Learning the Parameters of a Non Compensatory Sorting Model

Olivier Sobrie[1,2(✉)], Vincent Mousseau[1], and Marc Pirlot[2]

[1] LGI, CentraleSupelec, Grande Voie des Vignes, 92295 Châteney-Malabry, France
olivier.sobrie@gmail.com, vincent.mousseau@ecp.fr
[2] Faculté Polytechnique, Université de Mons, 9, Rue de Houdain, 7000 Mons, Belgium
marc.pirlot@umons.ac.be

Abstract. We consider a multicriteria sorting procedure based on a majority rule, called MR-Sort. This procedure allows to sort each object of a set, evaluated on multiple criteria, in a category selected among a set of pre-defined and ordered categories. With MR-Sort, the ordered categories are separated by profiles which are vectors of performances on the different attributes. Using the MR-Sort rule, an object is assigned to a category if it is at least as good as the category lower profile and not better than the category upper profile. To determine whether an object is as good as a profile, the weights of the criteria on which the object performances are better than the profile performances are summed up and compared to a threshold. If the sum of weights is at least equal to the threshold, then the object is considered at least as good as the profile. In view of increasing the expressiveness of the model, we substitute additive weights by a capacity to represent the power of coalitions of criteria. This corresponds to the Non-Compensatory Sorting model characterized by Bouyssou and Marchant. In the paper we describe a mixed integer program and a heuristic algorithm that enable to learn the parameters of this model from assignment examples.

1 Introduction

In Multiple Criteria Decision Analysis (MCDA), the "sorting problem setting" (or ordered classification) consists in assigning each alternative of a set, evaluated on several monotone criteria, in a category selected among a set of pre-defined and ordered categories. Several MCDA methods are designed to handle sorting problems. In this paper, we consider a sorting model that satisfies the requirements of the non-compensatory sorting models characterized in [1,2]. The model is a generalization of MR-Sort [3,4]. In MR-Sort, categories are separated by profiles which are vectors of performances on the different criteria. Each criterion of the model is associated a weight representing its importance or its voting power. Using this model, without veto, we assign an alternative to a category if it is considered at least as good as the category lower profile and not at least as good as the category upper profile. An alternative is considered as good as a profile if

T. Walsh (Ed.): ADT 2015, LNAI 9346, pp. 153–170, 2015.
DOI: 10.1007/978-3-319-23114-3_10

its performances are at least as good as the profile performances on a weighted majority of criteria. In MR-Sort, the weighted majority of criteria is reached if the sum of weights of criteria on which the alternative is at least as good as the profile is greater than a threshold.

Such a model contrasts with utility based models such as UTADIS [5,6]. It belongs to a class of decision models referred as noncompensatory in the literature [7,8], because it just takes into account whether or not an evaluation is above the profile value, not by how much it passes or misses this profile value. These methods are well suited to criteria assessed on ordinal scales.

Consider a MR-Sort model involving 4 criteria (c_1, c_2, c_3 and c_4) and 2 ordered categories ($C_2 \succ C_1$), separated by a profile b_1. Using this model, an alternative is assigned to the "good" category (C_2) iff its performances are as good as the profile b_1 on at least one of the four following minimal criteria coalitions: $c_1 \wedge c_2$, $c_3 \wedge c_4$, $c_1 \wedge c_4$ and $c_2 \wedge c_4$. A coalition of criteria is said to be minimal if removing any criterion is enough to reject the assertion "alternative a is as good as profile b". With an MR-Sort model, this can be achieved by selecting, for instance, the following weights and majority threshold: $w_1 = 0.3$, $w_2 = 0.2$, $w_3 = 0.1$, $w_4 = 0.4$ and $\lambda = 0.5$. We have $w_1 + w_2 = \lambda$, $w_3 + w_4 = \lambda$, $w_1 + w_4 > \lambda$ and $w_2 + w_4 > \lambda$. All the other coalitions of criteria, which are not supersets of the four minimal coalitions listed above, are not sufficient to be considered as good as b_1 (e.g. $w_1 + w_3 < \lambda$).

Assume that we want a model for which the two minimal sufficient criteria coalitions are: $c_1 \wedge c_2$ and $c_3 \wedge c_4$. Modeling this classification rule with an MR-Sort model is impossible. To model these rules, we have to choose weights w_i, $i = 1, \ldots, 4$, summing up to 1, such that $w_1 + w_2 \geq \lambda$ and $w_3 + w_4 \geq \lambda$. Summing these two inequalities yields $1 \geq 2\lambda$. If we want these coalitions to be the only minimal sufficient ones, we must also have : $w_1 + w_3 < \lambda$, $w_1 + w_4 < \lambda$, $w_2 + w_3 < \lambda$ and $w_2 + w_4 < \lambda$. Summing these four inequalities yields $2 < 4\lambda$. Hence, there exist no weights and majority threshold for which the 2 above coalitions are the only two minimal sufficient coalitions. In view of being able to represent such a type of rule, we consider in this paper an extension of MR-Sort allowing to model interactions between criteria. This formulation expresses the majority rule of MR-Sort by using a capacity like in the Choquet Integral [9]. This model is called the Non Compensatory Sorting Model (NCS model). It was introduced and characterized in [1,2].

In this paper, we aim at studying the additional descriptive ability of the NCS model as compared to MR-Sort. We assess this experimentally on real datasets. The paper is organized as follows. The next section describes formally what is a non compensatory sorting model. Section 3 recalls previous work dealing with learning the parameters of MR-Sort models from assignment examples. The next two sections describe respectively a Mixed Integer Program and a heuristic algorithm that allow to learn the parameters of a NCS model. Some experimental results are finally presented.

2 MR-Sort and NCS Models

2.1 MR-Sort Model

MR-Sort is a method for assigning objects to ordered categories. It is a simplified version of ELECTRE TRI, another MCDA method [10,11].

The MR-Sort rule works as follows. Formally, let X be a set of objects evaluated on n ordered attributes (or criteria), $F = \{1, ..., n\}$. We assume that X is the Cartesian product of the criteria scales, $X = \prod_{j=1}^{n} X_j$. An object $a \in X$ is a vector $(a_1, \ldots, a_j, \ldots, a_n)$, where $a_j \in X_j$ for all j. The ordered categories which the objects are assigned to by the MR-Sort model are denoted by C_h, with $h = 1, \ldots, p$. Category C_h is delimited by its lower limit profile b_{h-1} and its upper limit profile b_h, which is also the lower limit profile of category C_{h+1} (provided $0 < h < p$). The profile b_h is the vector of criterion values $(b_{h,1}, \ldots, b_{h,j}, \ldots, b_{h,n})$, with $b_{h,j} \in X_j$ for all j. We denote by $P = \{1, ..., p\}$ the list of category indices. By convention, the best category, C_p, is delimited by a fictive upper profile, b_p, and the worst one, C_1, by a fictive lower profile, b_0. It is assumed that the profiles dominate one another, i.e.: $b_{h-1,j} \leq b_{h,j}$, for $h = \{1, \ldots, p\}$ and $j = \{1, \ldots, n\}$.

Using the MR-Sort procedure, an object is assigned to a category if its criterion values are at least as good as the category lower profile values on a weighted majority of criteria while this condition is not fulfilled when the object's criterion values are compared to the category upper profile values. In the former case, we say that the object is *preferred* to the profile, while, in the latter, it is not. Formally, if an object $a \in X$ is *preferred* to a profile b_h, we denote this by $a \succcurlyeq b_h$. Object a is preferred to profile b_h whenever the following condition is met:

$$a \succcurlyeq b_h \Leftrightarrow \sum_{j:a_j \geq b_{h,j}} w_j \geq \lambda, \tag{1}$$

where w_j is the nonnegative weight associated with criterion j, for all j and λ sets a majority level. The weights satisfy the normalization condition $\sum_{j \in F} w_j = 1$; λ is called the *majority threshold*; it satisfies $\lambda \in [1/2, 1]$.

The preference relation $\succcurlyeq$ defined by (1) is called an *outranking* relation without veto or a *concordance* relation ([11]; see also [12,13] for an axiomatic description of such relations). Consequently, the condition for an object $a \in X$ to be assigned to category C_h reads:

$$\sum_{j:a_j \geq b_{h-1,j}} w_j \geq \lambda \quad \text{and} \quad \sum_{j:a_j \geq b_{h,j}} w_j < \lambda. \tag{2}$$

The MR-Sort assignment rule described above involves $pn + 1$ parameters, i.e. n weights, $(p-1)n$ profiles evaluations and one majority threshold.

A *learning set* A is a subset of objects $A \subseteq X$ for which an assignment is known. For $h \in P$, A_h denotes the subset of objects $a \in A$ which are assigned to category C_h. The subsets A_h are disjoint; some of them may be empty.

2.2 NCS Model

Limitation of MR-Sort. Before describing the NCS model, we show the limits of MR-Sort. As an illustration, consider an application in which a committee for a higher education program has to decide about the admission of students on the basis of their evaluations in 4 courses: math, physics, chemistry and history. To be accepted in the program, the committee considers that a student should have a sufficient majority of evaluations above 10/20. From the committee point of view, courses (criteria) coalitions don't have the same importance. The strength of a coalition of courses varies as a function of the courses belonging to the coalition. The committee stated that the following subsets are the minimal coalitions of courses in which the evaluation should be above 10/20 in order to be accepted: {math, physics}, {math, chemistry} and {chemistry, history}. To illustrate this rule, Table 1 shows evaluations of several students and, for each student, whether he is accepted or refused.

Representing these assignments by using a MR-Sort model with profile fixed at 10/20 in each course is impossible. There are no additive weights allowing to model such rules. MR-Sort is not adapted to handle such type of problems since it does not allow to model attribute interactions. In view of taking criterion interactions into account, we modify the definition of the global outranking relation, $a \succcurlyeq b_h$, given in (1).

Capacity. The new model described hereafter uses capacities. A capacity is a function $\mu : 2^F \to [0, 1]$ such that:

- $\mu(B) \geq \mu(A)$, for all $A \subseteq B \subseteq F$ (monotonicity);
- $\mu(\emptyset) = 0$ and $\mu(F) = 1$ (normalization).

The Möbius transform allows to express the capacity in another form:

$$\mu(A) = \sum_{B \subseteq A} m(B) \quad \forall A \subseteq F \quad \text{with } m(B) = \sum_{C \subseteq B} (-1)^{|B|-|C|} \mu(C).$$

The value $m(B)$ can be interpreted as the weight that is exclusively allocated to B as a whole. A capacity can be defined directly by its Möbius transform

Table 1. Evaluation of students and their acceptance/refusal status

	Math	Physics	Chemistry	History	A/R
James	11	11	9	9	A
Marc	11	9	11	9	A
Robert	9	9	11	11	A
John	11	9	9	11	R
Paul	9	11	9	11	R
Pierre	9	11	11	9	R

also called Möbius interaction. A Möbius interaction or Möbius mass m is a set function $m : 2^F \rightarrow [-1, 1]$ satisfying the following conditions:

$$\sum_{j \in K \subseteq J \cup \{j\}} m(K) \geq 0 \quad \forall j \in F, J \subseteq F \backslash \{i\} \quad \text{and} \quad \sum_{K \subseteq F} m(K) = 1. \tag{3}$$

If m is a Möbius interaction, the set function defined by $\mu(A) = \sum_{B \subseteq A} m(B)$ is a capacity. Conditions (3) guarantee that μ is monotone [14].

NCS Model. Using a capacity to express the weight of the coalition in favor of an object, we transform the outranking rule (1) as follows:

$$\begin{aligned} a \succcurlyeq b_h \Leftrightarrow \mu(A) \geq \lambda \text{ with } A = \{j \in F : a_j \geq b_{h,j}\} \\ \text{and } \mu(A) = \sum_{B \subseteq A} m(B) \end{aligned} \tag{4}$$

Computing the value of $\mu(A)$ with the Möbius transform requires the evaluation of $2^{|A|}$ parameters. In a model involving n criteria, this implies the elicitation of 2^n parameters, with $\mu(\emptyset) = 0$ and $\mu(F) = 1$. To reduce the number of parameters to elicit, we use a 2-additive capacity in which all the interactions involving more than 2 criteria are equal to zero. Inferring a 2-additive capacity for a model having n criteria requires the determination of $\frac{n(n+1)}{2} - 1$ parameters.

Finally, the condition for an object $a \in X$ to be assigned to category C_h can be expressed as follows:

$$\mu(F_{a \geq b_{h-1}}) \geq \lambda \quad \text{and} \quad \mu(F_{a \geq b_h}) < \lambda \tag{5}$$

with $F_{a \geq b_{h-1}} = \{j \in F : a_j \geq b_{h-1,j}\}$ and $F_{a \geq b_h} = \{j \in F : a_j \geq b_{h,j}\}$.

This model fits with the definition of a NCS model given in [1,2]. We note that MR-Sort is a special case of a NCS model in which a simple additive capacity is used.

3 Learning the Parameters of a MR-Sort Model

Learning the parameters of MR-Sort and ELECTRE TRI models has been already studied in several articles [3,4,15–21]. In this section, we recall how to learn the parameters of an MR-Sort model using respectively an exact method [3] and a heuristic algorithm [4].

3.1 Mixed Integer Programming

Learning the parameters of a MR-Sort model using linear programming techniques has been proposed in [3]. The paper describes a Mixed Integer Program (MIP) taking a set of assignment examples and their vector of performances as input and finding the parameters of a MR-Sort model such that the largest

possible number of examples are restored by the inferred model. We recall in this subsection the main steps to obtain the MIP formulation.

The condition for an object x to be assigned to category C_h (Equation (2)) can be written as follows:

$$a \in C_h \iff \begin{cases} \sum_{j=1}^{n} c_{a,j}^{h-1} \geq \lambda & \text{with } c_{a,j}^{h-1} = \begin{cases} w_j & \text{if } a_j \geq b_{h-1,j} \\ 0 & \text{otherwise} \end{cases} \\ \sum_{j=1}^{n} c_{a,j}^{h} < \lambda & \text{with } c_{a,j}^{h} = \begin{cases} w_j & \text{if } a_j \geq b_{h,j} \\ 0 & \text{otherwise} \end{cases} \end{cases}$$

To linearize these constraints, we introduce for each value $c_{a,j}^{l}$, with $l = \{h-1, h\}$, a binary variable $\delta_{a,j}^{l}$ that is equal to 1 when the performance of the object a is at least as good as or better than the performance of the profile b_l on criterion j and 0 otherwise. To obtain the value of $\delta_{a,j}^{l}$, we add the following constraints, where M is an arbitrary large positive constant:

$$M(\delta_{a,j}^{l} - 1) \leq a_j - b_{l,j} < M \cdot \delta_{a,j}^{l} \tag{6}$$

By using the value $\delta_{a,j}^{l}$, the values of $c_{a,j}^{l}$ are obtained as follows:

$$\begin{cases} c_{a,j}^{l} & \geq 0 \\ c_{a,j}^{l} & \leq w_j \end{cases} \qquad \begin{cases} c_{a,j}^{l} & \leq \delta_{a,j}^{l} \\ c_{a,j}^{l} & \geq \delta_{a,j}^{l} - 1 + w_j \end{cases}$$

The objective function of the MIP consists in maximizing the number of examples compatible with the learned model, i.e. minimizing the 0/1 loss function. In order to model this, new binary variables, γ_a for all $a \in A$, are introduced. The value of γ_a is equal to 1 if object a is assigned to the expected category, i.e. the category it is assigned to in the learning set, and equal to 0 otherwise. To obtain the correct value of γ_a variables, two additional constraints are added:

$$\begin{cases} \sum_{j=1}^{n} c_{a,j}^{h-1} & \geq \lambda + M(\gamma_a - 1) \\ \sum_{j=1}^{n} c_{a,j}^{h} & < \lambda - M(\gamma_a - 1) \end{cases}$$

The objective function chosen for the linear program consists in maximizing the number of examples compatible with the model. Formally it reads: $\max \sum_{a \in A} \gamma_a$. Finally, the combination of all the constraints leads to the MIP given in Appendix A.

3.2 A Heuristic Algorithm

The MIP presented in the previous section is not suitable for large data sets because of the high computing time that is required to infer the MR-Sort parameters. In view of learning MR-Sort models in the context of large data sets, a heuristic algorithm has been proposed in [4]. As in the MIP, the heuristic algorithm takes as input a set of assignment examples and their vectors of performances. The algorithm returns the parameters of a MR-Sort model.

$$
\begin{aligned}
&\min \textstyle\sum_{a \in A}(x'_a + y'_a) \\
&\text{s.t.} \\
&\sum_{j:a_j \geq b_{h-1,j}} w_j - x_a + x'_a = \lambda && \forall a \in A_h, h = \{2, ..., p\} \\
&\sum_{j:a_j \geq b_{h,j}} w_j + y_a - y'_a = \lambda - \epsilon && \forall a \in A_h, h = \{1, ..., p-1\} \\
&\sum_{j=1}^{n} w_j = 1 \\
&w_j \in [0; 1] && \forall j \in F \\
&\lambda \in [0.5; 1] \\
&x_a, y_a, x'_a, y'_a \in \mathbb{R}_0^+ \\
&\varepsilon \text{ a small positive number.}
\end{aligned}
\tag{7}
$$

The heuristic algorithm proposed in [4] works as follows. First a population of MR-Sort models is initialized. After the initialization, the two following steps are repeated iteratively on each model in the population:

1. A linear program optimizes the weights and the majority threshold on the basis of assignment examples and fixed profiles.
2. Given the inferred weights and the majority threshold, a heuristic adjusts the profiles of the model on the basis of the assignment examples.

After applying these two steps to all the models in the population, the $\lfloor \frac{n}{2} \rfloor$ models restoring the least numbers of examples are reinitialized. These steps are repeated until the heuristic finds a model that fully restores all the examples or after a number of iterations specified a priori.

The linear program designed to learn the weights and the majority threshold is given by (7). It minimizes a sum of slack variables, x'_a and y'_a, that is equal to 0 when all the objects are correctly assigned, i.e. assigned to the category defined in the input data set. We remark that the objective function of the linear program does not explicitly minimize the 0/1 loss but a sum of slacks. This implies that compensatory effects might appear, with undesirable consequences on the 0/1 loss. However in this heuristic, we consider that these effects are acceptable. The linear program doesn't involve binary variables. Therefore, the computing time remains reasonable when the size of the problem increases.

The objective function of the heuristic varying the profiles maximizes the number of examples compatible with the model. To do so, it iterates over each profile h and each criterion j and identifies a set of candidate moves for the profile, which correspond to the performances of the examples on criterion j located between profiles $h-1$ and $h+1$. Each candidate move is evaluated as a function of the probability to improve the classification accuracy of the model. To evaluate if a candidate move is likely or unlikely to improve the classification of one or several objects, the examples which have an evaluation on criterion

j located between the current value of the profile, $b_{h,j}$, and the candidate move, $b_{h,j} + \delta$ (resp. $b_{h,j} - \delta$), are grouped in different subsets:

- $V_{h,j}^{+\delta}$ (**resp.** $V_{h,j}^{-\delta}$): the sets of objects misclassified in C_{h+1} instead of C_h (resp. C_h instead of C_{h+1}), for which moving the profile b_h by $+\delta$ (resp. $-\delta$) on j results in a correct assignment.
- $W_{h,j}^{+\delta}$ (**resp.** $W_{h,j}^{-\delta}$): the sets of objects misclassified in C_{h+1} instead of C_h (resp. C_h instead of C_{h+1}), for which moving the profile b_h by $+\delta$ (resp. $-\delta$) on j strengthens the criteria coalition in favor of the correct classification but will not by itself result in a correct assignment.
- $Q_{h,j}^{+\delta}$ (**resp.** $Q_{h,j}^{-\delta}$): the sets of objects correctly classified in C_{h+1} (resp. C_{h+1}) for which moving the profile b_h by $+\delta$ (resp. $-\delta$) on j results in a misclassification.
- $R_{h,j}^{+\delta}$ (**resp.** $R_{h,j}^{-\delta}$): the sets of objects misclassified in C_{h+1} instead of C_h (resp. C_h instead of C_{h+1}), for which moving the profile b_h by $+\delta$ (resp. $-\delta$) on j weakens the criteria coalition in favor of the correct classification but does not induce misclassification by itself.
- $T_{h,j}^{+\delta}$ (**resp.** $T_{h,j}^{-\delta}$): the sets of objects misclassified in a category higher than C_h (resp. in a category lower than C_{h+1}) for which the current profile evaluation weakens the criteria coalition in favor of the correct classification.

A formal definition of these sets can be found in [4]. The evaluation of the candidate moves is done by aggregating the number of elements in each subset. Finally, the choice to move or not the profile on the criterion is determined by comparing the candidate move evaluation to a random number drawn uniformly. These operations are repeated multiple times on each profile and each criterion.

4 Mixed Integer Program to Learn a 2-Additive NCS Model

As compared to a MR-Sort model, a NCS model involves more parameters. In a standard MR-Sort model, a weight is associated to each criterion, which makes overall n parameters to elicit. With a NCS model limited to two-additive capacities, the computation of the strength of a coalition of criteria involves the weights of the criteria in the coalition and the pairwise interactions (Möbius coefficients) between these criteria. Overall there are $\frac{n(n+1)}{2} - 1$ coefficients. In the two-additive case, let us denote by m_j the weights of criterion j and by $m_{j,k}$ the Möbius interactions between criteria j and k. The capacity $\mu(A)$ of a subset of criteria is obtained as: $\mu(A) = \sum_{j \in A} m_j + \sum_{\{j,k\} \subseteq A} m_{j,k}$. The constraints (3) on the interaction read:

$$m_j + \sum_{k \in J} m_{j,k} \geq 0 \qquad \forall j \in F, \forall J \subseteq F \backslash \{j\} \tag{8}$$

and $\sum_{j \in F} m_j + \sum_{\{j,k\} \subseteq F} m_{j,k} = 1$.

The number of monotonicity constraints evolves exponentially as a function of the number of criteria, n. In [22], two other formulations are proposed in order to reduce significantly the number of constraints ensuring the monotonicity of the capacities. The first formulation reduces the number of constraints to $2n^2$ but leads to a non linear program. The second formulation reduces the number of constraints to $n^2 + 1$ without introducing non linearities but adds n^2 extra variables.

With a 2-additive MR-Sort model, the constraints for an alternative a to be assigned to a category h (5) can also be expressed as follows:

$$\begin{cases} \sum_{j=1}^{n} c_{a,j}^{h-1} + \sum_{j=1}^{n} \sum_{k=1}^{j} c_{a,j,k}^{h-1} & \geq \lambda + M(\gamma_a - 1) \\ \sum_{j=1}^{n} c_{a,j}^{h} + \sum_{j=1}^{n} \sum_{k=1}^{j} c_{a,j,k}^{h} & < \lambda - M(\gamma_a - 1) \end{cases} \tag{9}$$

with:

- $c_{a,j}^{h-1}$ (resp. $c_{a,j}^{h}$) equals m_j if the performance of alternative a is at least as good as the performance of profile b_{h-1} (resp. b_h) on criterion j, and equals 0 otherwise;
- $c_{a,j,k}^{h-1}$ (resp. $c_{a,j,k}^{h}$) equals $m_{j,k}$ if the performance of alternative a is at least as good as the performance of profile b_{h-1} (resp. b_h) on criteria j and k, and equals 0 otherwise.

For all $a \in A$, $j \in F$ and $l \in P$, constraints (8) imply that $c_{a,j}^{l} \geq 0$ and that $c_{a,j,k}^{l} \in [-1, 1]$. The values of $c_{a,j}^{h-1}$ and $c_{a,j}^{h}$ are obtained in a similar way as it is done for learning the parameters of a standard MR-Sort model by replacing the weights with the corresponding Möbius coefficients (10).

$$\begin{cases} c_{a,j}^{l} & \geq 0 \\ c_{a,j}^{l} & \leq m_j \end{cases} \qquad \begin{cases} c_{a,j}^{l} & \leq \delta_{a,j}^{l} \\ c_{a,j}^{l} & \geq \delta_{a,j}^{l} - 1 + m_j \end{cases} \tag{10}$$

However it is not the case for the variables $c_{a,j,k}^{h-1}$ and $c_{a,j,k}^{h}$, because they involve two criteria. To linearize the formulation, we introduce new binary variables, $\Delta_{a,j,k}^{l}$ equal to 1 if alternative a has better performances than profile b_l on criteria j and k and equal to 0 otherwise. We obtain the value of $\Delta_{a,j,k}^{l}$ thanks to the conjunction of constraints given in (6) and the following constraints:

$$2\Delta_{a,j,k}^{l} \leq \delta_{a,j}^{l} + \delta_{a,j}^{k} \leq \Delta_{a,j,k}^{l} + 1$$

In order to obtain the value of $c_{a,j,k}^{l}$, which can be either positive or negative, for all $l \in P$, we decompose the variable in two parts, $\alpha_{a,j,k}^{l}$ and $\beta_{a,j,k}^{l}$ such that $c_{a,j,k}^{l} = \alpha_{a,j,k}^{l} - \beta_{a,j,k}^{l}$ with $\alpha_{a,j,k}^{l} \geq 0$ and $\beta_{a,j,k}^{l} \geq 0$. The same is done for $m_{j,k}$ which is decomposed as follows: $m_{j,k} = m_{j,k}^{+} - m_{j,k}^{-}$ with $m_{j,k}^{+} \geq 0$ and $m_{j,k}^{-} \geq 0$. The values of $\alpha_{a,j,k}^{l}$ and $\beta_{a,j,k}^{l}$ are obtained thanks to the following constraints:

$$\begin{cases} \alpha_{a,j,k}^{l} & \leq \Delta_{a,j,k}^{l} \\ \alpha_{a,j,k}^{l} & \leq m_{j,k}^{+} \\ \alpha_{a,j,k}^{l} & \geq \Delta_{a,j,k}^{l} - 1 + m_{j,k}^{+} \end{cases} \qquad \begin{cases} \beta_{a,j,k}^{l} & \leq \Delta_{a,j,k}^{l} \\ \beta_{a,j,k}^{l} & \leq m_{j,k}^{-} \\ \beta_{a,j,k}^{l} & \geq \Delta_{a,j,k}^{l} - 1 + m_{j,k}^{-} \end{cases}$$

Finally, we obtain the MIP displayed in Appendix B.

$$
\begin{aligned}
&\min \textstyle\sum_{a\in A}(x'_a + y'_a)\\
&\text{s.t.}\\
&\sum_{j:a_j \geq b_{h-1,j}}^{n} \left(m_j + \sum_{k:a_k \geq b_{h-1,k}}^{j} m_{j,k} \right) - x_a + x'_a = \lambda \qquad \forall a \in A_h, \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad h = \{2, ..., p\}\\
&\sum_{j:a_j \geq b_{h,j}}^{n} \left(m_j + \sum_{k:a_k \geq b_{h,k}}^{j} m_{j,k} \right) + y_a - y'_a = \lambda - \varepsilon \; \forall a \in A_h, \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad h = \{1, ..., p-1\}\\
&\sum_{j=1}^{n} m_j + \sum_{j=1}^{n}\sum_{k=1}^{j} m_{j,k} = 1\\
&\qquad m_j + \sum_{k\in J} m_{j,k} \geq 0 \qquad\qquad\qquad\qquad \forall j \in F, \forall J \subseteq F \backslash \{j\}\\
&\qquad\qquad \lambda \in [0.5; 1]\\
&\qquad\qquad m_j \in [0, 1] \qquad\qquad\qquad\qquad \forall j \in F\\
&\qquad\qquad m_{j,k} \in [-1, 1] \qquad\qquad\qquad\qquad \forall j \in F, \forall k \in F, k < j\\
&\qquad x_a, y_a, x'_a, y'_a \in \mathbb{R}_0^+ \qquad\qquad\qquad\qquad a \in A\\
&\qquad\qquad \varepsilon \text{ a small positive number.}
\end{aligned}
\tag{11}
$$

5 A Heuristic Algorithm to Learn a 2-Additive NCS Model

The MIP described in the previous section requires a lot of binary variables and is therefore not well-suited for large problems. In the present section, we describe an adaptation of the heuristic described in Subsect. 3.2 in view of learning the parameters of a NCS model. Like for the MIP in the previous section, we limit the model to 2-additive capacities in order to reduce the number of coefficients as compared to a model with a general capacity.

One of the components that needs to be adapted in the heuristic in order to be able to learn a 2-additive NCS model is the linear program that infers the weights and the majority threshold (7). Like in the MIP described in the previous section, we use the Möbius transform to express capacities. In view of inferring Möbius coefficients, m_j and $m_{j,k}$, $\forall j, \forall k$ with $k < j$, we modify the linear program as shown in (11).

The value of $x_a - x'_a$ (resp. $y_a - y'_a$) represents the difference between the capacity of the criteria belonging to the coalition in favor of $a \in A_h$ w.r.t. b_{h-1} (resp. b_h) and the majority threshold. If both $x_a - x'_a$ and $y_a - y'_a$ are positive, then object a is assigned to the correct category. In order to try to maximize the number of examples correctly assigned by the model, the objective function of

the linear program minimizes the sum of x'_a and y'_a, i.e. the objective function is $\min \sum_{a \in A}(x'_a + y'_a)$.

The heuristic adjusting the profile also needs some adaptations in view of taking capacities into account. More precisely, the formal definition of the sets in which objects are classified for computing the candidate move evaluation should be adapted. The semantics of the sets, recalled in Sect. 3.2 remains identical. The formal definitions of these sets have to be adapted to take into account the capacity. The rest of the algorithm remains unchanged.

6 Experiments

The use of the MIP for learning a NCS model is limited because of the large number of binary variables involved. It contains more binary variables than the MIP learning the parameters of a simple additive MR-Sort model. Experiments reported in [3] have demonstrated that the computing time required to learn the parameters of a standard MR-Sort model having a small number of criteria and categories from a small set of assignment examples becomes quickly prohibitive. Therefore we cannot expect to be able to treat large problems using the MIP for learning NCS models.

In view of assessing the performance of the heuristic algorithm designed for learning the parameters of a NCS model, we use it to learn NCS models from several real data sets presented in Table 2. These data sets, available at http://www.uni-marburg.de/fb12/kebi/research/repository/monodata, have been already used to assess other algorithms (e.g. [4,23]). They involve from 120 to 1728 instances, from 4 to 8 monotone attributes and from 2 to 36 categories. In our experiments, categories have been binarized by thresholding at the median.

In our first experiment, we use 50 % of the alternatives in the data sets as learning set and the rest as test set. We learn MR-Sort and NCS models using both heuristics. We repeat this procedure for 100 random splits of the data sets

Table 2. Data sets

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 3. Average and standard deviation of the classification accuracy of the test set when using 50 % of the examples as learning set and the rest as test set

Data set	Heuristic MR-Sort	Heuristic NCS
DBS	0.8377 ± 0.0469	0.8312 ± 0.0502
CPU	0.9325 ± 0.0237	0.9313 ± 0.0272
BCC	0.7250 ± 0.0379	0.7328 ± 0.0345
MPG	0.8219 ± 0.0237	0.8180 ± 0.0247
ESL	0.8996 ± 0.0185	0.8970 ± 0.0173
MMG	0.8268 ± 0.0151	0.8335 ± 0.0138
ERA	0.7944 ± 0.0173	0.7944 ± 0.0156
LEV	0.8408 ± 0.0122	0.8508 ± 0.0188
CEV	0.9064 ± 0.0119	0.9118 ± 0.0263

Table 4. Average and standard deviation of the classification accuracy of the learning set when using the MR-Sort and NCS models learned on the whole data set

Data set	Heuristic MR-Sort	Heuristic NCS
DBS	0.9268 ± 0.0096	0.9326 ± 0.0087
CPU	0.9643 ± 0.0048	0.9703 ± 0.0091
BCC	0.7605 ± 0.0147	0.7761 ± 0.0085
MPG	0.8419 ± 0.0099	0.8389 ± 0.0069
ESL	0.9164 ± 0.0033	0.9168 ± 0.0042
MMG	0.8419 ± 0.0099	0.8409 ± 0.0091
ERA	0.8035 ± 0.0052	0.8027 ± 0.0053
LEV	0.8501 ± 0.0082	0.8643 ± 0.0038
CEV	0.9005 ± 0.0141	0.9172 ± 0.0101

in learning and test sets. We observe from Table 3 that the classification accuracy obtained with the NCS heuristic is on average comparable to the one obtained with the MR-Sort heuristic. The use of a more expressive model does not help much to improve the classification accuracy of the test set.

In a second experiment, we check the ability of MR-Sort and NCS to restore the whole data set. To do so, we run both heuristics 100 times. The average classification accuracy and standard deviation of the learning set are given in Table 4. The NCS heuristic does not always give better results than the MR-Sort one in restoring the learning set examples. Except for the MPG data set, we observe a slight advantage (of the order of one standard deviation) in favor of NCS when the number of attributes is at least 6. There is almost no difference for the data sets described by 4 or 5 attributes and for MPG (7 attributes).

Table 5. Average computing time (in seconds) required to find a solution with MR-Sort and NCS heuristics when using all the examples as learning set

Data set	Heuristic MR-Sort	Heuristic NCS
DBS	3.0508	6.9547
CPU	3.1646	5.2069
BCC	3.3700	7.7545
MPG	4.4136	9.9294
ESL	3.8466	7.2495
MMG	6.1481	13.4848
ERA	5.9689	14.4875
LEV	5.8986	13.2356
CEV	11.1122	31.7042

Average computing times of the results in Table 4 are displayed in Table 5. Learning a NCS model can take up to almost 3 times as much as learning a simple MR-Sort model.

The above experiments on benchmark data sets available in the literature failed to show a clear advantage at using NCS rather than MR-Sort. This raises the following question. Which type of data set would reveal a gain of expressivity provided by NCS over MR-Sort? We investigate this question in the next section.

7 Potential Gain in Descriptive Power with the NCS Model

Among NCS assignment rules, some can be exactly represented by additive weights and a threshold (the MR-Sort rules), while the others require a non-additive capacity and a threshold. We call the latter *non-additive* NCS rules. These are not MR-Sort rules but they can be *approximated* by a MR-Sort model. The experiment described below aims at assessing how well a non-additive NCS rule can be approximated by a MR-Sort rule.

Consider a NCS model assigning alternatives in two categories, C_1 and C_2. For a given profile, the set of all possible alternatives can be partitioned in 2^n subsets, where n is the number of criteria. Each of these subsets is characterized by one of the 2^n relative positions of an alternative w.r.t. the profile. On each criterion, the performance of an alternative is either at least as good as the profile or worse. Due to the ordinal nature of the NCS rule, all alternatives that share the same relative position w.r.t. the profile (i.e. all alternatives in the same class of the partition in 2^n subsets) are assigned to the same category. If we assume that the evaluations of the alternatives on all criteria range in the $[0, 1]$ interval, we can set the profile values to 0.5 on all criteria. The set of n-dimensional Boolean vectors is composed of exactly one example of each possible relative position w.r.t. the profile.

Our experiments are conducted as follows.

1. We modify the MIP described in Sect. 3.1 to learn only the weights and the majority threshold of a MR-Sort model on the basis of fixed profiles and assignment examples. The objective function of the MIP remains the minimization of the 0/1 loss.
2. We generate all possible NCS rules for $n = 4, 5, 6$ criteria. For more detail about how this can be done, see [24]; the list of all non-equivalent NCS rules is available at http://olivier.sobrie.be/shared/mbfs/. Each non-additive NCS rule, is used to assign the set of n-dimensional Boolean vectors to one of the two categories (using the 0.5 constant profile). These sets of representative alternatives constitute our learning sets.
3. The modified MIP is used to learn the weights and majority threshold of a MR-Sort model, which restores as well as possible the assignments made by the non-additive NCS rule.

The results of the experimentation are displayed in Table 6. Each row of the table contains the results for a given number of criteria, $n = 4, 5, 6$. The second column shows the percentage of non-additive NCS rules among all possible rules for each given number of criteria. The last three columns contain the min, max and average percentage of the 2^n examples assigned by non-additive rules that cannot be restored by a simple additive model.

Table 6. Average, minimum and maximum 0/1 loss of the learning sets after learning additive weights and the majority threshold of a MR-Sort model

n	% Non-additive	MR-Sort		
		Min.	Max.	Avg.
4	11 %	6.2 %	6.2 %	6.2 %
5	57 %	3.1 %	9.4 %	3.9 %
6	97 %	1.6 %	12.5 %	4.8 %

We observe that a MR-Sort model on 4 criteria is, in the worst case, not able to restore 6.2 % of the examples in the learning set (1 example out of 16). With 5 and 6 criteria, the maximum 0/1 loss increases respectively to 9.4 % (3 examples out of 32) and 12.5 % (8 examples out of 64).

Note that these proportions were obtained using learning sets in which each type of relative position w.r.t. the profile is represented exactly once. Therefore these conclusions should be valid for learning sets in which all types of relative positions are approximately equally represented. On a test set, the difference in classification performance between a non-additive NCS rule and its approximation by a MR-Sort rule can be amplified, or, on the contrary, can fade, depending on the proportion of the test alternatives belonging to the various types of relative positions w.r.t. the profile.

Table 6 reveals another important information. The proportion of non-additive NCS rules among all NCS rules quickly grows with the number of attributes: from 11 % of 2-additive NCS rules for $n = 4$ to 97 % for $n = 6$. It hence becomes more and more likely that a NCS rule is not a MR-Sort one when n grows.

The results in Table 6 could help to better understand the relatively poor gains observed in the previous section when comparing the heuristic algorithm for learning a 2-additive NCS model and a MR-Sort model. We noticed that the classification accuracy of the learned NCS rule tended to be slightly better for the data sets involving at least 6 attributes. The lack of an advantage for data sets involving 4 attributes might be due to the relative scarcity of non-additive NCS rules for $n = 4$ (11 %). When a gain is obtained, it is tiny, which might result from the fact that the approximation of a non-additive NCS rule by a MR-Sort rule is relatively good, at least up to $n = 6$. Investigating the NCS for $n \geq 7$ model in a systematic way, using the same method as we did in our last experiments, is almost impossible due to the extremely fast growth of the number of possible NCS rules (see [24]). It is however arguable that non-additive NCS rules could be at an advantage, as compared to MR-Sort rules, when the number of attributes is at least as large as 6.

A MIP Learning the Parameters of a MR-Sort Model

$$
\begin{array}{lll}
\max \sum_{a \in A} \gamma_a & & \\
\text{s.t.} & & \\
\sum_{j=1}^{n} c_{a,j}^{h-1} \geq \lambda + M(\gamma_a - 1) & \forall a \in A_h, h = \{2, ..., p\} & \\
\sum_{j=1}^{n} c_{a,j}^{h} < \lambda - M(\gamma_a - 1) & \forall a \in A_h, h = \{1, ..., p-1\} & \\
a_j - b_{l,j} < M \cdot \delta_{a,j}^{l} & \forall j \in F, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} & \\
a_j - b_{l,j} \geq M(\delta_{a,j}^{l} - 1) & \forall j \in F, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} & \\
c_{a,j}^{l} \leq \delta_{a,j}^{l} & \forall j \in F, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} & \\
c_{a,j}^{l} \leq w_j & \forall j \in F, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} & \\
c_{a,j}^{l} \geq \delta_{a,j}^{l} - 1 + w_j & \forall j \in F, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} & \\
b_{h,j} \geq b_{h-1,j} & \forall j \in F, h = \{2, ..., p-1\} & \\
\sum_{j=1}^{n} w_j = 1 & & \\
\delta_{a,j}^{l} \in \{0, 1\} & \forall j \in F, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} & \\
c_{a,j}^{l} \in [0, 1] & \forall j \in F, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} & \\
b_{h,j} \in \mathbb{R} & \forall j \in F, \forall h \in P & \\
\gamma_a \in \{0, 1\} & \forall a \in A & \\
w_j \in [0, 1] & \forall j \in F & \\
\lambda \in [0.5, 1] & &
\end{array}
\tag{12}
$$

B MIP Learning the Parameters of a 2-Additive NCS Model

$$
\begin{aligned}
&\max \textstyle\sum_{a\in A} \gamma_a \\
&\text{s.t.} \\
&\sum_{j=1}^{n}\left(c_{a,j}^{h-1}+\sum_{k=1}^{j}\alpha_{a,j,k}^{h-1}-\sum_{k=1}^{j}\beta_{a,j,k}^{h-1}\right)\geq \lambda + M(\gamma_a-1)\ \forall a\in A_h, && h=\{2,...,p\}\\
&\sum_{j=1}^{n}\left(c_{a,j}^{h}+\sum_{k=1}^{j}\alpha_{a,j,k}^{h}-\sum_{k=1}^{j}\beta_{a,j,k}^{h}\right)< \lambda - M(\gamma_a-1)\ \forall a\in A_h, && h=\{1,\cdots,p-1\}\\
&m_j+\sum_{k\in J}(m_{j,k}^{+}-m_{j,k}^{-})\geq 0 && \forall j\in F, \forall J\subseteq F\backslash\{j\}\\
&\sum_{j=1}^{n} m_j+\sum_{j=1}^{n}\sum_{k=1}^{j}(m_{j,k}^{+}-m_{j,k}^{-})=1 &&\\
&b_{h,j}\geq b_{h-1,j} && \forall j\in F, h=\{2,...,p\}\\
&c_{a,j}^{l}\leq \delta_{a,j}^{l} && \forall j\in F, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&c_{a,j}^{l}\leq m_j && \forall j\in F, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&c_{a,j}^{l}-m_j\geq \delta_{a,j}^{l}-1 && \forall j\in F, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&a_j-b_{l,j}< M\cdot\delta_{a,j}^{l} && \forall j\in F, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&a_j-b_{l,j}\geq M(\delta_{a,j}^{l}-1) && \forall j\in F, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&\delta_{a,j}^{l}+\delta_{a,k}^{l}\geq 2\Delta_{a,j,k}^{l} && \forall\{j,k\}\in F: k<j, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&\delta_{a,j}^{l}+\delta_{a,k}^{l}\leq \Delta_{a,j,k}^{l}+1 && \forall\{j,k\}\in F: k<j, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&\alpha_{a,j,k}^{l}\leq \Delta_{a,j,k}^{l} && \forall\{j,k\}\in F: k<j, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&\alpha_{a,j,k}^{l}\leq m_{j,k}^{+} && \forall\{j,k\}\in F: k<j, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&\alpha_{a,j,k}^{l}+m_{j,k}^{+}\geq \Delta_{a,j,k}^{l}-1 && \forall\{j,k\}\in F: k<j, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&\beta_{a,j,k}^{l}\leq \Delta_{a,j,k}^{l} && \forall\{j,k\}\in F: k<j, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&\beta_{a,j,k}^{l}\leq m_{j,k}^{-} && \forall\{j,k\}\in F: k<j, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&\beta_{a,j,k}^{l}-m_{j,k}^{-}\geq \Delta_{a,j,k}^{l}-1 && \forall\{j,k\}\in F: k<j, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&c_{a,j}^{l}\in[0,1] && \forall j\in F, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&\delta_{a,j}^{l}\in\{0,1\} && \forall j\in F, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&\alpha_{a,j,k}^{l},\beta_{a,j,k}^{l}\in[0,1] && \forall\{j,k\}\in F: k<j, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&\Delta_{a,j,k}^{l}\in\{0,1\} && \forall\{j,k\}\in F: k<j, \forall a\in A_h, \forall h\in P, l=\{h-1,h\}\\
&m_j\in[0,1] && \forall j\in F\\
&m_{j,k}^{+},m_{j,k}^{-}\in[0,1] && \forall j\in F, \forall k\in F, k<j\\
&b_{h,j}\in\mathbb{R} && \forall j\in F, \forall h\in P\\
&\gamma_a\in\{0,1\} && \forall a\in A\\
&\lambda\in[0,1] &&
\end{aligned}
\tag{13}
$$

References

1. Bouyssou, D., Marchant, T.: An axiomatic approach to noncompensatory sorting methods in MCDM, I: the case of two categories. Eur. J. Oper. Res. **178**(1), 217–245 (2007)
2. Bouyssou, D., Marchant, T.: An axiomatic approach to noncompensatory sorting methods in MCDM, II: more than two categories. Eur. J. Oper. Res. **178**(1), 246–276 (2007)
3. Leroy, A., Mousseau, V., Pirlot, M.: Learning the parameters of a multiple criteria sorting method based on a majority rule. In: Brafman, R. (ed.) ADT 2011. LNCS, vol. 6992, pp. 219–233. Springer, Heidelberg (2011)
4. Sobrie, O., Mousseau, V., Pirlot, M.: Learning a majority rule model from large sets of assignment examples. In: Perny, P., Pirlot, M., Tsoukiàs, A. (eds.) ADT 2013. LNCS, vol. 8176, pp. 336–350. Springer, Heidelberg (2013)
5. Jacquet-Lagrèze, E., Siskos, Y.: Assessing a set of additive utility functions for multicriteria decision making: the UTA method. Eur. J. Oper. Res. **10**, 151–164 (1982)
6. Doumpos, M., Zopounidis, C.: Multicriteria Decision Aid Classification Methods. Kluwer Academic Publishers, Dordrecht (2002)
7. Fishburn, P.C.: Noncompensatory preferences. Synth. **33**(1), 393–403 (1976)
8. Bouyssou, D.: Some remarks on the notion of compensation in MCDM. Eur. J. Oper. Res. **26**, 150–160 (1986)
9. Grabisch, M.: The application of fuzzy integrals in multicriteria decision making. Eur. J. Oper. Res. **89**(3), 445–456 (1996)
10. Yu, W.: Aide multicritère à la décision dans le cadre de la problématique du tri: méthodes et applications. Ph.D. thesis, LAMSADE, Université Paris Dauphine, Paris (1992)
11. Roy, B., Bouyssou, D.: Aide multicritère à la décision: méthodes et cas. Economica, Paris (1993)
12. Bouyssou, D., Pirlot, M.: A characterization of concordance relations. Eur. J. Oper. Res. **167**(2), 427–443 (2005)
13. Bouyssou, D., Pirlot, M.: Further results on concordance relations. Eur. J. Oper. Res. **181**, 505–514 (2007)
14. Chateauneuf, A., Jaffray, J.: Derivation of some results on monotone capacities by Möbius inversion. In: Bouchon-Meunier, B., Yager, R.R. (eds.) IPMU 1986. Lecture Notes in Computer Science, vol. 286, pp. 95–102. Springer, Heidelberg (1986)
15. Mousseau, V., Słowiński, R.: Inferring an ELECTRE TRI model from assignment examples. J. Global Optim. **12**(1), 157–174 (1998)
16. Mousseau, V., Figueira, J., Naux, J.P.: Using assignment examples to infer weights for ELECTRE TRI method: some experimental results. Eur. J. Oper. Res. **130**(1), 263–275 (2001)
17. The, A.N., Mousseau, V.: Using assignment examples to infer category limits for the ELECTRE TRI method. J. Multi-criteria Decis. Anal. **11**(1), 29–43 (2002)
18. Dias, L., Mousseau, V., Figueira, J., Clímaco, J.: An aggregation/disaggregation approach to obtain robust conclusions with ELECTRE TRI. Eur. J. Oper. Res. **138**(1), 332–348 (2002)
19. Doumpos, M., Marinakis, Y., Marinaki, M., Zopounidis, C.: An evolutionary approach to construction of outranking models for multicriteria classification: the case of the ELECTRE TRI method. Eur. J. Oper. Res. **199**(2), 496–505 (2009)

20. Cailloux, O., Meyer, P., Mousseau, V.: Eliciting ELECTRE TRI category limits for a group of decision makers. Eur. J. Oper. Res. **223**(1), 133–140 (2012)
21. Zheng, J., Metchebon, S., Mousseau, V., Pirlot, M.: Learning criteria weights of an optimistic Electre Tri sorting rule. Comput. OR **49**, 28–40 (2014)
22. Hüllermeier, E., Tehrani, A.F.: Efficient learning of classifiers based on the 2-additive choquet integral. In: Moewes, C., Nürnberger, A. (eds.) Computational Intelligence in Intelligent Data Analysis. SCI, vol. 445, pp. 17–29. Springer, Heidelberg (2013)
23. Tehrani, A.F., Cheng, W., Dembczynski, K., Hüllermeier, E.: Learning monotone nonlinear models using the choquet integral. Mach. Learn. **89**(1–2), 183–211 (2012)
24. Ersek Uyanık, E., Sobrie, O., Mousseau, V., Pirlot, M.: Listing the families of sufficient coalitions of criteria involved in sorting procedures. In: DA2PL 2014 Workshop From Multiple Criteria Decision Aid to Preference Learning, pp. 60–70, Paris, France (2014)