i and b_i the new profile value). Therefore the following constraints hold, $\forall i \in \mathcal{N}, j \in J^\star$:

$$\delta_{ij} = 1 \Longleftrightarrow |\alpha_i^j| \leq b_i \Longrightarrow M(\delta_{ij} - 1) \leq b_i - (\alpha_i^{j+} + \alpha_i^{j-}) \tag{6.2a}$$
$$\delta_{ij} = 0 \Longleftrightarrow |\alpha_i^j| > b_i \Longrightarrow b_i - (\alpha_i^{j+} + \alpha_i^{j-}) < M\,\delta_{ij} \tag{6.2b}$$
$$\delta_{ij} \in \{0, 1\} \tag{6.2c}$$

- **If i is a single-valley criterion :**

  Conversely in the case where i is a single-valley criterion, it is transformed into a gain criterion (considering $|\alpha_i^j|$, the new evaluation of alternative a^j on i and b_i the new profile value). Therefore we have, $\forall i \in \mathcal{N}, j \in J^\star$:

$$\delta_{ij} = 1 \Longleftrightarrow |\alpha_i^j| \geq b_i \Longrightarrow M(\delta_{ij} - 1) \leq (\alpha_i^{j+} + \alpha_i^{j-}) - b_i \tag{6.3a}$$
$$\delta_{ij} = 0 \Longleftrightarrow |\alpha_i^j| < b_i \Longrightarrow (\alpha_i^{j+} + \alpha_i^{j-}) - b_i < M\,\delta_{ij} \tag{6.3b}$$
$$\delta_{ij} \in \{0, 1\} \tag{6.2c}$$

As we do not know a priori the preference direction of i, we have to take into account both cases in the MIP. In order to do that, we introduce a binary variable $\sigma_i,\ i \in \mathcal{N}$ which indicates whether criterion i is a single-peaked ($\sigma_i = 1$) or a single-valley criterion ($\sigma_i = 0$). When $\sigma_i = 1$, the constraints (6.4c) and (6.4d) pertaining to the single-peaked criteria are active while the constraints (6.4a) and (6.4b) for single-valley criteria are inactive, and conversely when $\sigma_i = 0$.

$$-M\,\sigma_i + M(\delta_{ij} - 1) \leq \alpha_i^{j+} + \alpha_i^{j-} - b_i \tag{6.4a}$$
$$\alpha_i^{j+} + \alpha_i^{j-} - b_i < M\,\delta_{ij} + M\,\sigma_i \tag{6.4b}$$
$$M(\sigma_i - 1) + M(\delta_{ij} - 1) \leq b_i - \alpha_i^{j+} - \alpha_i^{j-} \tag{6.4c}$$
$$b_i - \alpha_i^{j+} - \alpha_i^{j-} < M\,\delta_{ij} + M\,(1 - \sigma_i) \tag{6.4d}$$
$$\delta_{ij} \in \{0, 1\} \tag{6.2c}$$
$$\sigma_i \in \{0, 1\} \tag{6.4e}$$

Finally, we enforce the bounds of the single-peaked/single-valley interval to be comprised within $[min_i - \epsilon, max_i + \epsilon]$ (where ϵ is a small positive value), by adding

the 2 following constraints :

$$b_i^{\perp} - b_i \geq min_i - \epsilon \tag{6.5a}$$
$$b_i^{\perp} + b_i \leq max_i + \epsilon \tag{6.5b}$$

By restraining the search of profile within a slightly larger interval than the evaluation scale $[min_i, max_i]$, we enable the MIP to fully embrace all types of preference directions. In particular, it enables to encapsulate cost or gain criteria expressed when $\sigma_i = 0$. The reader can refer to Section 6.2.5 on the interpretation of the resulting preference directions in order to grasp the rationale.

6.2.2 Variables and constraints related to weights

The variables pertaining to weights are $w_i, i \in \mathcal{N}$ and their corresponding constraint is $w_i \geq 0, i \in \mathcal{N}$ where w_i represents the weight of criterion i. We normalize weights with the following constraint : $\sum_{i \in \mathcal{N}} w_i = 1$.

In order to link weights with other decisions variables, we introduce in the MIP formulation, another variable (also described in [63]) : $c_{ij},\ i \in \mathcal{N},\ j \in J^{\star}$.

We define these continuous variables $c_{ij},\ i \in \mathcal{N},\ j \in J^{\star}$ such that $\delta_{ij} = 0 \Leftrightarrow c_{ij} = 0$ and $\delta_{ij} = 1 \Leftrightarrow c_{ij} = w_i$.

The variable c_{ij} value depends on the consideration of w_i in the weighted sum that contributes to the correct assignment of a^j. For instance, if $\delta_{ij} = 0$, then $c_{ij} = 0$ and the constraint 6.6b is inactive. We also have $\delta_{ij} = 1 \Leftrightarrow c_{ij} = w_i$. The criterion i does not contribute to the assignment of a^j whenever $c_{ij} = 0$. To ensure the correct definition of c_{ij}, we enforce the following constraints, $\forall i \in \mathcal{N},\ j \in J^{\star}$:

$$c_{ij} \leq \delta_{ij} \tag{6.6a}$$
$$\delta_{ij} - 1 + w_i \leq c_{ij} \tag{6.6b}$$
$$c_{ij} \leq w_i \tag{6.6c}$$
$$0 \leq c_{ij} \tag{6.6d}$$

6.2.3 Variables and constraints related to the assignment examples

Here we introduce some constraints concerning the restoration of assignment examples with the MR-Sort rule. We define binary variables $\gamma_j \in \{0, 1\},\ j \in J^{\star}$ to denote the number of correctly assigned alternatives. If $\gamma_j = 1$, then the alternative a^j is correctly assigned, otherwise $\gamma_j = 0$. The following constraints ensure the

correct definition of γ_j and $\lambda \in [0.5, 1]$ represents the MR-Sort majority threshold.

$$\sum_{i \in \mathcal{N}} c_{ij} \geq \lambda + M(\gamma_j - 1), \forall j \in J^{\star 2} \tag{6.7a}$$

$$\sum_{i \in \mathcal{N}} c_{ij} < \lambda - M(\gamma_j - 1), \forall j \in J^{\star 1} \tag{6.7b}$$

6.2.4 Objective function and the complete MIP formulation

The objective for the *Inv-MR-Sort-SP* problem is to learn the MR-Sort model that best matches the learning set, and consequently restore the unknown preferences directions, which possibly involve single-peaked/single-valley criteria. Therefore, in order to maximize the number of correctly restored assignment examples, we formulate the objective function as the following : $Max \sum_{j \in J^{\star}} \gamma_j$.

Finally, the MIP formulation for the *Inv-MR-Sort-SP* problem with single-peaked and single-valley criteria is given below. We fix M to an arbitrary large positive value, and ε an arbitrary small positive value. The table 6.1 synthesizes the variables involved in this mathematical program.

Refinement of the objective function

In practice, the expression of the objective function (6.8a) is insufficient to fully restore both MR-Sort usual parameters and the unknown preference directions. More precisely, we frequently fail to diagnose gain and cost criteria since the results indicate single-peaked/single-valley criteria instead. In fact, with this objective function 6.8a, the MIP ignores how to discriminate between single-peaked/single-valley criteria and monotone criteria.

In order to solve this issue, we add another term to the objective function : $\sum_{i \in \mathcal{N}} b_i$ (to be maximized). The rationale behind the maximization of this term is to foster b_i (which is half of the width of $[\underline{b}_i, \overline{b}_i]$) to extend $[\underline{b}_i, \overline{b}_i]$ as much as possible. This would enable to easily deduce a gain or a cost criterion if one of the bounds reaches min_i or max_i.

This term should not be optimized at the cost of the number of correctly restored assignments; for this reason, we set to $\frac{1}{\sum_{i \in \mathcal{N}} max_i}$ the coefficient of this new term of the objective function.

Therefore the new version of the objective of the MIP is the following :

$$\max \sum_{j \in J^{\star}} \gamma_j + \frac{1}{\sum_{i \in \mathcal{N}} max_i} \sum_{i \in \mathcal{N}} b_i \tag{6.9}$$

Obviously, the first term maximizes the number of correctly assigned alternatives

$$\max \sum_{j \in J^\star} \gamma_j \tag{6.8a}$$

$$\sum_{i \in \mathcal{N}} c_{ij} \geq \lambda + M(\gamma_j - 1) \qquad \forall j \in J^{\star 2} \tag{6.7a}$$

$$\sum_{i \in \mathcal{N}} c_{ij} + \varepsilon \leq \lambda - M(\gamma_j - 1) \qquad \forall j \in J^{\star 1} \tag{6.7b}$$

$$\sum_{i \in \mathcal{N}} w_i = 1 \tag{6.8b}$$

$$c_{ij} \leq \delta_{ij} \qquad \forall j \in J^\star, \forall i \in \mathcal{N} \tag{6.6a}$$

$$c_{ij} \geq \delta_{ij} - 1 + w_i \qquad \forall j \in J^\star, \forall i \in \mathcal{N} \tag{6.6b}$$

$$c_{ij} \leq w_i \qquad \forall j \in J^\star, \forall i \in \mathcal{N} \tag{6.6c}$$

$$b_i^\perp - a_i^j = \alpha_i^{j+} - \alpha_i^{j-} \qquad \forall j \in J^\star, \forall i \in \mathcal{N} \tag{6.1a}$$

$$\alpha_i^{j+} \leq \beta_i^j M \qquad \forall j \in J^\star, \forall i \in \mathcal{N} \tag{6.1b}$$

$$\alpha_i^{j-} \leq (1 - \beta_i^j) M \qquad \forall j \in J^\star, \forall i \in \mathcal{N} \tag{6.1c}$$

$$-M\sigma_i + M(\delta_{ij} - 1) \leq \alpha_i^{j+} + \alpha_i^{j-} - b_i \qquad \forall j \in J^\star, \forall i \in \mathcal{N} \tag{6.4a}$$

$$\alpha_i^{j+} + \alpha_i^{j-} - b_i + \varepsilon \leq M.\delta_{ij} + M.\sigma_i \qquad \forall j \in J^\star, \forall i \in \mathcal{N} \tag{6.4b}$$

$$M(\sigma_i - 1) + M(\delta_{ij} - 1) \leq b_i - \alpha_i^{j+} - \alpha_i^{j-} \qquad \forall j \in J^\star, \forall i \in \mathcal{N} \tag{6.4c}$$

$$b_i - \alpha_i^{j+} - \alpha_i^{j-} + \varepsilon \leq M.\delta_{ij} + M.(1 - \sigma_i) \qquad \forall j \in J^\star, \forall i \in \mathcal{N} \tag{6.4d}$$

$$b_i^\perp - b_i \geq min_i - \epsilon \qquad \forall i \in \mathcal{N} \tag{6.5a}$$

$$b_i^\perp + b_i \leq max_i + \epsilon \qquad \forall i \in \mathcal{N} \tag{6.5b}$$

$$c_{ij} \in [0,1], \delta_{ij} \in \{0,1\} \qquad \forall j \in J^\star, \forall i \in \mathcal{N} \tag{6.8c}$$

$$\alpha_i^{j+}, \alpha_i^{j-} \in \mathbb{R}^+ \qquad \forall j \in J^\star, \forall i \in \mathcal{N} \tag{6.8d}$$

$$\beta_i^j \in \{0,1\} \qquad \forall j \in J^\star, \forall i \in \mathcal{N} \tag{6.8e}$$

$$b_i \in \mathbb{R}, w_i \in [0,1], b_i^\perp \in \mathbb{R}, \sigma_i \in \{0,1\} \qquad \forall i \in \mathcal{N} \tag{6.8f}$$

$$\gamma_j \in \{0,1\} \qquad \forall j \in J^\star \tag{6.8g}$$

$$\lambda \in [0.5,1] \tag{6.8h}$$

and the second term fosters the restoration of monotone preference direction of criteria.

Variable	Domain	Number of variables	Definition
α_i^{j+}	$\mathbb{R}^+$	$n \times \lvert A^\star \rvert$	First component of the absolute value $\lvert a_i^j - b_i^\perp \rvert$
α_i^{j-}	$\mathbb{R}^+$	$n \times \lvert A^\star \rvert$	Second component of the absolute value of $\lvert a_i^j - b_i^\perp \rvert$
β_i^j	$\{0,1\}$	$n \times \lvert A^\star \rvert$	Binary variable indicating the sign of $a_i^j - b_i^\perp$
σ_i	{0,1}	n	$\sigma_i = 1$ if criterion i is single-peaked, $\sigma_i = 0$ if i is single-valley
γ_j	{0,1}	$\lvert A^\star \rvert$	$\gamma_j = 1$ if alternative a^j is correctly assigned by the model, $\gamma_j = 0$ if not
δ_{ij}	{0,1}	$n \times \lvert A^\star \rvert$	$\delta_{ij} = 1$ if $a_i^j \in \mathcal{A}_i$, $\delta_{ij} = 0$ if $a_i^j \notin \mathcal{A}_i$
c_{ij}	[0,1]	$n \times \lvert A^\star \rvert$	$c_{ij} = w_i$ iff $a_i^j \in \mathcal{A}_i$ (i.e, iff $\delta_{ij} = 1$), $c_{ij} = 0$ if $a_i^j \notin \mathcal{A}_i$ (i.e, if $\delta_{ij} = 0$)
$b_i^\perp$	$\mathbb{R}$	n	Middle of the interval $[\underline{b}_i, \overline{b}_i]$
b_i	$\mathbb{R}$	n	Value of half the width of the interval $[\underline{b}_i, \overline{b}_i]$ on criterion i
w_i	[0,1]	n	Weight of a criterion i
λ	[0,1]	1	Majority threshold

Table 6.1: Description of decision variables

6.2.5 Interpretation of the optimal solution

Once the above mathematical program is solved, the last remaining step is to derive the MR-Sort learned parameters : $(\{d_i, i \in \mathcal{N}; w_i, i \in \mathcal{N}; \lambda; \langle b \rangle = (b^h, h \in \{1, ..., p\}\}$. We directly obtain weights values over criteria $(w_i, i \in \mathcal{N})$ and the associated majority threshold λ from the corresponding variables in the optimal solution.

Here below, we illustrate how to infer the type of the criterion (gain, cost, single-peaked, or single-valley) and the corresponding profile value by relying on the variable σ_i :

Case $\sigma_i = 1$

This case means that the MIP identifies a single-peaked criterion for i. Therefore we distinguish 3 cases :

- if $b_i^\perp - b_i \leq min_{j \in J^\star}\{a_i^j\}$, then criterion i is a cost criterion, and the profile value is $b_i^\perp + b_i$, i.e. $\mathcal{A}_i$ is covered by $]-\infty, b_i^\perp + b_i]$, see Figure 6.1 case 3,

- if $b_i^{\perp} + b_i \geq max_{j \in J^{\star}}\{a_i^j\}$, then criterion i is a gain criterion, and the profile value is $b_i^{\perp} - b_i$, i.e. $\mathcal{A}_i$ is covered by $[b_i^{\perp} - b_i, \infty[$, see Figure 6.1) case 2,
- otherwise, i is a single-peaked criterion, and $\mathcal{A}_i$ is covered by $[b_i^{\perp} - b_i, b_i^{\perp} + b_i]$ that is the interval profile, see Figure 6.1 case 1

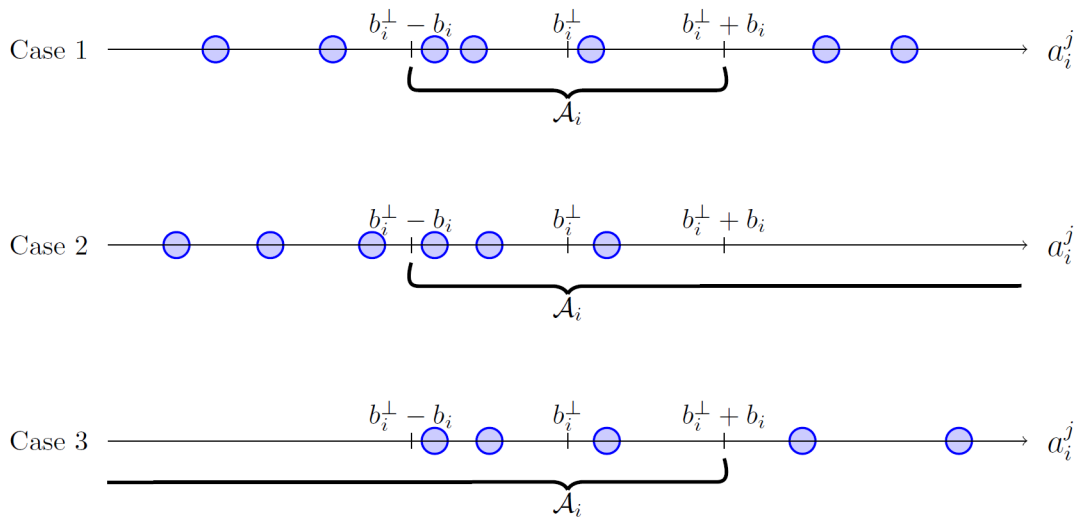


Figure 6.1: Three cases for single-peaked criteria

Case $\sigma_i = 0$

This case means that the MIP identifies a single-valley criterion for i. Therefore we distinguish 3 cases :

- if $b_i^{\perp} - b_i < min_{j \in J\star}\{a_i^j\}$, then criterion i is a gain criterion, and the profile value is $b_i^{\perp} + b_i$, i.e. $\mathcal{A}_i$ is covered by $[b_i^{\perp} + b_i, \infty[$, see Figure 6.2 case 3,
- if $b_i^{\perp} + b_i > max_{j \in J^{\star}}\{a_i^j\}$, then criterion i is a cost criterion, and the profile value is $b_i^{\perp} - b_i$, i.e. $\mathcal{A}_i$ is covered by $[-\infty, b_i^{\perp} - b_i]$, see Figure 6.2 case 2,
- otherwise, i is a single-valley criterion, and $\mathcal{A}_i$ is covered by $[-\infty, b_i^{\perp} - b_i] \cup [b_i^{\perp} + b_i, \infty[$ that is the interval profile, see Figure 6.2 case 1.

6.2.6 General case

The MIP implementation previously given, is designed to retrieve the preference directions of the whole set of criteria. However, in the general case, one can deal

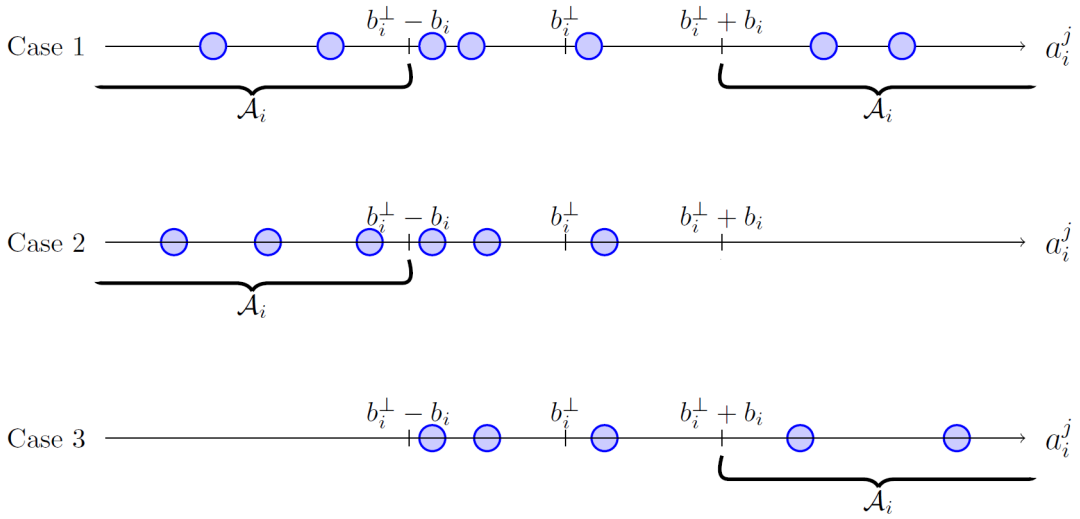


Figure 6.2: Three cases for single-valley criteria

with some criteria whose preference directions are already known. It is possible to translate such knowledge in the MIP formulation. We recall $\mathcal{Q}$, with $\mathcal{Q} \subseteq \mathcal{N}$, the set of criteria with unknown preference directions.

In order to bring additional information on preference directions to the implementation, we apply the following changes:

- Regarding the objective function 6.9, we need to replace $\sum_{i \in \mathcal{N}} b_i$ by $\sum_{i \in \mathcal{Q}} b_i$, since this term - that involves b_i, the half width of $[\underline{b}_i, \overline{b}_i]$ - pertains to criteria in $\mathcal{Q}$.
- Regarding the preference direction of a criterion $i \in \mathcal{N} \setminus \mathcal{Q}$, we distinguish 4 cases:
  - If i is a gain criterion, w.l.o.g. we add the constraint $\sigma_i = 1$. Moreover, the constraint 6.5b becomes $b_i^{\perp} + b_i \geq max_i$.
  - If i is a cost criterion, w.l.o.g. we add the constraint $\sigma_i = 0$. Moreover, considering ϵ a small positive value, the constraint 6.5a becomes $b_i^{\perp} - b_i \leq min_i - \epsilon$.
  - If i is a single-peaked criterion, we add the constraint $\sigma_i = 1$.
  - If i is a single-valley criterion, we add the constraint $\sigma_i = 0$.

6.2.7 Extension to more than two categories

Our implementation can be extended to more than two categories by adding supplementary variables and constraints to the mathematical program.

We introduce the following variables δ_{ij}^h, c_{ij}^h, α_i^{jh+}, α_i^{jh-}, β_i^{jh}, b_i^h, and $b_i^{\perp h}$, $\forall h \in \{1, 2, \ldots, p-1\}$, which extend respectively the variables δ_{ij}, c_{ij}, α_i^{j+}, α_i^{j-}, β_i^j, b_i, and $b_i^{\perp}$ to take into account the multiple categories aspect.

In order to operate the extension, we must keep in mind how profiles that delimits categories $C^h, h = \{1, 2, \ldots, p-1\}$ are related to the approved sets which are embedded : $\mathcal{A}_i^p \subseteq \mathcal{A}_i^{p-1} \subseteq \mathcal{A}_i^{p-2} \subseteq ... \subseteq \mathcal{A}_i^1 = X_i$.

More precisely, if i is a single-peaked criterion, we must ensure the inclusion of the following intervals :

$$\begin{array}{lll} [b_i^{\perp p-1} - b_i^{p-1}, b_i^{\perp p-1} + b_i^{p-1}] & \subseteq & [b_i^{\perp p-2} - b_i^{p-2}, b_i^{\perp p-2} + b_i^{p-2}] \\ & \subseteq & \ldots \\ & \subseteq & [b_i^{\perp 1} - b_i^1, b_i^{\perp 1} + b_i^1] \\ & \subseteq & [minx_i, max_i] \end{array}$$

Conversely, if i is a single-valley criterion, we must ensure the inclusion of the following intervals :

$$\begin{array}{ll} & [min_i, b_i^{\perp p-1} - b_i^{p-1}] \cup [b_i^{\perp p-1} + b_i^{p-1}, max_i] \\ \subseteq & [min_i, b_i^{\perp p-2} - b_i^{p-2}] \cup [b_i^{\perp p-2} + b_i^{p-2}, max_i] \\ \subseteq & \ldots \\ \subseteq & [min_i, b_i^{\perp 1} - b_i^1] \cup [b_i^{\perp 1} + b_i^1, max_i] \\ \subseteq & [minx_i, max_i] \end{array}$$

We reformulate constraints 6.7a and 6.7b by taking into account more than two categories. Regarding the extreme categories C^1 and C^p, we must ensure these two following constraints :

- $\sum_{i \in \mathcal{N}} c_{ij}^1 + \varepsilon \leq \lambda - M(\gamma_j - 1),\ \forall a^j \in C^1$
- $\sum_{i \in \mathcal{N}} c_{ij}^{p-1} \geq \lambda + M(\gamma_j - 1),\ \forall a^j \in C^p$

The first one is the constraint that enables the sorting of alternatives into C^1, whereas the second equation pertains to the sorting of alternatives into C^p.

In the general case (i.e. $\forall a^j \in C^h \subset [C^2, C^{p-1}]$), the two following constraints must hold :

- $\sum_{i \in \mathcal{N}} c_{ij}^{h-1} \geq \lambda + M(\gamma_j - 1),\ \forall a^j \in C^h, h \in [2, p-1]$
- $\sum_{i \in \mathcal{N}} c_{ij}^h + \varepsilon \leq \lambda - M(\gamma_j - 1),\ \forall a^j \in C^h, h \in [2, p-1]$

In this case, the first constraint qualifies winning coalitions (or sufficient coalitions) of criteria in favor of the assignment of alternatives in C^h, while the second constraint prevents these coalitions from being in favor of the assignment of alternatives in C^{h+1}.

To sum up, in this section, we presented a MIP implementation for the resolution of the *Inv-MR-Sort-SP*. In next sections, we apply this method to artificial and real data. We therefore discuss the performance of the method.

6.3 Experiments with artificial data

In this section, we study the performance of our proposed algorithm in terms of three indicators : the computing time, the classification accuracy in generalization, and the ability to restore MR-Sort models with correct preference directions (gain, cost, single-peaked or single-valley). The experiments concern artificially generated datasets.

6.3.1 Experimental design

In this part, we follow a similar experiment designed as in Chapter 4, except that we consider two categories in our experiments. We recall the main steps.

Let consider a generated MR-Sort model $\mathcal{M}^0$ representing the Decision Maker preferences. Alternatives are randomly generated and correspond to n-tuples of values (each tuple corresponding to n criteria evaluations). In order to form the learning set L, used as input to our MIP algorithm, we apply the MR-Sort rule with the model $\mathcal{M}^0$ on these alternatives, and thus derive their assignments.

Alternatives are carefully generated so that we arrange balanced datasets (i.e. equal number of assignments in each category). Therefore we solve the *Inv-MR-Sort-SP* problem using the proposed algorithm and generate a learned model noted $\mathcal{M}'$, an optimal model in terms of the classification accuracy (CA_v).

Generation of instances and model parameters

In our experiments, we consider a learning set of 200 assignment examples.

We draw a vector of performance values of alternatives in an independent and identically distributed manner, such that the performance values are contained in the unit interval discretized by tenths. In a similar manner, we randomly generate profile values (either b_i for monotone criteria or $\underline{b}_i$ and $\overline{b}_i$ with $\underline{b}_i \leq \overline{b}_i$ for single-peaked/single-valley criteria). These values are chosen within the interval [0,1] discretized by tenths (which gives in total 11 possible values).

To draw uniformly distributed weight vectors [23], we uniformly generate $|\mathcal{N}|-1$ random values in $[0,1]$ sorted in ascending order. We then prepend 0 and append 1 to this set of values obtaining a sorted set of $|\mathcal{N}|+1$ values. Finally, we calculate the difference between each successive pair of values resulting in a set of $|\mathcal{N}|$ weights. Therefore, their sum is equal to 1. We randomly draw the majority threshold λ in [0,1].

To assess the ability of the algorithm to restore preference directions, we randomly assign to q criteria over n, a preference direction among the four (gain, cost, single-peaked and single-valley). For instance, for each criterion $i \in \mathcal{Q}$, we have 1 chance over 4 to assign to i a gain criterion, and the same chance for the other preference directions. Thereby, the preference directions of these criteria are assumed to be unknown in the learning set, meanwhile the remaining $n-q$ criteria are considered as gain criteria.

Performance metrics and tests parameters

We consider three main metrics in order to assess the performance of our method. First, we consider *computing time*, which is the time (CPU) necessary to solve the MIP algorithm.

Second, we compute the *restoration rate of assignment examples*. Our MIP algorithm is an exact method, thus, we expect to restore the entire learning set L with $\mathcal{M}'$, the resulted model. Therefore, we assess the restoration performance on test sets which are run through $\mathcal{M}^0$ and $\mathcal{M}'$. Test sets comprise newly and randomly generated alternatives whose evaluations are drawn uniformly in [0.05,0.95]. This allows us to compute the restoration rate (also called classification accuracy in generalization noted CA_g). As described in Chapter 2, CA_g is the ratio between the number of alternatives identically assigned in categories by both $\mathcal{M}^0$ and $\mathcal{M}'$, and the number of alternatives.

Finally, we introduce the *preference direction restoration rate (PDR)*. This metric is the ratio between the number of criteria where preference directions have been correctly restored in $\mathcal{M}'$ and the cardinality of the set of criteria with unknown preference directions (i.e. q).

In order to account for the statistical distribution of all the randomly selected values, we independently select 100 different learning sets (each set is associated to an individual $\mathcal{M}^0$) and apply our experimental design on each. Therefore we obtain 100 results per metric that are aggregated. The series of tests consider the following parameters. The number of criteria n varies in $\{4,5,6,7,8,9\}$, q varies in $\{0,1,2,3,4\}$, and the number of categories is set to 2. Test sets sizes contain 10000 alternatives. We executed experiments on a server endowed with an Intel Xeon [1] Gold 6248 CPU @ 2.50GHz, 80 cores and 384 GB RAM. CPLEX 20.1 [30]

[1]Intel, and Intel Xeon are trademarks or registered trademarks of Intel Corporation or its

was used for the MIP resolution. In terms of CPU, we run our experiments using 10 reserved threads and adopt a timeout of one hour.

6.3.2 Results

In the following, we present the results of the randomly generated tests.

6.3.3 Computing time performance

Table 6.2 presents the median CPU time of the terminated instances (timeout fixed to 1 hour). The execution time grows with the number of criteria and with the number of criteria with unknown preference direction up to $n = 7$ and $q = 2$. Beyond this limit, the execution time fluctuates. This behaviour can be explained by the fact that it is influenced by irregularities on the number of terminated instances which are considered for the median time computation.

In addition, Table 6.2 shows for each setting (n and q fixed), the percentage of instances - over 100 - that terminated within the time limit, set to 1 hour.

Unsurprisingly, the number of terminated instances diminishes with both the number of criteria and the number of criteria with unknown preference directions. In particular, the rate falls from 95% with $n = 4$ to 31% with 9 criteria in the model when $q = 3$.

Restoration rate in generalization

Concerning the classification accuracy (CA_g) of the learned models (involving 4 to 9 criteria in the model and 0 to 4 criteria with unknown preference directions), the performance values are globally comprised between 0.9 and 0.95 with 0.93 on average. We do not notice a major trend over both the number of criteria and the number of criteria with unknown preference directions. However, the results obtained correspond to the performances of only terminated instances. Therefore, we predict that the CA_g rate would possibly degrade when executions above the timeout are considered. Logically, the latter are the most difficult instances to learn.

Preference direction restoration rate

Figure 6.3 shows the evolution of the preference direction restoration rate (PDR).

Globally, the PDR decreases with the increase of number of criteria in the model. In addition, this indicator degrades moderately with q, the number of criteria with

subsidiaries in the United States and other countries.

# unknown	Number of criteria (n)					
direction (q)	4	5	6	7	8	9
0	0.34	0.56	0.84	2.38	2.61	3.37
	(**100**)	(**100**)	(**100**)	(**100**)	(**100**)	(**100**)
1	1.51	3.23	4.53	7.22	19.15	18.68
	(**100**)	(**100**)	(**100**)	(**100**)	(**100**)	(**91**)
2	6.12	12.97	30.06	54.38	43.19	58.03
	(**100**)	(**100**)	(**94**)	(**90**)	(**72**)	(**59**)
3	37.48	76.68	72.46	76.29	59.32	25.91
	(**95**)	(**89**)	(**80**)	(**54**)	(**47**)	(**31**)
4	96.34	129.49	61.01	108.25	22.17	23.63
	(**59**)	(**57**)	(**52**)	(**42**)	(**27**)	(**28**)

Table 6.2: Median CPU Time (sec.) of instances solved in 1h, and number of terminated instances in parentheses, with 4 to 9 criteria (n), and 0 to 4 criteria with unknown preference directions (q)

unknown preference directions with respectively 55% and 35% for $q = 1$ and $q = 4$ considering 9 criteria in the model.

The results illustrated in the Table 6.3 give more insights into the behaviour of the algorithm regarding PDR. We consider instances (with n varying from 4 to 9) involving one criterion with unknown preference direction, $q = 1$ (w.l.o.g. c_1 is assumed to be this criterion).

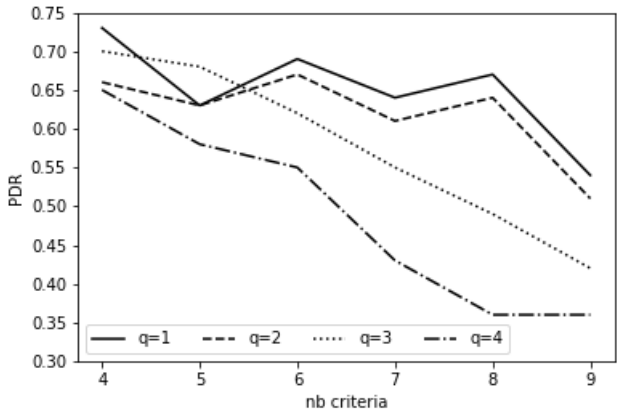


Figure 6.3: Preference direction restoration rate (PDR) considering 1 to 4 criteria with unknown preference direction (q) (average performance over terminated instances)

	$w_1 \leq \frac{1}{2n}$	$\frac{1}{2n} < w_1 < \frac{2}{n}$	$w_1 \geq \frac{2}{n}$
PDR	0.44	0.74	0.78

Table 6.3: PDR averaged over n (n varying from 4 to 9), according to the range of weight of criterion c_1

We analyze the impact of the weight of this criterion (w_1) on the preference direction restoration rate. The PDR rate is averaged over the number of criteria (n) in the model ($n \in \{4, .., 9\}$) and distributed over three intervals:

$[0, \frac{1}{2n}]$,$]\frac{1}{2n}, \frac{2}{n}[$, $[\frac{2}{n}, 1]$. We interpreted these three intervals as three levels of importance of w_1 (respectively low level for $[0, \frac{1}{2n}]$, medium level for $]\frac{1}{2n}, \frac{2}{n}[$, and high level for $[\frac{2}{n}, 1]$). As expected, the average PDR rises with the level of w_1; we have PDR = 44% for low level of importance, whereas 74% and 78% correspond respectively to medium and high levels of importance of w_1. It appears that the

MIP has more difficulty in correctly detecting the preference direction of a criterion when this criterion has low importance.

Discussion

The experiments carried out on randomly generated instances give us an overview of the performance of our method.

First, in terms of computation time, the MIP is affordable with medium-sized models (less than 3 minutes for 200 alternatives in the learning set and up to $n = 9$ and $q = 4$). Furthermore, it is expected to perform better with more CPU threads per run.

Second, regarding to the restoration of new assignment examples, our MIP is quite accurate (with $CA_g = 0.93$ on average, up to 9 criteria). It could be interesting to allow all instances to terminate so we could have a more precise judgment.

Third, our algorithm struggles to restore preference directions with large dimension of the problem (high values of n, q). But it still performs better than the random choice that is 25% (that is one chance over four types of criteria).

Fourth, we observe a correlation between the PDR of criteria and the importance of such criteria in the model. It is very difficult to restore the preference directions of criteria with low importance (with weights less than $\frac{1}{2n}$). Obviously this behaviour opens questions on the overfitting of parameters, which requires further studies.

Finally, it would be interesting to further investigate experimentally whether larger learning sets, and more categories would impact the learning process.

6.4 Tests on a real-world data: the ASA dataset

The ASA[2] dataset [62] constitutes a list of 898 patients evaluated on 14 medical indicators (see Table 6.4). Indicators enable to assign patients into 4 ordered categories (ASA1, ASA2, ASA3, ASA4). These categories are 4 different scores that reveal the patient health. Based on the score of a given patient, anesthesiologists decide whether or not to admit such a patient to surgery. This dataset is relevant for our studies since it encompasses a criterion with single-peaked preference, which is “Blood glucose level" (i.e. glycemia). For scalability reasons, our experiments rely on the ASA dataset with the 8 most relevant criteria. They are presented in bold in Table 6.4.

In order to construct two categories, we divide the dataset into two parts: *Category 2* representing patients in categories ASA1 and ASA2 (67% of the population,

[2]ASA stands for “*American Society of Anesthesiologists*”.

which represents the healthiest) and *Category 1* representing those in categories ASA3 and ASA4 (33% of the population, which are the sickest).

In the following, we construct three scenarios with three different sets of assignment examples. Thus, we intend to learn the preference type of the criterion "Glycemia" which is assumed to be unknown and yet to be "discovered" as a single-peaked criterion.

For each experiment we report the number of distinct performances (values) considered per criterion.

Attribute	Domain (Unit)	Direction
Age	$[0-105]$ (year)	cost
Diabetic	{0,1}	cost
Hypertension	{0,1}	cost
Respiratory failure	{0,1}	cost
Heart failure	{0,1}	cost
Heart rate	$[55-123]$ (bpm)	SP
Heart rate steadiness	{0,1}	gain
Pacemaker	{0,1}	cost
Atrioventricular block	{0,1}	cost
Left ventricular hypertrophy	{0,1}	min .
Oxygen saturation	$[43-100](\%)$	gain
Blood glucose level (glycemia)	$[0.5-3.8]$(g/l)	SP
Systolic blood pressure	$[9-20.5]$(cmHg)	cost
Diastolic blood pressure	$[5-13]$(cmHg)	*cost*

Table 6.4: Original criteria in the ASA dataset

First Dataset

First, the whole original dataset is considered with all 898 assignment examples in the learning set. With this dataset as input to the MIP, we infer the type of criterion of "Glycemia" and the MR-Sort parameters.

The resulting model given in Table 6.5 is computed in $40h33mn$ ($\approx$ 145,980s) execution time. The restoration rate of the assignment examples in the learning set is $CA = 99.4\%$. However, in the inferred model, the glycemia criterion is detected as a cost criterion. Let us note that the inferred value for the limit profile on the glycemia criterion (1.18 g/l) enables us to distinguish patients with hyperglycemia from the others, but does not distinguish hypoglycemia from normal glycemia (normal glycemia roughly corresponds to [0.9,1.2]). This is due to the distribution of the glycemia values over the patients shown in Figure 6.4. This distribution gives use the following observations :

- All patients with glycemia higher than 1.2g/l (hyperglycemia) are assigned to *Category 1*.
- Some patients with normal glycemia $[0.9, 1.2]$ g/l are also assigned to *Category 1*,
- Some patients with glycemia equal to 0.8 g/l or less (hypoglycemia) are assigned to *Category 2*.

Attributes	**Instance settings**			**Model parameters learned**		
	#values	Direction	pref. dir.	b_i	w_i	pref. dir.
Age	103 (origin)	cost	known	72.9	0.01	_
Diabetic	2 (origin)	cost	known	0.99	0	_
Hypertension	2 (origin)	cost	known	0	0.01	_
Respiratory F	2 (origin)	cost	known	0.99	0.88	_
Pacemaker	2 (origin)	cost	known	0	0.02	_
Systolic BP	24 (origin)	cost	known	15	0.03	_
Diastolic BP	17 (origin)	cost	known	8.92	0.02	_
Glycemia	82 (origin)	SP	unknown	1.18	0.03	*cost*
				$\lambda =$	0.98	

Table 6.5: Inferred model with the first dataset (898 assignment examples)

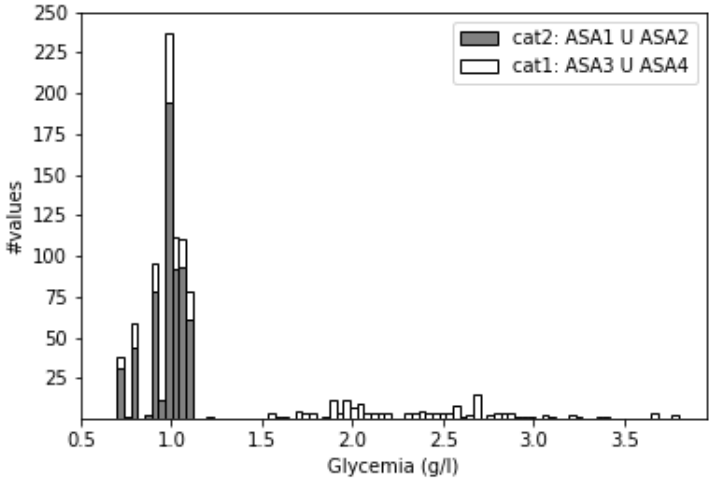


Figure 6.4: Distribution of patients' glycemia in the first dataset

Second Dataset

Continuing our study, we investigate on ways to restore the "correct" preference direction (i.e. single-peaked) of "Glycemia" with a subset of carefully selected patients. Thus, in the second step, we choose to remove from the first dataset the 97 patients assigned to *Category 1* and whose glycemia values is within $[0.9, 1.2]$ g/l (i.e., with normal glycemia). We aim at retrieving a single-peaked preference for the "Glycemia" criterion. The distribution of glycemia values in the new learning set of the remaining 801 patients is provided in Figure 6.5.

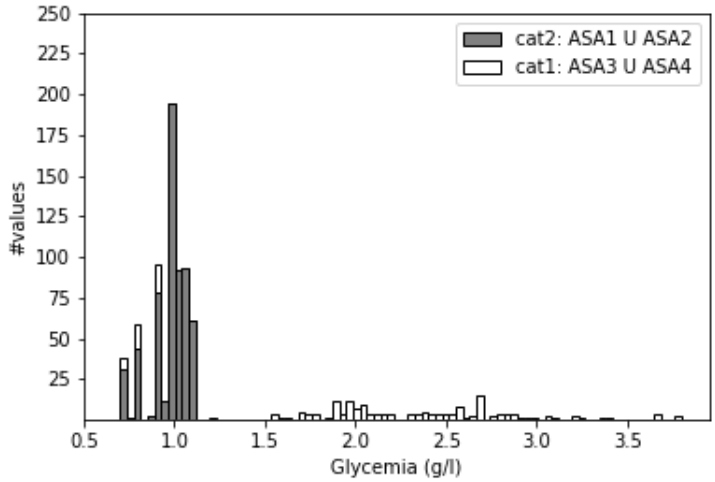


Figure 6.5: Distribution of patients' glycemia in the second dataset

We solve the inference problem with the MIP algorithm using this second learning set. The computation time falls to $56mn$ (≈ 3360s), and the inferred model (see Table 6.6) restores 99.8% of the learning set. Once again, the restoration rate is high, but the algorithm still detects a cost criterion for the glycemia criterion. The outputted model does not make any difference between patients with hypoglycemia and those with normal glycemia.

Third Dataset

At the final step, we remove from the second dataset, patients in *Category 2* for which the glycemia value is less than 0.9 (hypoglycemia). This new configuration leads to a dataset of 624 patients. In this last dataset, patients suffering from hypoglycemia or hyperglycemia are both assigned to *Category 1* while patients with normal glycemia are assigned to *Category 2*. This illustration is depicted by the histogram in Figure 6.6.

Attributes	Instance settings			Model parameters learned		
	$\#values$	Direction	pref. dir.	b_i	w_i	pref. dir.
Age	103 (origin)	cost	known	5.9	0	_
Diabetic	2 (origin)	cost	known	0.99	0	_
Hypertension	2 (origin)	cost	known	0	0.01	_
Respiratory F	2 (origin)	cost	known	0	0.01	_
Pacemaker	2 (origin)	cost	known	-0.01	0	_
Systolic BP	23	cost	known	15	0.01	_
Diastolic BP	15	cost	known	8.5	0.01	_
Glycemia	82 (origin)	SP	unknown	1.18	0.96	*cost*
				$\lambda =$	0.99	

Table 6.6: Inferred model with the second dataset (801 assignment examples)

With this dataset, the MIP algorithm runs in $4mn30s$ ($\approx 270s$) and results are given in Table 6.7. The computed model restores all the assignment examples, and the MIP finally identifies glycemia as single-peaked criterion. In addition, the approved values [0.93, 1.18] can be reasonably interpreted as normal glycemia.

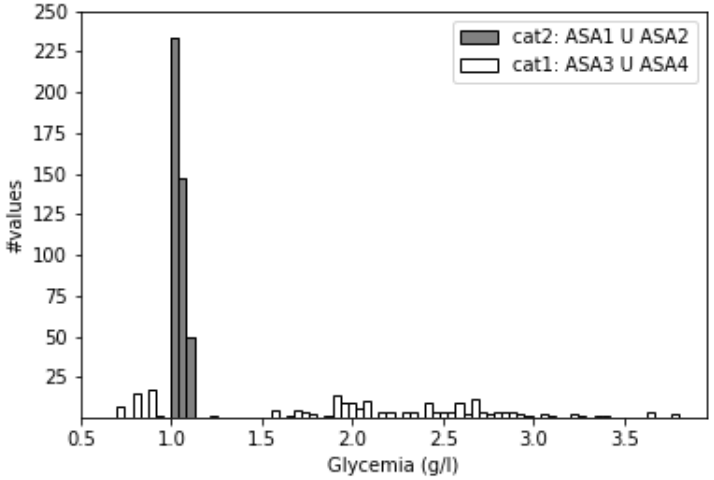


Figure 6.6: Patients' glycemia in the third dataset

This illustrative example shows that our model is able to infer an MR-Sort model and to retrieve single-peaked criteria. However, the learning set should be sufficiently informative i.e should encompass more assignment examples that are not redundant.

Attributes	Instance settings			Model parameters learned		
	$\#values$	Direction	pref. dir.	b_i	w_i	pref. dir.
Age	103 (origin)	cost	known	3.3	0	_
Diabetic	2 (origin)	cost	known	0	0	_
Hypertension	2 (origin)	cost	known	0	0	_
Respiratory F	2 (origin)	cost	known	0.99	0	_
Pacemaker	2 (origin)	cost	known	0	0	_
Systolic BP	23	cost	known	12.88	0.01	_
Diastolic BP	15	cost	known	9	0.01	_
Glycemia	73 (origin)	SP	unknown	[0.93,1.18]	0.99	SP
				$\lambda =$	1	

Table 6.7: Inferred model with the third dataset (624 assignment examples)

Typically, a carefully crafted set of assignment examples which takes into account all the combinations of alternatives values (e.g. values randomly drawn from each approved values sets of criteria) is useful for these learning tasks.

6.5 Conclusion

As a conclusion to our chapter, we proposed a MIP-based method to infer an MR-Sort model from a set of assignment examples when considering single-peaked/single-valley preferences. Our inference procedure enables simultaneously to retrieve from the dataset, an MR-Sort model and preference directions of criteria. Our experiments give good results, except it is limited by the size of the model which becomes rapidly intractable (200 alternatives, 4 criteria). Experiments suggest that the correct restoration of criteria preference directions requires datasets of significant size. The design of an efficient heuristic which is tractable with large datasets is the subject of the next chapter.

Chapter 7

An heuristic algorithm for the resolution of the Inverse MR-Sort problem with single-peaked preferences

Contents

7.1 Introduction and reminder

This chapter is about the implementation of an heuristic for the *Inverse-MR-Sort* problem with single-peaked preferences. In this chapter, we assume that the preference directions are known in advance including single-peaked/single-valley preferences. First, we prepare the ground for our proposal by introducing the principle. Second, we fully describe our approach which comprises two variants for the learning of MR-Sort parameters with single-peaked preferences in Section 7.2. In the third section, we carry out some experiments on both artificial data and real case instances. We analyse our results and compare them in Section 7.3.

7.1.1 Motivations and specificity of the approach

In the last chapter, we studied how to discover a single-peaked pattern on criteria given a learning set. On the one hand, the method performs two tasks : retrieving appropriate preference directions (*i*), and the rest of the MR-Sort parameters (*ii*). On the other hand, we show that experiments are not sustainable at large scale.

From these lessons, a compromise resides on focusing on a single task (learning MR-Sort parameters) while using an algorithm that is more scalable. Thus, we consider for this goal an heuristic-based approach. By doing that, we expect to reduce considerably the computation burden. In addition, we ground our method on the Sobrie's metaheuristic, which has been tested and proved to be efficient [90].

Therefore, we are interested in investigating an heuristic for the learning of MR-Sort models with single-peaked preferences (knowing in advance the preference directions of criteria).

Considering a learning set $L = \{(a, c(a)); \forall a \in A^\star \subset X\}$, we intend to learn the following parameters : $(\{w_i : i \in \mathcal{N}\}, \lambda, \langle b \rangle)$. Thus, we call this problem $IMSS$ (*Inverse-MR-Sort-SP*) which consists to retrieve the MR-Sort parameters - including single-peaked preferences - that best restore the assignment examples.

7.2 The heuristic-based method

The core of our contribution resides on the learning of single-peaked profiles (in order words, the values of the profile intervals $[\underline{b}_i, \overline{b}_i]$). The optimization of the weights is similar to that in Sobrie's metaheuristic [90].

Therefore in the following, we recall briefly and specifically the heuristic of Sobrie [90] that pertains to the learning of profiles. Then, we expose our heuristic in two variants. The first one consists to randomly and successively learn a first then a second interval value of the profiles of single-peaked criteria. Our second

variant consists in learning simultaneously both interval values of single-peaked criteria.

7.2.1 The Sobrie heuristic for the learning of profiles

The Sobrie's metaheuristic [90] processes profiles in two phases : the initialization phase and the optimization phase.

Therefore, we propose to detail these two phases which are useful for their adaptation to single-peaked criteria.

Initialization of profiles

First, we recall the initialization algorithm described in [90] (see Algorithm 3). The goal of this phase is to construct heuristically profiles values i.e by leveraging the information contained in the learning set.

Let us define by $A_i^{\star h}$, the set of evaluations on criterion i of alternatives assigned in category C^h, $\forall i \in \mathcal{N}$, $\forall h \in \{1, ..., p\}$. In other words, $A_i^{\star h} = \{\forall a_i^j \in A^\star : c(a^j) = C^h\}$. The rationale of Algorithm 3, is to allocate to b_i^h a value that plays the most discriminating role in assigning alternatives into categories (i.e. optimizes the classification accuracy).

The construction of b_i^h takes into account only 2 sets $A_i^{\star h}$ and $A_i^{\star h+1}$, $\forall h \in \{1, ..., p-1\}$. Thus, the set of candidate values is $A_i^{\star h} \cup A_i^{\star h+1}$.

Assuming w.l.o.g. that i is a gain criterion, ideal values for b_i^h are alternative evaluations a_i^j such that alternatives assigned to C^h or above (resp. strictly below C^h) , are those whose evaluations on i are greater (resp. lower) than a_i^j. In addition, the disproportion between the size of the sets $A_i^{\star h}$ and $A_i^{\star h+1}$ is also taken into account for the choice of b_i^h.

Therefore, the candidate values $b \in A_i^{\star h} \cup A_i^{\star h+1}$ for b_i^h are assessed according to the following score [90]:

$$\begin{aligned} score(b) = \quad & (|\{a_i \in A_i^{\star h+1}; a_i \geq b\}| - |\{a_i \in A_i^{\star h+1}; a_i < b\}|) \times (1 - \frac{|A_i^{\star h+1}|}{|A_i|}) \\ & + (|\{a_i \in A_i^{\star h}; a_i < b\}| - |\{a_i \in A_i^{\star h}; a_i \geq b\}|) \times (1 - \frac{|A_i^{\star h}|}{|A_i|}), \\ & b \in A_i^{\star h} \cup A_i^{\star h+1} \end{aligned} \tag{7.1}$$

To sum up, the choice of b_i^h depends on three factors :

- the maximum and the gap between the number of alternatives whose evaluations are greater than b_i^h and those whose evaluation are lower than b_i^h.

Algorithm 3: Initialization of profile values (Sobrie's algorithm [90])

Input: Subsets of the evaluations (on criterion i) of alternatives (assigned to category C^h) : $A_i^{\star h}, \forall i \in \mathcal{N}, \forall h \in \{1, ..., p\}$

1 **foreach** *profile* $b^h \in \{b^1, ..., b^{p-1}\}$ **do**
2 **foreach** *criterion i randomly chosen* **do**
3 $b_i^h = \underset{b \in A_i^{\star h} \cup A_i^{\star h+1}}{\text{argmax}} \; score(b)$;
4 **end**
5 **end**

- the representativeness over $A_i^\star$ of $A_i^{\star h}$ ($\frac{|A_i^{\star h}|}{|A_i|}$) or $A_i^{\star h+1}$ ($\frac{|A_i^{\star h+1}|}{|A_i|}$). In the score, this factor is used in order to balance under-represented or over-represented sets $A_i^{\star h}$ and $A_i^{\star h+1}$.

The same score is used in Algorithm 3 line 3. The final value of b_i^h is the candidate value $b \in A_i^{\star h} \cup A_i^{\star h+1}$ whose score is the maximum.

Optimization of profiles

Here, we first give some details about the optimization of a single value b_i^h (described in Algorithm 4) before exposing the complete heuristic for the entire profile $\langle b \rangle$ (described in Algorithm 5).

The objective of the optimization of profiles is to readjust them after possible changes occurred during the optimization of weights and the majority threshold. First, the procedure of Algorithm 4 takes as input : A_i^*, the set of alternatives evaluations on criterion i, and $I = [b_i^{h-1}, b_i^{h+1}]$, the interval of moves of b_i^h.

Then, we define $B_i^h = \{a_i \in A_i^* : b_i^{h-1} < a_i < b_i^{h+1}\}$ as the set of possible moves of b_i^h (Algorithm 4 line 1). To evaluate the quality of each move m, the choice of m is made as a compromise between the two following measures :

1. the desirability ratio of the move, noted r_m, which is an evaluation of the gain of better assignments obtained by the move (Algorithm 4 line 4).

2. the distance between the current profile value b_i^h and the next move m : $d(b_i^h, m) = |b_i^h - m|$. The goal of this measure is to encourage extreme moves (moves with a high $d(b_i^h, m)$ value) which helps to avoid local optima (Algorithm 4 line 5).

We introduce the following sets in order to compute the desirability ratio. It is the same sets as described in [90] in which we use m as indices instead of δ ($\delta = b_i^h - m$) :

Algorithm 4: Improvement of the profile value given the criterion i, and category h

Input: Current profile value : b_i^h
The interval of moves of b_i^h : $I = [b_i^{h-1}, b_i^{h+1}]$
The set of alternatives evaluations on criterion i : A_i^*
Output: New profile value b_i^h

```
Construct B_i^h = {a_i ∈ A_i^* : b_i^{h-1} < a_i < b_i^{h+1}}, the set of possible moves of
  b_i^h ;
m = b_i^h, r_m = 0;
foreach move m' randomly chosen in B_i^h do
    Compute r_{m'}, the desirability ratio of the move m' ;
    Compute the d(b_i^h, m') = |b_i^h − m'|, the distance between b_i^h and m';
    if r_{m'} ≥ r_m and d(b_i^h, m') ≥ d(b_i^h, m) then
        m = m' ;
        r_m = r_{m'};
    end
end
Draw a random number ε ∈ [0, 1] ;
if ε ≤ r_m then
    b_i^h = m;
    Update the alternatives assignments ;
end
```

- $V_{i,m}^h$: the set of alternatives misclassified in C^{h+1} instead of C^h (or C^h instead of C^{h+1}), for which the move m on the criterion i leads to a correct assignment.
- $Q_{i,m}^h$: the sets of alternatives correctly classified in C^{h+1} (or C^h) for which the move m on the criterion i leads to a misclassification.
- $W_{i,m}^h$: the set of alternatives misclassified in C^{h+1} instead of C^h (or C_h instead of C^{h+1}), for which the move m on the criterion i does not lead to a correct assignment, but strengthens the criteria coalition in favor of a correct classification.
- $R_{i,m}^h$: the sets of alternatives misclassified in C^{h+1} instead of C^h (or C^h instead of C^{h+1}), for which the move m on the criterion i does not lead to a correct assignment, but weakens the criteria coalition in favor of the correct classification.
- $T_{i,m}^h$: the sets of alternatives misclassified in a category higher than C^{h+1}

(or in a category lower than C^h) for which the move m on the criterion i strengthens the criteria coalition in favor of a classification that comes closer to the correct one.

The rationale of the desirability score is to foster the choice of values that optimize the classification accuracy (CA). Therefore the desirability ratio is :

$$r_m = \frac{k_V|V^h_{i,m}| + k_W|W^h_{i,m}| + k_T|T^h_{i,m}| + k_Q|Q^h_{i,m}| + k_R|R^h_{i,m}|}{d_V|V^h_{i,m}| + d_W|W^h_{i,m}| + d_T|T^h_{i,m}| + d_Q|Q^h_{i,m}| + d_R|R^h_{i,m}|} \tag{7.2}$$

In this equation, k_V,k_W,k_T,k_Q,d_V,d_W,d_T,d_Q and k_V represent empiric coefficients (set in [90, 88]) that enable to stress the importance of some sets in comparison to others.

The algorithm of [90] considers the optimal move m in B^h_i as the one that maximizes the desirability score while also maximizing the distance between the move and the current profile value b^h_i (Algorithm 4: lines 6 to 9).

Once such move m has been preselected, the achievement of m is conditioned by a random factor, which prevents bad moves (for instance moves with very small r_m) to be achieved (Algorithm 4 lines 11-12). Finally, if the move m has been achieved, the assignment examples are also updated, as well as the CA score (Algorithm 4 line 14).

After the improvement of a single value b^h_i, it is trivial to achieve the update of the complete profile $\langle b \rangle$.

As shown in Algorithm 5, it consists in readjusting one after the other, each frontier profile $b^h, h \in \{1, ..., p-1\}$ that delimits categories . For each frontier $b^h = (b^h_1, ..., b^h_i, ..., b^h_n)$, the update of profiles values b^h_i (performed by Algorithm 4) is random with regards to criteria. This is key since it randomizes the construction order of the components of profiles (Algorithm 5).

Algorithm 5: Initial algorithm of Sobrie [90] for the update of profiles

Input: Current profile b
Output: The improved profile b

```
1 foreach profile b^h ∈ {b^1, ..., b^{p-1}} do
2 |   foreach criterion i randomly chosen do
3 |   |   Run Algorithm 2 on b^h_i with the parameters I = [b^{h-1}_i, b^{h+1}_i], A^*_i ;
4 |   end
5 end
```

In the next subsections, we introduce the adaptations of the learning of profiles when the learning set reflects single-peaked preferences. We explain how to initialize and learn the profiles of single-peaked criteria.

7.2.2 Initialization of single-peaked profiles

In order to heuristically compute initial values of the single-peaked profiles, we describe the following algorithm (Algorithm 6). Let us reconsider the sets $A_i^{\star h}$, $\forall i \in \mathcal{N}$, $\forall h \in \{1, ..., p\}$, described previously as $A_i^{\star h} = \{\forall a_i^j \in A^\star : c(a^j) = C^h\}$. Let us assume w.l.o.g. that i is a single-peaked criterion. Here, we consider the initialization of profile intervals values $\underline{b}_i^h, \overline{b}_i^h$, $\forall h \in \{1, ..., p-1\}$, $\forall i \in \mathcal{N}$.

The construction of the profiles begins with inner intervals (i.e. $[\underline{b}_i^{p-1}, \overline{b}_i^{p-1}]$). Let $b_i^{\perp p-1} = \frac{\sum\limits_{a_i^j \in A_i^{\star p}} a_i^j}{|A_i^{\star p}|}$, the mean value of evaluations comprised in $[\underline{b}_i^{p-1}, \overline{b}_i^{p-1}]$, which is an estimation of the middle of the interval profile $[\underline{b}_i^{p-1}, \overline{b}_i^{p-1}]$ (Algorithm 6 line 2).

Then, we approximate $\underline{b}_i^{p-1}$ (resp. $\overline{b}_i^{p-1}$) by averaging evaluations $a_i^j \in A_i^{\star p-1}$ that are lower (resp. greater) than $b_i^{\perp p-1}$ (Algorithm 6 lines 3-4).

We iterate the same reasoning throughout the categories until the last iteration which pertains to the construction of $\underline{b}_i^1$ and $\overline{b}_i^1$

Algorithm 6: Initialization algorithm for single-peaked profiles

Input: Single-peaked criterion i,
$A_i^{\star h}, \forall h \in \{1, ..., p\}$
Output: The initial value of profile b_i^h, $\forall h \in \{1, ..., p-1\}$

1 **foreach** *category* $h \in \{p-1, ..., 1\}$ **do**
2 $\quad b_i^{\perp h} = \frac{\sum_{a_i^j \in A_i^{\star h+1}} a_i^j}{|A_i^{\star h+1}|}$;
3 $\quad \underline{b}_i^h = \frac{\sum_{a_i^j \in A^{\star h} : a_i^j < b_i^{\perp h}} a_i^j}{|A_i^{\star h}|}$;
4 $\quad \overline{b}_i^h = \frac{\sum_{a_i^j \in A^{\star h} : a_i^j > b_i^{\perp h}} a_i^j}{|A_i^{\star h}|}$;
5 **end**
6 **foreach** *category* $h \in \{p-1, ..., 2\}$ **do**
7 $\quad$ **if** $\underline{b}_i^{h-1} \geq \underline{b}_i^h$ **then** $\underline{b}_i^{h-1} = \underline{b}_i^h$;
8 $\quad$ **if** $\overline{b}_i^h \geq \overline{b}_i^{h-1}$ **then** $\overline{b}_i^{h-1} = \overline{b}_i^h$;
9 **end**

At the end of Algorithm 6, it is not guaranteed that profiles intervals are properly embedded. Therefore, lines 6-9 of Algorithm 6 enable to meet the following conditions : $\underline{b}_i^{h-1} \leq \underline{b}_i^h$ and $\overline{b}_i^h \leq \overline{b}_i^{h-1}$, $\forall h \in \{p-1, ..., 2\}$

In the following we expose our two variants of the heuristic for the optimization of single-peaked profiles. The first one consists in updating both values of the

profile intervals successively, and the second consists in updating both values simultaneously.

7.2.3 First strategy for learning of single-peaked profiles

With single-peaked criteria, the Sobrie heuristic for learning profiles in [90] is no more applicable since there are two profile values per single-peaked criterion, per category.

Our first strategy consists of learning successively and randomly the two intervals bounds of single-peaked profiles (see Algorithm 7).

Let us assume w.l.o.g. that i is a single-peaked criterion.

First, the interval bounds are chosen randomly (Algorithm 7 lines 6-7). Lines 8-9 and lines 11-12 of Algorithm 7 show two orders of learning the bounds : whenever we learn $\underline{b}_i^h$ (with Algorithm 4), $\overline{b}_i^h$ is fixed, and vice-versa. The optimization run of both bounds is performed in one iteration of the inner loop of Sobrie algorithm [90].

7.2.4 Second strategy for learning of single-peaked profiles

Assuming i a single-peaked criterion, our goal is to learn the lower bound $\underline{b}_i^h$ and the upper bound $\overline{b}_i^h$, which are the interval profiles of i. Our second strategy consists in simultaneously learning $\underline{b}_i^h$ and $\overline{b}_i^h$ of this criterion (see Algorithm 8).

At each iteration for the optimization of the profiles, we enumerate all the possible moves of both bounds namely $B_i^h = \underline{B}_i^h \times \overline{B}_i^h$, the Cartesian product of the set of moves of each bound (see Algorithm 8 line 7).

Similarly to the case of monotone criteria, we evaluate a couple of moves (m, m') (m for the lower bound and m' for the upper bound of the interval profile) regarding two aspects :

- the desirability ratio $r_{(m,m')}$ of the couple (m, m'), which indicates the level of improvement in the classification accuracy. It is computed with the formula of Equation 7.2, except that the sets of possible moves are constructed in accordance with the single-peaked construction (see Algorithm 8 line 11).

- the distance between the two intervals $[\underline{b}_i^h, \overline{b}_i^h]$ and $[m, m']$ (see Algorithm 8 line 12). In our case with single-peaked criteria, we adapt this measure by considering the following distance between two intervals of profiles :

$$d([\underline{b}_i, \overline{b}_i], [\underline{b'}_i, \overline{b'}_i]) = |\underline{b'}_i - \underline{b}_i| + |\overline{b'}_i - \overline{b}_i| \tag{7.3}$$

  with $[\underline{b}_i, \overline{b}_i]$ the current interval profile, $[\underline{b'}_i, \overline{b'}_i]$ the new interval profile.

Algorithm 7: Reformulation of Sobrie's algorithm for the learning of profiles with SP/SV criteria (variant 1)

Input: Current profile b
Output: The improved profile b

foreach *profile* $b^h \in \{b^1, ..., b^{p-1}\}$ *randomly chosen* **do**
  foreach *criterion* i *randomly chosen* **do**
    if i *is a gain or a cost criterion* **then**
      Run Algorithm 2 on b_i^h with the parameters $I = [b_i^{h-1}, b_i^{h+1}]$, A_i^* ;
    else
      Draw a random number $\epsilon \in [0, 1]$;
      if $\epsilon < 0.5$ **then**
        Run Algorithm 2 on $\underline{b}_i^h$ with the parameters $I = [\underline{b_i^{h-1}}, \underline{b_i^{h+1}}]$, A_i^* considering $\overline{b}_i^h$ fixed ;
        Run Algorithm 2 on $\overline{b}_i^h$ with the parameters $I = [\overline{b_i^{h-1}}, \overline{b_i^{h+1}}]$, A_i^* considering $\underline{b}_i^h$ fixed ;
      else
        Run Algorithm 2 on $\overline{b}_i^h$ with the parameters $I = [\overline{b_i^{h-1}}, \overline{b_i^{h+1}}]$, A_i^* considering $\underline{b}_i^h$ fixed ;
        Run Algorithm 2 on $\underline{b}_i^h$ with the parameters $I = [\underline{b_i^{h-1}}, \underline{b_i^{h+1}}]$, A_i^* considering $\overline{b}_i^h$ fixed ;
      end
    end
  end
end

After the evaluation of the quality of the couple of moves, we select the optimal couple : the one that maximizes the desirability score $r_{(m,m')}$ and the distance measure regarding the current position (i.e. $[\underline{b'}_i, \overline{b'}_i]$) (see Algorithm 8 lines 13-16). Once again, we achieve the move if $r_{(m,m')}$ is be sufficiently strong, and also update the resulting changes in assignment examples (see Algorithm 8 lines 18-22).

Strategy for the selection process of the profile moves

For both variants, we describe in the following a local heuristic for the selection process of the profiles moves. In the previous versions, the selection of move occurs only when a better score is simultaneously found in terms of desirability score ($r_{(m,m')}$) and distance ($d([\underline{b}_i, \overline{b}_i], [\underline{b'}_i, \overline{b'}_i])$).

A slightly different selection strategy consists in calculating a score, which is a

Algorithm 8: Sobrie's algorithm for the learning of profiles adapted to SP/SV criteria (variant 2)

Input: Current profile b
Output: The improved profile b

foreach *profile* $b^h \in \{b^1, ..., b^{p-1}\}$ *randomly chosen* **do**
  foreach *criterion* i *randomly chosen* **do**
    if i *is a gain or a cost criterion* **then**
      Run Algorithm 2 on b_i^h with the parameters b_i^{h-1}, b_i^{h+1}, A_i^* ;
    else
      $\underline{I} = [\underline{b_i^{h-1}}, \underline{b_i^{h+1}}]$, $\overline{I} = [\overline{b_i^{h-1}}, \overline{b_i^{h+1}}]$;
      Define $\underline{B}_i^h = \{a_i \in A_i^* : \underline{b_i^{h-1}} < a_i < \underline{b_i^{h+1}}\}$,
       $\overline{B}_i^h = \{a_i \in A_i^* : \overline{b_i^{h-1}} < a_i < \overline{b_i^{h+1}}\}$;
      Build $B_i^h = \{(m, m') : \forall m \in \underline{B}_i^h, \forall m' \in \overline{B}_i^h\}$, the set of couple of moves ;
      $(m, m') = (\underline{b}_i^h, \overline{b}_i^h)$, $r_{(m,m')} = 0$;
      foreach *couple of moves* (m'', m''') *randomly chosen in* B_i^h **do**
        Compute $r_{(m'',m''')}$, the desirability ratio of the couple (m'', m''') ;
        Compute the $d([\underline{b}_i^h, \overline{b}_i^h], [m'', m''']) = |\underline{b}_i^h - m''| + |\overline{b}_i^h - m'''|$, the distance between the two intervals $[\underline{b}_i^h, \overline{b}_i^h]$ and $[m'', m''']$;
        if $r_{(m'',m''')} \geq r_{(m,m')}$ *and* $d([\underline{b}_i^h, \overline{b}_i^h], [m'', m''']) \geq d([\underline{b}_i^h, \overline{b}_i^h], [m, m'])$ **then**
          $(m, m') = (m'', m''')$;
          $r_{(m,m')} = r_{(m'',m''')}$;
        end
      end
      Draw a random number $\epsilon \in [0, 1]$;
      if $\epsilon \leq r_{(m,m')}$ **then**
        $(\underline{b}_i^h, \overline{b}_i^h) = (m, m')$;
        Update of the alternatives assignments ;
      end
    end
  end
end

weighted sum of the two measures :

$$\frac{it * \omega_1}{\mathcal{N}_o} \quad r_{(m,m')} \quad + \quad \frac{(\mathcal{N}_o - it) * \omega_2}{\mathcal{N}_o} \quad d([\underline{b}_i, \overline{b}_i], [\underline{b'}_i, \overline{b'}_i])$$

, with ω_1, ω_2 two appropriate coefficients. Thus, the coefficients $\frac{it*\omega_1}{\mathcal{N}_o}$ and $\frac{(\mathcal{N}_o - it)*\omega_2}{\mathcal{N}_o}$ are function of the current iteration. On the one hand, the rationale is to select moves with large amplitude, which promote diversification of moves at the beginning of iterations. On the other hand, we choose moves with higher score of desirability, which foster the precision, near the end of iterations (maximum number of iterations).

In this section, we learn how to take into account single-peaked criteria in the *Inv-MR-Sort-SP* problem with two variants of the component of Sobrie's metaheuristic that learns profiles. For the sake of simplicity and readability, we specifically consider the case of single-peaked criteria on these proposals. Nevertheless, the same rationale can be applied in the case of single-valley criteria. As previously mentioned, the remaining of the algorithm (learning of weights and majority threshold, renewal process of the evolutionary algorithm) does not differ from the initial algorithm. In the rest of the chapter, we investigate the performance of our contributions through some numerical experiments.

7.3 Numerical tests and discussion

We detail in this section some experiments carried out in order to assess our algorithms. First, we present some results on the comparison between the two variants of our contribution : we compare and discuss on the advantages and disadvantages of our variants. As we solve the *MR-Sort-Inv-SP* problem, we are interested to evaluate the performance of our approaches in terms of computation time, classification accuracy in validation (CA_v) and the classification accuracy in generalization (CA_g) of the MR-Sort models learned.

Secondly, we extend our tests to larger datasets sizes and noisy data using the most appropriate variant. We consider large learning sets with 2000 assignment examples and noisy datasets (10%, 20% of erroneous data). We also conduct some experiments on real case datasets (ASA dataset [62] and instances from the UCI repository [29]).

7.3.1 Tests and comparisons between the two variants

In the following experiments, we consider the complete metaheuristic in order to learn the MR-Sort parameters (weights and profiles) including single-peaked criteria. Here is the following settings of this experiment. We consider 2 categories and 5 criteria in the models, including 4 gain criteria, and the last criterion that varies between gain, cost, single-peaked, and single-valley criterion. The learning set size is fixed to $|L| = 500$.

In order to generate the instances and the model parameters, we randomly draw alternative evaluations in $[0.05, 0.95]$. In addition, profiles evaluations were drawn in $[0.1, 0.9]$ for gain criteria, while for single-peaked criteria, $\underline{b}_i$ and $\overline{b}_i$ are randomly chosen in [0.1,0.9], with $\underline{b}_i < \overline{b}_i$. Weights are randomly distributed w_i in [0,1], $\forall i \in \mathcal{N}$ and $\lambda \in [0.5, 1]$.

To assess the performance of our algorithm in generalization, we run our method with 10000 alternatives as a test set. We also repeat our experiments 100 times in order to have a meaningful estimation of our metrics.

The algorithm parameters opted for this experiment are the same as the parameters chosen in the initial algorithm (i.e 20 iterations for the heuristic that updates the profiles, 30 iterations for the whole optimization of weights and profiles, and 10 models in the population)

We consider our two resolution strategies :

1. the first variant (successive update of the 2 limits profiles of the single-peaked interval) and the improvement for the selection of good moves.

2. the second variant (simultaneous update of the 2 limits profiles of the single-peaked interval) and the improvement for the selection of good moves.

Crit. type	Strategy	Time	CA_v (validation)	CA_g (generalization)
Gain crit.	_	15s	99%	98%
Cost crit.	_	16s	99%	98%
SP crit.	variant 1	24s	99%	98%
SV crit.	variant 1	33s	99%	98%
SP crit.	variant 2	$>$ 1h	99% (10 instances)	98% (10 instances)
SV crit.	variant 2	$>$ 1h	99% (10 instances)	98% (10 instances)

Table 7.1: Results of MR-Sort learning algorithm when varying the preference direction of one criterion (5 criteria, 2 categories, 500 alternatives). Tests on variant 2 carried out with only 10 problem instances because of the computation burden

The experiments results of Table 7.1 show that the variant 2 struggles to learn models in a reasonable time (more than 1 hour run) compared to the version 1.

This can be explained by the behaviour of variant 2 which is a greedy algorithm since it computes and compares at each iteration, all the possible moves which represent a cartesian product of two sets. The classification accuracy (CA_v and CA_g) is almost perfect ($\approx 99.9\%$), whenever the last criterion preference direction is single-peaked (SP) or single-valley (SV) and whatever the chosen method.

Despite the good results in classification accuracy of the second variant, the first variant outranks the second one in most of the cases because of its efficiency in terms of computation time. Nevertheless, the second variant could be an equally-ranked competitor to the first one if the evaluation set is small (i.e. small $|X_i|$ for single-peaked criteria i).

In the remaining of the experiments, we only use the first variant (and not the second variant for the sake of reducing computational burden) to solve *Inv-MR-Sort-SP* problems.

7.3.2 Advanced tests with the first variant on synthetic data

In this part, we extend our tests to more alternatives in the learning set and noisy datasets. We follow the analogous experimental protocol as in the previous tests, except some particularities.

Here, we consider in the MR-Sort models 7 criteria and 2 categories. We vary the following parameters :

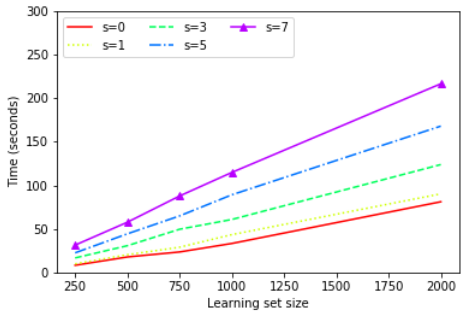
- the number of single-peaked criteria : $|\mathcal{S}| \in \{0, ..., 7\}$. Other criteria are gain criteria by default.
- the learning set size : $|L| \in \{250, 500, 750, 1000, 2000\}$
- the noise level : $\rho \in \{0, 0.1, 0.2\}$

In order to properly illustrate single-peaked criteria, we randomly generate alternatives but constrain the values in such a way to have a minimum of 20% of alternatives (L) in each category. The profiles values of gain criteria are randomly drawn in [0.1,0.9], whereas $\underline{b}_i$ is randomly chosen between the 2^{nd} and 4^{th} decile, and $\overline{b}_i$ is randomly chosen between the 6^{th} and 8^{th} decile of the evaluation scale X_i for single-peaked/single-valley criteria.

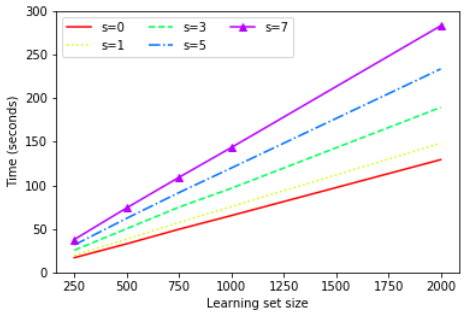
Results on computation time

We carried out some experiments related to computational time involving 7 criteria, 2 categories, and two degrees of noise in the dataset ($\rho \in \{0, 0.2\}$).

We first varied the number of single-peaked criteria in the model (see Figure 7.1a). We clearly observe a linear trend of the curves, and a slope that increase

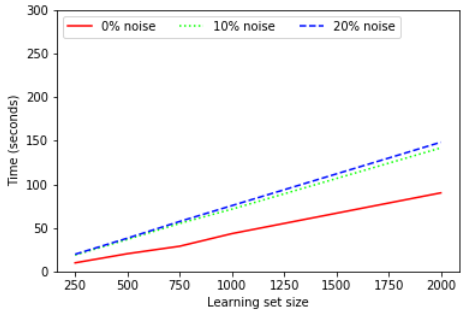


(a) Noise free datasets ($\rho = 0$)

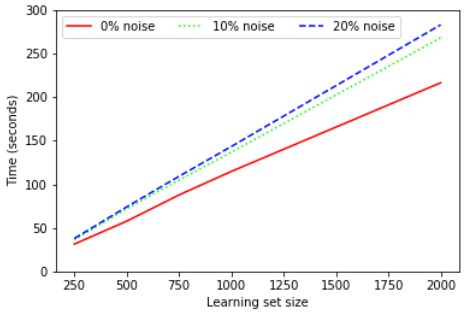


(b) Noisy datasets ($\rho = 0.2$)

Figure 7.1: Computation time (seconds) for the learning of problems involving 7 criteria, 2 categories and 2 levels of noise, per number of single criteria (s) and learning set size



(a) Case with 1 single-peaked criterion



(b) Case with 7 single-peaked criteria

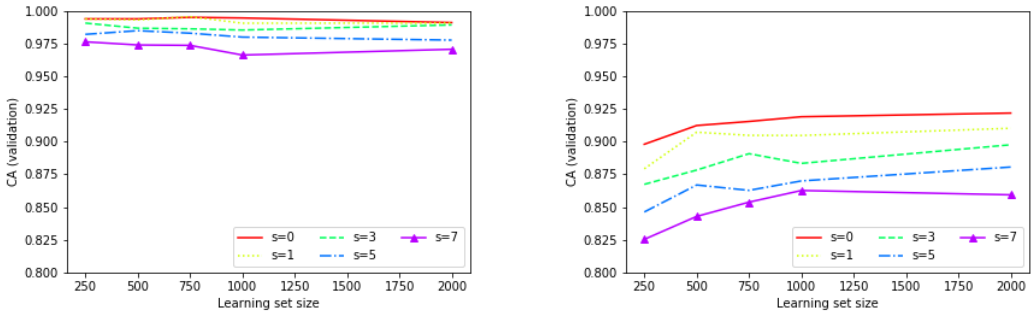
Figure 7.2: Computation time (seconds) for the learning of problems involving 7 criteria, 2 categories per level of noise and learning set size

with to the number of single-peaked criteria. Obviously, it demands more time to retrieve the parameters of more single-peaked criteria in the model since there are more profile values to learn. However, the algorithm requires only 300s (less than 5 minutes) for learning an instance of 2000 alternatives, which is fairly satisfactory. Besides, we still observe a linear curve (Figure 7.1b) on the computation time for instances with noisy datasets.

Furthermore, when contrasting the noisy free datasets ($\rho = 0$) and noisy datasets ($\rho = 0.1$ or $\rho = 0.2$) results (see Figure 7.2a and Figure 7.2b), there are two distinct curve slopes. This denotes a higher computation time for noisy datasets (independently of the level of noise) compared to the one with noise-free datasets. Seemingly, this behaviour does not depend on the number of single-peaked criteria

in the model.

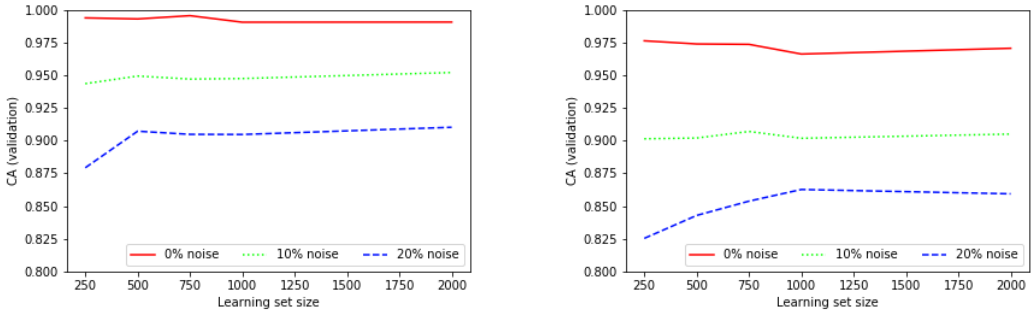
Results on classification accuracy in validation (learning phase)



(a) Classification accuracy in validation (with $\rho = 0$)

(b) Classification accuracy in validation (with $\rho = 0.2$)

Figure 7.3: Classification accuracy (CA) of the learning set : for problems involving 7 criteria, 2 categories and noisy-free learning sets per number of single-peaked criteria (s) and learning set size



(a) Classification accuracy in validation (with one single-peaked criterion)

(b) Classification accuracy in validation (with 7 single-peaked criteria)

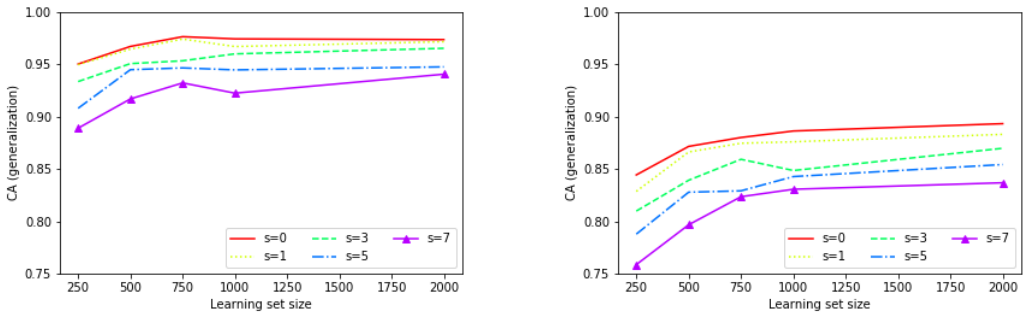
Figure 7.4: Classification accuracy in validation (CA_v) of the learning set : for problems involving 7 criteria, 2 categories per level of noise and learning set size

We test our algorithm on *Inv-MR-Sort-SP* problems with the same parameters as the latter experiments in order to assess classification accuracy of the learning set (CA_v) (see Figure 7.3a and Figure 7.3b). We clearly observe a difference of performance between noisy datasets (CA_v ranges between 0.825 and 0.925) and free

noise datasets ($CA_v > 0.95$). With noisy datasets, we remark a slight increase of the CA_v throughout the rise of assignment examples in learning sets. It is also more difficult to restore assignment examples with instances with more single-peaked criteria.

More generally, the results in Figure 7.4a and Figure 7.4b showed that the algorithm restores more than a 80% of assignment examples even with 20% of noise and 7 single-peaked criteria.

Results on classification accuracy in generalization



(a) Classification of the test set ($\rho = 0$) (b) Classification of the test set ($\rho = 0.2$)

Figure 7.5: Classification accuracy (CA) of the test set : for problems involving 7 criteria, 2 categories and 2 noise levels ($\rho = 0$ and $\rho = 0.2$) per number of single-peaked criteria (s) and learning set size

The last series of tests enables us to evaluate the algorithm in terms of classification accuracy in generalization (CA_g). We can observe a similar trend as the performance regarding CA_v.

In particular, the generalization CA_g ranges between 0.88 and 0.97 with noise free datasets while with noisy datasets, it ranges between 0.75 and 0.88, with a level of noise of 20%. We also notice an upward trend regardless the number of single-peaked criteria throughout the increase of the learning set.

In conclusion, results obtained through our algorithms with artificial data are instructive. Extended experiments shows that the gain obtained with large learning sets is moderate in terms of classification accuracy, and profitable in terms of computation time (compared to the execution time obtained with the exact method in Chapter 6). It could be informative to pursue some experiments regarding the learning performance with more than two categories, and different number of

criteria in the model. This would give us more insights on our approach and either confirm or not our present results.

7.3.3 Tests on ASA dataset

In this section, we run our algorithm on the ASA dataset (an extended presentation of the dataset is already given in Chapter 6). Briefly speaking, the ASA dataset is made of 898 patients considered as alternatives and 14 medical indicators considered as criteria. We choose to binarize this dataset into 2 categories : *category 2* which represents healthiest patients and *category 1*, the remaining. We recall the descriptive table of criteria of the dataset.

Attribute	Domain (Unit)	Direction
Age	$[0-105]$ (year)	cost
Diabetic	{0,1}	cost
Hypertension	{0,1}	cost
Respiratory failure	{0,1}	cost
Heart failure	{0,1}	cost
Heart rate	$[55-123]$ (bpm)	SP
Heart rate steadiness	{0,1}	gain
Pacemaker	{0,1}	cost
Atrioventricular block	{0,1}	cost
Left ventricular hypertrophy	{0,1}	min .
Oxygen saturation	$[43-100](\%)$	gain
Blood glucose level (glycemia)	$[0.5-3.8]$(g/l)	SP
Systolic blood pressure	$[9-20.5]$(cmHg)	cost
Diastolic blood pressure	$[5-13]$(cmHg)	*cost*

Table 7.2: Original criteria in the ASA dataset

For the sake of comparison with the exact method (the MIP) implemented in Chapter 6, we performed our heuristic algorithm on the same dataset as in Chapter 6. Hence, we only consider 8 criteria for this test (in bold in Figure 7.2). In Table 7.3, we show the results obtained after the execution of the metaheuristic.

One advantage of this algorithm resides on its efficiency on computation time. Compared to the exact method used to solve the problem in Chapter 6 (more than 40 hours of execution time), this metaheuristic runs in only 16min. In addition, we obtain for both methods the same performance in classification accuracy : 99.4%.

Another advantage of the metaheuristic is that it outputs not only the best model in terms of CA, but it also outputs the remaining models of the population.

Attributes	Instance settings			Model parameters learned		
	#values	Direction	pref. dir.	b_i	w_i	pref. dir.
Age	103 (origin)	cost	known	78.96.3	0.0001	_
Diabetic	2 (origin)	cost	known	1	0	_
Hypertension	2 (origin)	cost	known	0.99	0.0001	_
Respiratory F	2 (origin)	cost	known	0.6	0.9993	_
Pacemaker	2 (origin)	cost	known	0.99	0.0001	_
Systolic BP	23	cost	known	15.99	0.0001	_
Diastolic BP	15	cost	known	8.5	0.0001	_
Glycemia	73 (origin)	SP	known	[0.7,1.1]	0.0002	_
				$\lambda =$	0.9999	

Table 7.3: Inferred model with the ASA dataset (898 assignment examples)

Therefore, it is possible to choose among "optimal models", the one whose parameters are the most instructive.

7.3.4 Tests on public repository datasets

Wine Quality

For these experiments, we dispose of 2 datasets (*Red Wine Quality* and *White Wine Quality*) from the UCI Repository [1] [29]. As their names evokes, these datasets are about the classification made of a large set of wines. The first dataset contains 1599 red wines and the second 4898 white wines ; all are assessed on their qualities regarding 11 physicochemical properties considered as criteria.

We summarize in Table 7.4, the description of the criteria in our tests.

The considered categories varies from 3 (for poor quality) to level 9 (for excellent quality) with 6 levels for *Red Wine Quality* dataset and 7 levels *White Wine Quality* dataset.

We assume the criteria are single-peaked. One advantage of this assumption is to be able to capture even gain and cost criteria, in addition to single-peaked preferences.

Indeed, if the lower (resp. upper) bound of the learned profile interval is smaller (resp. greater) than the minimum (resp. maximum) value of the evaluation scale of the single-peaked criterion, we can easily deduce a cost (resp. gain) criterion since only the upper (resp. lower) bound of the learned profile interval is discriminating.

Unfortunately, it is impossible to capture single-valley preferences with a similar deduction. Reciprocally, if the criteria were single-valley criteria at the beginning, we could have deduced gain, cost criteria, but not single-peaked criteria.

[1] https://archive.ics.uci.edu/ml/datasets/wine

Attribute	Domain (Unit)	Direction
Fixed acidity	$[3.8 - 14.2]$ (g/dm^3)	SP
Volatile acidity	$[0.1 - 1.1]$ (g/dm^3)	SP
Citric acid	$[0 - 1.7]$ (g/dm^3)	SP
Residual sugar	$[0.6 - 65.8]$ (g/dm^3)	SP
Chlorides	$[0.01 - 0.35]$ (g/dm^3)	SP
Free sulfur dioxide	$[2 - 289]$ (mg/dm^3)	SP
Total sulfur dioxide	$[9 - 440]$ (mg/dm^3)	SP
Density	$[0.987 - 1.039]$ (g/cm^3)	SP
Ph	$[2.7 - 3.8]$	SP
Sulphates	$[0.2 - 1.1]$ (g/dm^3)	SP
Alcohol	$[8 - 14.2]$ (%)	SP

Table 7.4: Criteria of both datasets (*Red Wine Quality* and *White Wine Quality*) : 11 criteria

Firstly, we run our algorithm on both *Red Wine Quality* and *White Wine Quality* datasets in order to make a comparison with the algorithms tested in [29]. In order to compare our results, we perform 3 binary classifications, that result from three bi-partition of wine qualities :

1. **Experiment 1** : **Category 1** = qualities {3,4,5}. **Category 2** = qualities {6,7,8}.
2. **Experiment 2** : **Category 1** = qualities {3,4}, **Category 2** = qualities {5,6,7,8}.
3. **Experiment 3** : **Category 1** = qualities {3,4,5,6}, **Category 2** = qualities {7,8}.

Therefore, we run the metaheuristic on these 3 experiments

In [29], the authors achieved learning tasks on both datasets with an SVM algorithm.

First, we aggregate the entries of the confusion matrix of SVM (obtained in the paper), into three 2×2 confusion matrices following the three considered binary classifications. Therefore we deduce the three classification accuracy of the SVM from these 2×2 confusion matrices which correspond to the three binary classifications.

We report in Table 7.5) the summary of the classification accuracy obtained for SVM algorithm and our metaheuristic, for both (*Red Wine Quality* and *White Wine Quality* datasets considering the three binary classifications.

We observe that the classification accuracy obtained with both SVM and the metaheuristic are relatively close.

		SVM	Metaheuristic
Red Wine Dataset	**Experiment 1 cat 1 : 744, cat 2 : 855**	75.92%	73.54%
	Experiment 2 cat 1 : 63, cat 2 : 1536	95.93%	96.24%
	Experiment 3 cat 1 : 1382, cat 2 : 217	88.23%	88.99%
White Wine Dataset	**Experiment 1 cat 1 : 3258 , cat 2 : 1640**	79.93%	74.9%
	Experiment 2 cat 1 : 183, cat 2 : 4715	96.46%	96.36%
	Experiment 3 cat 1 : 3838 , cat 2 : 1060	74.92%	80.66%

Table 7.5: Comparative table of the performance on classification accuracy in validation of the SVM algorithm [29] and the metaheuristic

More specifically, the metaheuristic performs slightly better than the SVM algorithm in experiments 2 and 3 (which correspond to those with the most unbalanced datasets) in both datasets. Nevertheless, we advocate that MR-Sort is more interpretable than traditional machine learning models, particularly SVM.

Interpretability of MR-Sort models with single-peaked preferences

In the following, we give an illustration of how MR-Sort inferred models can be interpreted in the context of *Inv-MR-Sort-SP* problem.

For the sake of simplicity, we consider **Experiment 1**, the binary classification previously defined, and the *Red Wine Quality* dataset restricted to the seven most important attributes according to [29] as learning set. Criteria were assumed as single-peaked.

We reported in the Table 7.6, a learned model that involves only 5 criteria (since the weight of two criteria are 0). However, let us notice that $CA_v = 73.2\%$ for this model, which is close to the case with 11 criteria ($CA_v = 73.54\%$).

Given the profiles learned for single-peaked criteria, we discover these following preference directions :

- *Sulfate*, *Alcohol*, and *Fixed acidity* are single-peaked criteria.
- *Ph* is a gain criterion.
- *Total sulfur dioxide* is a cost criterion.

Attributes	Instance settings			Model parameters learned		
	$\#values$	Direction	pref. dir.	b_i	w_i	pref. dir.
Sulphates	96 (origin)	SP	unknown	[0.53,1.95]	0.0001	SP
Ph	89 (origin)	SP	unknown	[3.752,4.01]	0.4998	*gain*
Total sulfur dioxide	144 (origin)	SP	unknown	[6,88.8]	0.0001	*cost*
Alcohol	65 (origin)	SP	unknown	[9.81,14]	0.0001	SP
Volatile acidity	143 (origin)	SP	unknown	[0.21,0.55]	0	–
Free sulfur dioxide	60 (origin)	SP	unknown	[4.96,72]	0	–
Fixed acidity	96 (origin)	SP	unknown	[4.696,15]	0.4999	SP
				$\lambda =$	0.5002	

Table 7.6: Inferred model from 1599 assignment examples. $CA = 73.2\%$ Execution time : $3m55s$.

- The criteria types of *Free sulfur dioxide* and *Volatile acidity* cannot be directly determined since they do not count in this particular model (their weights are 0).

According to the model, we can determine the list of Minimal Sufficient Coalitions (MSC) that make it possible an interpretation of the model.

For instance, thanks to the two MSCs of the model, we can provide these simple decision rules :

- With the first MSC ($\{$*Ph*,*Fixed acidity*$\}$) : a wine is qualified as *Good* if the fixed acidity is between 4.696 g/dm^3 and 15 g/dm^3 **and** its Ph higher than 3.752.
- With the second MSC ($\{$*Sulfate*, *Total sulfur dioxide*, *Alcohol*, *Fixed acidity*$\}$) : a wine is qualified as *Good* if these conditions are fulfilled **at the same time** :
  - its sulftate content is between 0.53 g/dm^3 and 1.95 g/dm^3,
  - its content in total sulfur dioxide is lower than 88.8 mg/dm^3,
  - its alcohol content is between than 9.81 % and 14 %,
  - its fixed acidity is between 4.696 g/dm^3 and 15 g/dm^3.

7.3.5 Discussion

The experiments carried out in the previous sections are edifying and gives us more insights on single-peaked preferences on the scope of the learning of MR-Sort models.

First, we validated our two algorithm versions through the experiments with hand-made instances. Version 2 is suited for small instances (<50 alternatives), but can be helpful for an exhaustive search of compatible models. However, we focused on version 1, which is more practical (we use it in the other experiments).

Second, this algorithm was tested with large artificial data to evaluate its scalability (less than 5min for a dataset of 2000 alternatives and 7 criteria), and with noisy data to assess its robustness.

Learning single-peaked preferences with MR-Sort models does not influence importantly the CA_v and CA_g, and does not add large computational burden, even with noisy datasets.

The results showed that our algorithm still fairly learns in the presence of the noise.

Finally our experiments with real-world datasets are informative because it shows the practical aspect of our work. We are able to retrieve a good model ($CA_v = 99.4\%$) with ASA dataset in a limited time (16min). Regarding the instances related to wines, we are able to deal with large datasets and learn several single-peaked criteria. In addition, our algorithm is capable of learning simultaneously some preference directions and the usual parameters of MR-Sort. As our algorithm extends the metaheuristic developed for the learning of monotone preferences, we challenged our algorithm on several datasets from the UCI repository in the same spirit as [88].

Although the considered datasets were assumed to be monotone as mentioned in [42], the results obtained with our algorithm are fairly similar to the most related algorithms [42, 63, 88], and could lead to more investigations on relations between monotone and single-peaked preferences.

We refer the reader in the Appendices A for the details of these studies.

7.4 Conclusion

In this chapter, we presented an algorithm in order to solve the inverse-MR-Sort problem with single-peaked preferences, which consists in learning the parameters of an MR-Sort model from data with single-peaked preferences. Our method is based on the metaheuristic proposed by [90, 88]. We presented two variants, which adapt the learning of profiles. The experiments on synthesized data reveal good behaviours of our algorithm even with noisy learning sets. The extended tests carried out with real data, show that our algorithm is advantageous in terms of computational time and can compete with machine learning algorithms.

Chapter 8

Conclusion

In this thesis we have investigated how unknown monotone preference directions and unknown non monotone preferences – more precisely, single-peaked and single-valley – can be handled within an MR-Sort algorithmic framework. We have focused on MR-Sort as this model is interpretable.

Optimization techniques are often subject to a tradeoff between optimality and execution time. This is a natural compromise when dealing with large datasets. In this work, we have considered both aspects: we proposed exact solutions that are workable for "small" datasets and heuristic based methods to address the problem of "large" datasets. On the one hand, heuristics provide an approximated solution (more than 90% on classification accuracy for average instances) but allow to handle large learning set ; that is, in our case, over 1500 alternatives. On the other hand, exact methods give the optimal solution but their execution time becomes prohibitive as the dataset size increases.

The experimental study of the proposed methods stresses the importance of the quality of the learning sets. This quality can be quantified using two indicators : the representativeness (in percentage) of alternatives into categories and the choice of balanced parameters (i.e., model parameters with non-extreme values). Obviously, datasets with more balanced alternatives over the categories and models with balanced parameters lead to learn more accurate models.

The three main contributions of this thesis can be summarized as follows:

- **Contribution I**. We presented two heuristics to learn latent criteria preferences directions as well as the MR-Sort model parameters. These approaches are workable on large datasets and are based on the metaheuristic proposed in [90]: a duplicated-based approach (i) that globally outperforms the mixed-based approach (ii) - although (ii) is barely better than (i) in some cases. The main benefit is the ability of (i) to learn preference directions together with the other parameters without adding any difficulty whatever the number

of unknown preference direction in the input instance.

- **Contribution II**. We proposed an exact method - the Mixed Integer Program (MIP) - to detect and learn single-peaked and single-valley preferences. The characterization of single-peaked preferences for NCS and MR-Sort models that we establish suits with its use for retrieving gain, cost, single-peaked and single-valley criteria together with the other MR-Sort parameters. The method shows a good restoration rate comparable to those obtained in the **Contribution I**. However, this method is suited only for relatively small dataset as its computational complexity grows very quickly with the dataset size. Our experiments showed that the method provides an acceptable computation time for datasets up to 200 alternatives.

- **Contribution III**. We presented two heuristics to learn single-peaked preferences in large datasets. These two heuristics, the two-steps method (*iii*) and the brute force method (*iv*), produce comparable results. The brut force method, however, has an exponential execution time with the number of alternatives and is, therefore, not suitable for large datasets. Results obtained on both synthetic and real datasets show that the two-step method produces similar results to machine learning techniques (for instance, with SVM [29] on the *Wine dataset* from UCI repository).

In the future, our contributions can be extended in multiple directions:

1. In order to pursue the work initiated with the learning of single-peaked preferences in (**Contribution III**), we suggest to tackle the case where the preference directions are not known in advance. The approach proposed in **Contribution III** enables to detect either single-peaked or single-valley criteria , depending on the algorithm's settings. If it is accordingly configured, it can learn monotone preferences; however it cannot learn single-peaked and single-valley preferences at the same time. To be able to learn these different types of preference, we can extend the mixed-based approach of **Contribution I** to integrate the four types of criteria in the population of models, and foster the evolution of models with the correct preference directions. Such implementation would provide a method that can deal with the learning of the four preference direction types from large datasets.

2. We propose to investigate the learning of single-peaked criteria through a max-SAT formulation (generally known as the optimization version of the Boolean Satisfaction problem). In fact, this method was used for the learning of monotone preferences with MR-Sort and NCS models [8]. **Contribution**

II gave us not only the characterization of single-peaked criteria but also a transformation technique (from single-peaked to monotone criteria) in order to learn single-peaked criteria with the MIP algorithm. The use of binary variables, for this purpose, in the MIP can certainly be adapted to a Boolean formulation for a resolution with a Max-SAT algorithm. We expect that such a method could bring a solution possibly being a compromise between the evolutionary algorithm and the MIP algorithm. Hence, such a method could increase the accuracy while limiting the execution time [10].

3. Another direction, is to explore how single-peaked preferences can be learned in closer models. Such models can be, for example, MR-Sort with veto [91] and MR-Sort variants considering concordance, veto and dictator relations [67]. In the case of MR-Sort with veto, we could consider veto profiles as intervals similar to limit profiles for single-peaked criteria. In this case, veto profiles should be embedded in the limit profiles of a given single-peaked criterion. For the purpose of introducing single-peaked preferences with NCS models, an interesting starting point could be considering 2-additive MR-Sort models. These models are more expressive than MR-Sort, and single-peaked preferences could also improve the interpretability power of such models.

4. Finally, an interesting area is the study of preferences with multiple-peaked in MR-Sort models. In practice, such models would embody several disjoint intervals pertaining to values in favor of the same set of approved values. It could be possible to generalize this concept into non-monotone preferences. Therefore, such a criterion (a non-monotone criterion) could be composed of multiple local monotonicities). An example of these types of preferences is double-peaked preferences. In this context, two peaks are preferred instead of a single-peaked. In some cases, double-peaked preferences better reflects the reality e.g. for adopting public policies [40] where conservative and liberal policies are both preferred to a moderate *status quo*, or for facility location problems [46] where we may have two favorite places to build a primary school along a street.

In this thesis, we have extended significantly the use of MR-Sort techniques to be able to handle single-peaked and single valley criteria. We will soon apply these techniques in the context of industrial applications which will greatly benefit from these steps towards a generalization of criteria handling.

Appendix A

Comparative results of algorithms for monotone sorting problems on UCI instances

In this appendix, we present additional tests performed on the algorithm implemented in Chapter 7 for learning MR-Sort models from single-peaked preferences. We consider 8 data sets from the UCI machine learning repository and the WEKA repository detailled in Table A.1.

Data set	#instances	#attributes	#categories
DBS	120	8	2
CPU	209	6	4
BCC	286	7	2
MPG	392	7	36
ESL	488	4	9
MMG	961	5	2
ERA	1000	4	4
LEV	1000	4	5
CEV	1728	6	4

Table A.1: Data sets of UCI and WEKA repositories

In Table A.2, we give a comparative performance of the 0/1 loss measure obtained after running 5 different algorithms :

- META-SP : represent our algorithm (Chapter 7) based for the learning of MR-Sort models with single-peaked criteria.
- META is the original metaheuristic for the learning of MR-Sort models with monotone criteria [90].

- MIP is the mathematical programming algorithm for learning MR-Sort with monotone criteria in [63]. Some of these results have not been obtained because of large size of instances.

- UTADIS is the algorithm implemented in [56] for learning additive-based models with monotone criteria.

- CR refers the Choquistic Regression implemented for Preference Learning problems in [42].

We choose to transform datasets in binary classification instances and operate cross-validations for the learning process, considering 20%, 50% and 80% of each dataset as learning sets and the rest as test sets. We consider 100 different generations of these sets corresponding to 100 runs of META-SP. The setting of META-SP is the following : 10 iterations of the outer loop, 20 iterations of the inner loop, 10 models in the population. We reported from [88, 92], results of other algorithms.

We present in table A.2, the mean value of the 0/1 loss (and standard deviations) obtained when evaluating learned models on test sets. The size refers to the learning set size percentage of the dataset. The first row shows algorithms we compare. Some results have not been obtained with the MIP algorithm because of the large size of instances (computational difficulty).

Overall, the performance of META-SP are slightly outranked on average by others except for the instance MPG where META-SP is better than than META, MIP and UTADIS (by 7% on average) but is worse than CR (by 7% on average). It is interesting to note that in this case, META-SP can learn more complex features of criteria than META, but is still limited since CR learns more interactions between criteria. It is also worth to mention that on the instance ERA, our algorithm outperform META and CR by more than 8%. Standard deviations decreases in average with the learning set size. In addition, whenever the META-SP is worse than another algorithm in terms of 0/1 loss, the deviation is all the more bigger, which confirms that META-SP is not strictly outranked by others.

Size	Data set	META-SP	META	MIP	UTADIS	CR
	DBS	24.7 ± 6.15	18.97 ± 4.23	19.77 ± 4.81	20.08 ± 5.33	17.13 ± 4.24
	CPU	11.92 ± 5.12	9.94 ± 3.23	9.00 ± 3.45	6.52 ± 3.62	8.11 ± 1.03
	BCC	29.52 ± 3.84	28.24 ± 2.73	26.78 ± 2.76	29.15 ± 3.07	27.75 ± 3.35
	MPG	13.28 ± 2.4	20.25 ± 3.56	20.80 ± 3.26	22.25 ± 3.18	7.09 ± 1.93
20 %	ESL	11.77 ± 1.96	10.42 ± 1.71	10.75 ± 1.58	8.89 ± 1.60	6.82 ± 1.29
	MMG	18.45 ± 1.56	16.97 ± 0.87	17.16 ± 1.40	18.40 ± 1.84	17.25 ± 1.20
	ERA	21.25 ± 1.56	21.36 ± 2.05	20.93 ± 1.74	23.68 ± 1.87	28.89 ± 2.73
	LEV	17.5 ± 1.76	16.74 ± 1.87	16.08 ± 1.73	16.54 ± 1.60	14.99 ± 1.22
	CEV	11.46 ± 2.24	9.37 ± 1.12	-	7.94 ± 0.59	4.48 ± 0.89
	DBS	20.33 ± 5.45	16.23 ± 4.69	16.27 ± 4.26	14.80 ± 4.21	15.72 ± 4.16
	CPU	7.74 ± 3.6	6.75 ± 2.37	6.40 ± 2.39	2.30 ± 2.38	4.64 ± 2.81
	BCC	28.1 ± 3.17	27.50 ± 3.17	-	28.54 ± 2.46	26.87 ± 2.82
	MPG	11.24 ± 1.84	17.81 ± 2.37	-	20.90 ± 2.36	5.77 ± 2.51
50 %	ESL	10.88 ± 1.79	10.04 ± 1.86	10.18 ± 1.55	7.83 ± 1.63	6.01 ± 1.26
	MMG	17.68 ± 1.66	17.32 ± 1.51	-	17.58 ± 1.52	16.67 ± 1.44
	ERA	20.09 ± 1.26	20.56 ± 1.73	19.58 ± 1.37	23.42 ± 1.71	28.44 ± 3.06
	LEV	15.36 ± 2.	15.92 ± 1.22	14.22 ± 1.54	15.56 ± 1.32	13.72 ± 1.25
	CEV	10.74 ± 2.27	9.36 ± 1.19	-	7.99 ± 0.91	3.76 ± 0.59
	DBS	16.16 ± 6.13	15.92 ± 6.98	14.80 ± 8.11	12.80 ± 5.01	14.16 ± 6.81
	CPU	6.47 ± 3.85	6.40 ± 3.04	5.98 ± 3.15	1.52 ± 2.14	2.12 ± 3.01
	BCC	26.28 ± 5.94	26.77 ± 5.47	-	29.13 ± 5.10	24.96 ± 4.85
	MPG	11.13 ± 3.35	16.86 ± 3.69	-	20.80 ± 3.88	5.51 ± 1.60
80 %	ESL	10.67 ± 2.65	10.01 ± 2.97	10.08 ± 2.47	7.44 ± 2.35	5.42 ± 2.18
	MMG	17.39 ± 3.08	16.98 ± 2.79	-	17.34 ± 2.65	15.84 ± 2.51
	ERA	19.9 ± 2.88	20.31 ± 2.50	18.56 ± 2.60	23.56 ± 2.92	28.13 ± 2.80
	LEV	14.93 ± 2.66	16.16 ± 2.22	13.59 ± 1.85	15.72 ± 2.22	13.14 ± 1.76
	CEV	10.94 ± 2.48	9.66 ± 1.74	-	7.99 ± 1.32	2.73 ± 0.89

Table A.2: 0/1 Loss and standard deviations results of 5 algorithms on 8 datasets

Appendix B

Synthèse de la thèse en Français

Dans les problèmes de décision, la classification est le terme générique utilisé pour décrire l'affectation d'alternatives dans des catégories prédéfinies ; quand les catégories sont ordonnées, on parle de classification ordinale. Il existe plusieurs familles de classification ordinales repertoriées dans la littérature [54, 55, 93]. Cependant, nous nous interessons à la classification monotone un sous-cas de la classification ordinale qui se définit par l'existence d'une dépendance monotone entre les valeurs des critères et les classes d'affectation (catégories). En d'autres termes, en augmentant (resp. diminuant) la valeur du critère d'une alternative a, on contribue à affecter cette alternative dans une catégorie supérieure ou égale à a dans le cas d'une monotonie croissante (resp. décroissante).

Nous définisons la *classification partiellement monotone* comme le cas où la dépendance monotone entre critères et catégories est restreinte à un sous ensemble de critères B de l'ensemble des critères $\mathcal{N}$ ($B \subset \mathcal{N}$) [76]. Plus particulièrement les critères de l'ensemble B admettent une relation de monotonie avec les categories tandis que les critères $\mathcal{N} \setminus B$ établissent une relation non-monotone avec les catégories. Dans notre cas, la non-monotonie est une monotonie par morceaux et nous considérons particulièrement deux cas : soit une monotonie croissante puis décroissante (appelée *single-peaked*) soit une monotonie décroissante puis croissante (appelée *single-valley*). Dans ce contexte, nous désignons ainsi les critères B comme des critères monotones et les critères $\mathcal{N} \setminus B$ comme des critères single-peaked (ou single-valley).

Plusieurs travaux dans la littérature de l'aide à la décision multicritère ont été réalisés pour des problèmes de tri (classification ordinale) impliquant des critères non monotones soit avec des modèles additifs [22, 64, 58, 58, 53] ou avec des modèles à base de règles logiques [14].

Parmi les modèles de surclassement, ELECTRE TRI (Mousseau and Słowiński [71]) demeure le modèle majoritairement étudié pour les problèmes d'inférence de paramètres à partir d'exemples d'affectation [70, 73]. Le modèle considéré dans

notre travail est le modèle à règle majoritaire MR-Sort [63], un sous cas de NCS [17, 18] (modèle de tri non compensatoire) qui est lui même un modèle réduit de ELECTRE TRI [104, 44]. Le but de ces modèles, en particulier MR-Sort, est de classer les alternatives dans des catégories prédéfinies et ordonnées $C^p \triangleright \ldots \triangleright C^2 \triangleright C^1$ (avec C^p la meilleure catégorie et C^1 la pire catégorie).

Le problème *Inv-MR-Sort* est celui de l'inférence des paramètres du modèle MR-Sort à partir d'exemples d'affectation. Ce problème a été résolu avec une programmation mathématique en nombres entiers par Leroy *et al.* [63], par une métaheuristique évolutionnaire [92, 88], et par une formulation en problème de satisfiabilité booléenne [11, 8].

Cependant, ces méthodes de résolution ne prennent en compte que des données monotones. Dans notre travail, nous étendons cet existant pour prendre en compte des données single-peaked. De plus, nous portons notre analyse sur l'inférence de modèles MR-Sort dont les sens de préférences sur les critères ne sont pas connus d'avance contrairement à l'état de l'art où les sens de préférence sont connus d'avance. Comme point de départ pour deux de nos contributions, nous adaptons et améliorons l'approche métaheuristique présentée par Sobrie *et al.* [92, 88] pour résoudre nos deux principaux problèmes : le problème Inv-MR-Sort avec des sens de préférences inconnus et le problème de Inv-MR-Sort avec des données single-peaked/single-peaked (de petites ou grandes tailles).

Le plan de notre thèse se déroule comme suit.

Dans un premier temps, nous nous interessons à présenter brièvement des définitions et quelques de notions de l'aide à la décision multicritère ainsi que l'état de l'art - aussi bien sur l'inférence de modèles de surclassement comme MR-Sort, que sur les problèmes de tri à partir de données partiellement monotones.

Dans un deuxième temps, nous abordons le problème d'inférence des paramètres de modèles MR-Sort à partir de données dont les sens de préférences des critères sont monotones mais inconnus d'avance. Nous nous interessons aussi à apprendre les sens de préférences inconnus. Nous proposons deux approches de résolution : une basée sur la duplication des critères des sens de préférences inconnus, et une autre fondée sur l'introduction de modèles hétérogènes (c-à-d des modèles dont les critères ont des sens de préference différents).

Dans un troisième temps, nous proposons une formulation sous forme d'un programme mixte en nombres entiers pour apprendre des modèles MR-Sort à partir de préférences monotones et single-peaked/single-valley. Cette proposition est basée sur une transformation de critères single-peaked (resp. single-valley) en critères à sens de préférence décroissant (resp. croissant). Cette méthode permet de trouver des solutions optimales au problème d'inférence, mais est coûteuse en temps de calcul.

Quatrièmement, nous proposons deux heuristiques (basées sur [92, 88]) pour

résoudre le problème d'inférence de paramètres de MR-Sort à partir de préférences monotones et single-peaked/single-valley considérant des ensembles d'apprentissage de grandes tailles.

En dernier lieu, nous concluons ce document en rappelant nos contributions et en proposant des pistes de recherches découlant de notre travail.

Bibliography

[1] J. Alcala-Fdez, A. Fernández, J. Luengo, J. Derrac, S. Garc'ia, L. Sanchez, and F. Herrera. KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic and Soft Computing*, 17:255–287, 2010.

[2] K. Amel. From shallow to deep interactions between knowledge representation, reasoning and machine learning. In *Proceedings 13th International Conference Scalable Uncertainty Management (SUM 2019), Compiègne, LNCS*, pages 16–18, 2019.

[3] M. Amine Lazouni, M. Habib Daho, N. Settouti, M. Chikh, and S. Mahmoudi. Machine learning tool for automatic ASA detection. In *Modeling Approaches and Algorithms for Advanced Computer Applications*, pages 9–16. Springer, 2013.

[4] J. Anderson. Regression and ordered categorical variables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 46(1):1–22, 1984.

[5] C. Bana e Costa and J. Vansnick. MACBETH — an interactive path towards the construction of cardinal value functions. *International Transactions in Operational Research*, 1:489–500, 1994.

[6] C. Bana e Costa and J. Vansnick. General overview of the MACBETH approach. *Advances in multicriteria analysis*, pages 93–100, 1995.

[7] C. Bana e Costa and J. Vansnick. Applications of the MACBETH approach in the framework of an additive aggregation model. *Journal of Multi-Criteria Decision Analysis*, 6(2):107–114, 1997.

[8] K. Belahcène. *Towards accountable decision aiding : explanations for the aggregation of preferences*. PhD thesis, CentraleSupélec, Université Paris-Saclay, 2018.

[9] K. Belahcene, C. Labreuche, N. Maudet, V. Mousseau, and W. Ouerdane. Explaining robust additive utility models by sequences of preference swaps. *Theory and Decision*, 82(2):151–183, 2017.

[10] K. Belahcene, C. Labreuche, N. Maudet, V. Mousseau, and W. Ouerdane. An efficient SAT formulation for learning multiple criteria non-compensatory sorting rules from examples. *Computers and Operations Research*, 97:58–71, 2018.

[11] K. Belahcène, C. Labreuche, N. Maudet, V. Mousseau, and W. Ouerdane. An efficient SAT formulation for learning multiple criteria non-compensatory sorting rules from examples. *Computers & Operations Research*, 97:58–71, 2018.

[12] D. Black. On the rationale of group decision-making. *Journal of political economy*, 56(1):23–34, 1948.

[13] D. Black. *The theory of committees and elections*. University Press, Cambridge, 1958.

[14] J. Blaszczynski, S. Greco, and R. Slowinski. Inductive discovery of laws using monotonic rules. *Engineering Applications of Artificial Intelligence*, 25(2):284–294, 2012.

[15] J. Błaszczyński, S. Greco, R. Słowiński, and M. Szelag. Monotonic variable consistency rough set approaches. *International journal of approximate reasoning*, 50(7):979–999, 2009.

[16] G. Bous, P. Fortemps, F. Glineur, and M. Pirlot. ACUTA: A novel method for eliciting additive value functions on the basis of holistic preference statements. *European Journal of Operational Research*, 206(2):435–444, 2010.

[17] D. Bouyssou and T. Marchant. An axiomatic approach to noncompensatory sorting methods in MCDM, I: The case of two categories. *European Journal of Operational Research*, 178:217–245, 2007.

[18] D. Bouyssou and T. Marchant. An axiomatic approach to noncompensatory sorting methods in MCDM, II: More than two categories. *European Journal of Operational Research*, 178(1):246–276, 2007.

[19] Q. Brabant. *Fonctions latticielles polynomiales pour l'interpolation et la classification monotone*. PhD thesis, Université de Lorraine, 2019.

[20] Q. Brabant, M. Couceiro, D. Dubois, H. Prade, and A. Rico. Learning rule sets and Sugeno integrals for monotonic classification problems. *Fuzzy Sets and Systems*, 401:4–37, 2020.

[21] J. Brans. *L'ingénierie de la décision: l'élaboration d'instruments d'aide a la décision*. Université Laval, Faculté des sciences de l'administration, 1982.

[22] V. Bugera, H. Konno, and S. Uryasev. Credit cards scoring with quadratic utility functions. *Journal of Multi-Criteria Decision Analysis*, 11(4-5):197–211, 2002.

[23] J. Butler, J. Jia, and J. Dyer. Simulation techniques for the sensitivity analysis of multi-criteria decision models. *European Journal of Operational Research*, 103(3):531–546, 1997.

[24] J. R. Cano, P. A. Gutierrez, B. Krawczyk, M. Wozniak, and S. Garcia. Monotonic classification: An overview on algorithms, performance measures and data sets. *Neurocomputing*, 341:168–182, 2019.

[25] J. Chen, Z. Li, X. Wang, and J. Zhai. A hybrid monotone decision tree model for interval-valued attributes. *Advances in Computational Intelligence*, 2(1):1–11, 2022.

[26] W. Cheng, K. Dembczynski, and E. Hüllermeier. Graded multilabel classification: The ordinal case. In *ICML*, pages 223–230, 2010.

[27] M. Choulak, D. Marage, M. Gisbert, M. Paris, and Y. Meinard. A meta-decision-analysis approach to structure operational and legitimate environmental policies – With an application to wetland prioritization. *Science of the Total Environment*, 655:384–394, 2019.

[28] S. Corrente, S. Greco, M. Kadziński, and R. Słowiński. Robust ordinal regression in preference learning and ranking. *Machine Learning*, 93(2):381–422, 2013.

[29] P. Cortez, A. L. Cerdeira, F. Almeida, T. Matos, and J. Reis. Modeling wine preferences by data mining from physicochemical properties. *Decis. Support Syst.*, 47:547–553, 2009.

[30] IBM ILOG Cplex. *IBM ILOG CPLEX Optimization Studio CPLEX User's Manual, Version 12, Release 8*. IBM ILOG, 20.1.0 edition, 2017.

[31] D. K. Despotis and C. Zopounidis. Building Additive Utilities in the Presence of Non-Monotonic Preferences. In *Advances in Multicriteria Analysis*,

Nonconvex Optimization and Its Applications, pages 101–114. Springer US, Boston, MA, 1995.

[32] J. M. Devaud, G. Groussaud, and E. Jacquet-Lagrèze. UTADIS: Une méthode de construction de fonctions d'utilité additives rendant compte de jugements globaux. *European Working Group on Multicriteria Decision Aid, Bochum*, 94, 1980.

[33] L. Dias, V. Mousseau, J. Figueira, and J. Clımaco. An aggregation/disaggregation approach to obtain robust conclusions with ELECTRE TRI. *European Journal of Operational Research*, 138(2):332–348, 2002.

[34] L. Dias and A. Tsoukiàs. On the constructive and other approaches in decision aiding. In *Eds, Proceedings of the 56th meeting of the EURO MCDA working group*, pages 13–28, 2004.

[35] M. Doumpos, Y. Marinakis, M. Marinaki, and C. Zopounidis. An evolutionary approach to construction of outranking models for multicriteria classification: The case of the ELECTRE TRI method. *European Journal of Operational Research*, 199(2):496–505, 2009.

[36] M. Doumpos and C. Zopounidis. *Multicriteria decision aid classification methods*, volume 73. Springer Science and Business Media, 2002.

[37] M. Doumpos and C. Zopounidis. On the development of an outranking relation for ordinal classification problems: An experimental investigation of a new methodology. *Optimization Methods and Software*, 17(2):293–317, 2002.

[38] W. Duivesteijn and A. Feelders. Nearest neighbour classification with monotonicity constraints. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 301–316. Springer, 2008.

[39] A. Eckhardt and T. Kliegr. Preprocessing algorithm for handling non-monotone attributes in the UTA method. In *Proceedings of the ECAI-12 Workshop on Preference Learning: Problems and Applications in AI (PL-12)*, page 28–32, 2012.

[40] P. Egan. "Do something" politics and double-peaked policy preferences. *The Journal of Politics*, 76(2):333–349, 2014.

[41] B. Escoffier, J. Lang, and M. Öztürk. Single-peaked consistency and its complexity. In *ECAI*, volume 8, page 366–370, 2008.

[42] A. Fallah Tehrani, W. Cheng, K. Dembczyński, and E. Hüllermeier. Learning monotone nonlinear models using the Choquet integral. *Machine Learning*, 89(1):183–211, 2012.

[43] V. Ferretti, L. Jinyan, V. Mousseau, and W. Ouerdane. Reference-based ranking procedure for environmental decision making: Insights from an ex-post analysis. *Environmental Modelling and Software*, 99:11–24, 2018.

[44] J. Figueira, V. Mousseau, and B. Roy. ELECTRE methods. In *Multiple criteria decision analysis: State of the art surveys*, pages 133–153. Springer, 2005.

[45] J. Figueira, T. Tervonen, J. Almeida-Dias, R. Lahdelma, and P. Salmiken. SMAA-TRI: a parameter stability analysis method for ELECTRE TRI. In *NATO advanced research workshop*, pages 20–24, 2004.

[46] A. Filos-Ratsikas, M. Li, J. Zhang, and Q. Zhang. Facility location with double-peaked preferences. *Autonomous Agents and Multi-Agent Systems*, 31(6):1209–1235, 2017.

[47] J. Fürnkranz and E. Hüllermeier. *Preference Learning*, pages 789–795. Springer US, Boston, MA, 2010.

[48] S. Greco, B. Matarazzo, and R. Slowinski. Rough sets theory for multicriteria decision analysis. *European Journal of Operational Research*, 129(1):1–47, 2001.

[49] S. Greco, B. Matarazzo, and R. Slowinski. Decision rule approach. *Multiple criteria decision analysis: state of the art surveys*, pages 507–562, 2005.

[50] S. Greco, B. Matarazzo, R. Slowinski, and J. Stefanowski. An algorithm for induction of decision rules consistent with the dominance principle. In *International Conference on Rough Sets and Current Trends in Computing*, pages 304–313. Springer, 2000.

[51] S. Greco, V. Mousseau, and R. Słowiński. Multiple criteria sorting with a set of additive value functions. *European Journal of Operational Research*, 207(3):1455–1470, 2010.

[52] J. W. Grzymala-Busse. LERS - a system for learning from examples based on rough sets. In *Intelligent decision support*, pages 3–18. Springer, 1992.

[53] M. Guo, X. Liao, and J. Liu. A progressive sorting approach for multiple criteria decision aiding in the presence of non-monotonic preferences. *Expert Systems with Applications*, 123:1–17, 2019.

[54] P. Gutiérrez and S. García. Current prospects on ordinal and monotonic classification. *Prog. Artif. Intell.*, 5(3):171–179, 2016.

[55] P. Gutiérrez, M. Pérez-Ortiz, J. Sánchez-Monedero, F. Fernandez-Navarro, and C. Martínez. Ordinal regression methods: survey and experimental study. *IEEE Transactions on Knowledge and Data Engineering*, 28(1):127–146, 2015.

[56] E. Jacquet-Lagreze and J. Siskos. Assessing a set of additive utility functions for multicriteria decision-making, the UTA method. *European Journal of Operational Research*, 10(2):151–164, 1982.

[57] E. Jacquet-Lagrèze and Y. Siskos. Preference disaggregation: 20 years of MCDA experience. *European Journal of Operational Research*, 130(2):233–245, 2001.

[58] M. Kadziński, K. Martyn, M. Cinelli, R. Słowiński, S. Corrente, and S. Greco. Preference disaggregation for multiple criteria sorting with partial monotonicity constraints: Application to exposure management of nanomaterials. *International Journal of Approximate Reasoning*, 117:60–80, 2020.

[59] R. L. Keeney and H. Raiffa. *Decisions with multiple objectives: preferences and value trade-offs*. Cambridge university press, 1993.

[60] T. Kliegr. UTA-NM: Explaining stated preferences with additive non-monotonic utility functions. *Preference Learning*, page 56, 2009.

[61] E. Kujawski, E. Triantaphyllou, and J. Yanase. Additive multicriteria decision analysis models: Misleading aids for life-critical shared decision making. *Medical Decision Making*, 39(4):437–449, 2019.

[62] M. A. Lazouni, M. A. Chikh, and S. Mahmoudi. A new computer aided diagnosis system for pre-anesthesia consultation. *Journal of Medical Imaging and Health Informatics*, 3(4):471–479, 2013.

[63] A. Leroy, V. Mousseau, and M. Pirlot. Learning the parameters of a multiple criteria sorting method. In *International Conference on Algorithmic Decision Theory*, pages 219–233. Springer, 2011.

[64] J. Liu, X. Liao, M. Kadziński, and R. Słowiński. Preference disaggregation within the regularization framework for sorting problems with multiple potentially non-monotonic criteria. *European Journal of Operational Research*, 276(3):1071–1089, 2019.

[65] D. Martin and M. Mazzotta. Non-monetary valuation using multi-criteria decision analysis: Sensitivity of additive aggregation methods to scaling and compensation assumptions. *Ecosystem Services*, 29:13–22, 2018.

[66] P. McCullagh. Regression models for ordinal data. *Journal of the Royal Statistical Society: Series B (Methodological)*, 42(2):109–127, 1980.

[67] P. Meyer and A. Olteanu. Integrating large positive and negative performance differences into multicriteria majority-rule sorting models. *Computers and Operations Research*, 81:216–230, 2017.

[68] P. Meyer and A. Olteanu. Handling imprecise and missing evaluations in multi-criteria majority-rule sorting. *Computers & Operations Research*, 110:135–147, 2019.

[69] P. Minoungou, V. Mousseau, W. Ouerdane, and P. Scotton. Learning an MR-sort model from data with latent criteria preference direction. In *DA2PL'2020, from multiple criteria Decision Aid to Preference Learning*, Trento, Italy, 2020.

[70] V. Mousseau, J. Figueira, and J. Naux. Using assignment examples to infer weights for ELECTRE TRI method: Some experimental results. *European Journal of Operational Research*, 130(2):263–275, 2001.

[71] V. Mousseau and R. Slowinski. Inferring an ELECTRE TRI model from assignment examples. *Journal of global optimization*, 12(2):157–174, 1998.

[72] O. Nefla, M. Öztürk, P. Viappiani, and I. Brigui-Chtioui. Interactive elicitation of a majority rule sorting model with maximum margin optimization. In *International Conference on Algorithmic DecisionTheory*, pages 141–157. Springer, 2019.

[73] A. Ngo The and V. Mousseau. Using assignment examples to infer category limits for the ELECTRE TRI method. *Journal of Multi-Criteria Decision Analysis*, 11(1):29–43, 2002.

[74] O. Özpeynirci, S. Özpeynirci, and V. Mousseau. An interactive approach for inverse multiple criteria sorting problem. *Journal of Multi-Criteria Decision Analysis*, 28(3-4):160–169, 2021.

[75] S. Özpeynirci, O. Özpeynirci, and V. Mousseau. An interactive algorithm for multiple criteria constrained sorting problem. *Annals of Operations Research*, 267(1):447–466, 2018.

[76] S. Pei and Q. Hu. Partially monotonic decision trees. *Information Sciences*, 424:104–117, 2018.

[77] M. Pereira, E. Dias, and D. Fontes. A MCDA model for olive oil supplier selection using MACBETH. *International journal for quality research*, 13(4):849–862, 2019.

[78] R. Potharst and J. Bioch. A decision tree algorithm for ordinal classification. In *International Symposium on Intelligent Data Analysis*, pages 187–198. Springer, 1999.

[79] R. Potharst and A. Feelders. Classification trees for problems with monotonicity constraints. *ACM SIGKDD Explorations Newsletter*, 4:1–10, 2002.

[80] B. Roy. *Méthodologie multicritère d'aide à la décision*. Economica, 1985.

[81] B. Roy, J. Figueira, and J. Dias. ELECTRE TRI-C: A multiple criteria sorting method based on characteristic reference actions. *European Journal of Operational Research*, 204:565–580, 2010.

[82] T. Saaty. The analytic hierarchy process (AHP) for decision making. In *Kobe, Japan*, pages 1–69, 1980.

[83] J. Simos. Évaluer l'impact sur l'environnement. une approche originale par l'analyse multicritère et la négotiation. *Géographie physique et Quaternaire*, 1990.

[84] J. Simos. *L'évaluation environnementale: Un processus cognitif négocié. Lausanne*. PhD thesis, Thèse, École Polytechnique Fédérale de Lausanne, 1990.

[85] Y. Siskos. Analyse de systèmes de décision multicritère en univers aléatoire. *Foundations of Control Engineering*, 8(3-4):193–212, 1983.

[86] Y. Siskos, E. Grigoroudis, and N. F. Matsatsinis. UTA Methods. In *Multiple Criteria Decision Analysis*, International Series in Operations Research and Management Science, pages 315–362. Springer, 2016.

[87] Y. Siskos and D. Yannacopoulos. UTASTAR: An ordinal regression method for building additive value functions. *Investigaçao Operacional*, 5(1):39–53, 1985.

[88] O. Sobrie. *Learning preferences with multiple-criteria models*. PhD thesis, Université de Mons (Faculté Polytechnique) and Université Paris-Saclay (CentraleSupélec), 2016.

[89] O. Sobrie, M. Lazouni, V. Mousseau, and M. Pirlot. A new decision support model for preanesthetic evaluation. *Computer Methods and Programs in Biomedicine*, 133:183–193, 2016.

[90] O. Sobrie, V. Mousseau, and M. Pirlot. Learning a Majority Rule Model from Large Sets of Assignment Examples. In *Algorithmic Decision Theory*, volume 8176, pages 336–350. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[91] O. Sobrie, V. Mousseau, and M. Pirlot. A population-based algorithm for learning a majority rule sorting model with coalitional veto. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 575–589. Springer, 2017.

[92] O. Sobrie, V. Mousseau, and M. Pirlot. Learning monotone preferences using a majority rule sorting model. *International Transactions in Operational Research*, 26(5):1786–1809, 2019.

[93] R. Sousa, I. Yevseyeva, J. Costa, and J. Cardoso. Multicriteria models for learning ordinal data: A literature review. *Artificial Intelligence, Evolutionary Computing and Metaheuristics*, pages 109–138, 2013.

[94] A. Tlili, K. Belahcène, O. Khaled, V. Mousseau, and W. Ouerdane. Learning non-compensatory sorting models using efficient SAT/MaxSAT formulations. *European Journal of Operational Research*, 298(3):979–1006, 2022.

[95] A. Tsoukiàs. On the concept of decision aiding process: an operational perspective. *Annals of Operations Research*, 154(1):3–27, 2007.

[96] G. Tsoumakas and I. Katakis. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining (IJDWM)*, 3(3):1–13, 2007.

[97] A. Valko, A. Olteanu, D. Brosset, and P. Meyer. Integrating a temporal component into multi-criteria majority-rule sorting models. In *DA2PL'2018: from Multiple Criteria Decision Aid to Preference Learning*, 2018.

[98] A. Valko, A. Olteanu, and P. Meyer. Integrating multiple contexts into multi-criteria majority-rule sorting. In *ADT 2019: 6th International Conference on Algorithmic Decision Theory*, pages 177–179. Springer, 2019.

[99] V. Vapnik. *Statistical learning theory*. Wiley, 1998.

[100] W. Verbeke, D. Martens, and B. Baesens. RULEM: A novel heuristic rule learning approach for ordinal classification with monotonicity constraints. *Applied Soft Computing*, 60:858–873, 2017.

[101] P. Vojtas and A. Eckhardt. Considering data-mining techniques in user preference learning. In *2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, volume 3, pages 33–36. IEEE, 2008.

[102] H. Wang, M. Zhou, and K. She. Induction of ordinal classification rules from decision tables with unknown monotonicity. *European Journal of Operational Research*, 242(1):172–181, 2015.

[103] J. Wątróbski, J. Jankowski, P. Ziemba, A. Karczmarczyk, and M. Zioło. Generalised framework for multi-criteria method selection. *Omega*, 86:107–124, 2019.

[104] W. Yu. *Aide multicritère à la décision dans le cadre de la problématique du tri: méthodes et applications.* PhD thesis, LAMSADE, Université Paris Dauphine, Paris, 1992.

université PARIS-SACLAY | ÉCOLE DOCTORALE Interfaces: matériaux, systèmes, usages

Titre: Apprentissage de modèles à règle majoritaire à partir de données partiellement monotones
Mots clés: ADMC, MR-Sort, préférences single-peaked, apprentissage de préférences

Résumé:

Le domaine de l'Aide à la Décision Multicritère (ADMC), s'interesse à évaluer des alternatives suivant des critères dans le but de recommander la "meilleure" solution au décideur. Dans ce contexte, nous considerons le paradigme d'apprentissage de préférences - comparable à l'approche en Machine Learning - qui consiste à déduire à partir des observations passées du décideur, les paramètres du modèle qui correspondent au mieux à ses préférences. Notre modèle (MR-Sort) est issu de la famille des modèles de surclassement, dans lequel une alternative a surclasse une autre alternative b s'il existe une forte coalition de critères (majorité) favorable au surclassement de a par rapport à b. Dans la littérature de l'ADMC, les méthodes et algorithmes étudiés pour les problèmes de tri - classification dans des catégories prédéfinies et ordonnées - ont toujours eu pour but l'inférence de modèles MR-Sort connaissant le sens de préférence des critères et à partir de préférences monotones (croissantes ou décroissantes). Dans cette thèse, nous étendons l'état de l'art à l'étude des préférences dites "single-peaked" (resp. "single-valley"), qui améliorent l'expressivité des modèles MR-Sort. Un critère single-peaked est caractérisé par deux monotonies successives (croissante puis décroissante). Ainsi, nous étudions des problèmes d'apprentissage des paramètres de MR-Sort à partir de préférences monotones et single-peaked, quelle que soit la connaissance à priori des sens de préférences des critères. Nous proposons une méthode exacte, des heuristiques et des tests pour évaluer et comparer nos algorithmes suivant le temps de calcul, le taux de classification et de restitution des sens de préférences.

Title: Learning Majority-Rule models with partially monotone data
Keywords: MCDA, MR-Sort, single-peaked preferences, preference learning

Abstract:

The field of Multiple Criteria Decision Analysis (MCDA) deals with alternatives evaluated by several criteria, aiming to recommend the "best" decision to the decision-maker (DM). In this context, we are interested in the indirect learning paradigm which is comparable to machine learning tasks as it consists of inferring from past observations of the DM, the model parameters that suit the DM's preferences. Our model (MR-Sort) stems from the MCDA family of outranking models, where an alternative a outranks another alternative b if there is a strong support of criteria (a majority in MR-Sort) that favors a compared to b. In the literature, methods and algorithms used for sorting problems - classification into predefined and ordered categories - always infer MR-Sort models with known criteria preference directions and monotone (increasing or decreasing) preferences. In this thesis, we extend the state-of-the-art to single-peaked (and single-valley) criteria which improves the expressivity of MR-Sort models. A single-peaked criterion relates to two successive monotonicities (increasing then decreasing). Therefore we investigate the problem of learning the MR-Sort parameters from monotone and single-peaked preferences regardless of the knowledge of preference directions of criteria. We propose an exact method and heuristics, and conducted experiments to assess and compare our algorithms regarding the computational cost, the classification accuracy and the preference directions retrieval.