Learning the parameters of a majority rule sorting model taking attribute interactions into account

Olivier Sobrie[1,3,4] **, Vincent Mousseau**[2] and **Marc Pirlot**[3]

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. An object is assigned in a category if it is as good as the category lower profile and not better than the category upper profile. To determine if 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. In view of improving the expressiveness of the model, we modify it by introducing capacities to quantify the power of the coalitions. In the paper we describe a mixed integer program and a metaheuristic that give the possibility to learn the parameters of this model from examples of assignment. We test the metaheuristic on real datasets.

1 Introduction

In Multiple Criteria Decision Analysis, the sorting problematic 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 such type of problematic. In this paper, we consider a sorting model based on a majority rule, called MR-Sort [11, 17]. In MR-Sort, the categories are separated by profiles which are vectors of performances on the different criteria. Each criterion of the model is associated to a weight representing its importance. Using this model, an alternative is assigned in a category if (a) it is considered at least as good as the category lower profile and (b) it is not considered at least as good as the category upper profile. An alternative is considered as good as a profile if its performances are at least as good as the profile performances on a weighted majority of criteria.

Consider a MR-Sort model composed of 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 in 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 coalition:

1. $c_1 \wedge c_2$
2. $c_3 \wedge c_4$
3. $c_1 \wedge c_4$
4. $c_2 \wedge c_4$

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". Using an additive MR-Sort model, it 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 3 minimal coalitions listed above are not sufficient to be considered as good as b_1 (e.g. $w_1 + w_3 < \lambda$).

Now consider the same type of model, but with the following minimal criteria coalitions:

1. $c_1 \wedge c_2$
2. $c_3 \wedge c_4$

Modeling this classification rule with an additive MR-Sort model is impossible. There exist no weights and majority threshold satisfying solely the 2 minimal criteria coalitions. In view of being able to represent such type of rule, we propose in this paper a new formulation of MR-Sort, called Capacitive-MR-Sort. This formulation expresses the majority rule of MR-Sort with capacities like in the Choquet Integral [8].

The paper is organized as follows. The next section describes formally the MR-Sort model and the new formulation of MR-Sort with capacities. Section 3 recalls the literature dealing with learning parameters of MR-Sort models from assignment examples. The next two sections describe respectively a Mixed Integer Program and a metaheuristic that allow to learn the parameters of a Capacitive-MR-Sort. Some experimental results are finally presented.

2 MR-Sort and Capacitive-MR-Sort

2.1 MR-Sort

MR-Sort is a method for assigning objects in ordered categories. Each object is described by a multicriteria vector of attribute values. The attribute values can be meaningfully ordered, i.e. there is an underlying order on each attribute scale, which is interpreted as a "better than" relation. Categories are determined by limit profiles, which are vectors of attribute values. The lower limit profile of a category is the upper limit profile of the category below. The MR-Sort rule works as follows. An object is assigned to a category if it is better than

[1] email: olivier.sobrie@gmail.com
[2] École Centrale Paris, Grande Voie des Vignes, 92295 Châtenay Malabry, France, email: vincent.mousseau@ecp.fr
[3] Université de Mons, Faculté Polytechnique, 9, rue de Houdain, 7000 Mons, Belgium, email: marc.pirlot@umons.ac.be

the lower limit profile of the category on a sufficiently large coalition of (weighted) attributes and this condition is not met for the upper limit profile of this category. Obviously, MR-Sort is a monotone rule, i.e. an object that is at least as good as another on all attributes cannot be assigned to a lower category.

The MR-Sort rule is a simplified version of the ELECTRE TRI procedure, a method that is used in MCDA to assign objects to predefined categories [19, 16]. The underlying semantic is generally to assign the objects labels such as "good", "average", "bad", ...

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 thus 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 $P = \{1, ..., p-1\}$ the list of profile 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.:

$$b_{h-1,j} \leq b_{h,j}, \quad h = 1, \ldots, p; \ j = 1, \ldots, n.$$

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 denoted 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 ([16]; see also [2, 3] for an axiomatic description of such relations).

Consequently, the condition for an object $a \in X$ to be assigned to category C_h writes:

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

The MR-Sort assignment rule described above involves $pn+1$ parameters, i.e. n weights, $(p-1)n$ profiles evaluations and one majority threshold. Note that the profiles b_0 and b_p are conventionally defined as follows: $b_{0,j}$ is a value such that $a_j \geq b_{0,j}$ for all $a \in X$ and $j = 1, \ldots, n$; $b_{p,j}$ is a value such that $a_j < b_{p,j}$ for all $a \in X$ and $j = 1, \ldots, n$.

A *learning set* A is a subset of objects $A \subseteq X$ for which an assignment is known. For $h = 1, \ldots, p$, 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 Capacitive-MR-Sort

Before describing the Capacitive-MR-Sort model, we introduce the notion of capacity. To illustrate this, we consider an application in which a committee for a higher education program has to decide about the admission of students on basis of their evaluations in 4 courses: math, physics, chemistry and history. To be accepted in the program, the committee judges that a student should have a sufficient majority of evaluations above 10/20. The 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 of courses are the minimal coalition of courses in which the evaluation should be above 10/20 in view of being accepted:

- {math, physics}
- {math, chemistry}
- {chemistry, history}

As an example of this rule, Table 1 shows evaluations of several students and, for each student, if he is accepted or refused.

	Math	Physic	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

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

Representing these assignments by using a MR-Sort model with profiles fixed at 10/20 in each course is impossible. There are no weights allowing to model such rules. MR-Sort is not adapted to model such types of rules because it does not handle criteria interactions.

In view of taking criterion interactions into account, we propose to modify the definition of the global outranking relation, $a \succcurlyeq b_h$, given in (1). We introduce the notion of capacity. 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:

$$\mu(A) = \sum_{B \subseteq A} m(B) \tag{3}$$

for all $A \subseteq F$, with $m(B)$ defined as:

$$m(B) = \sum_{C \subseteq B} (-1)^{|B|-|C|} \mu(C) \tag{4}$$

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 also called "interaction". An interaction m is a set function $m : 2^F \to [-1, 1]$ satisfying the following conditions:

$$\sum_{j \in K \subseteq J \cup \{j\}} m(K) \geq 0 \qquad \forall j \in F, J \subseteq F \backslash \{i\} \tag{5}$$

and

$$\sum_{K \subseteq F} m(K) = 1.$$

If m is an interaction, the set function defined by $\mu(A) = \sum_{B \subseteq A} m(B)$ is a capacity. Conditions (5) guarantee that μ is monotone [5].

Using a capacity to express the weight of the coalition in favor of an object, we transform the outranking rule as follows:

$$\begin{aligned} a \succcurlyeq b_h \Leftrightarrow \mu(A) \geq \lambda \text{ with } A = \{j : a_j \geq b_{h,j}\} \\ \text{and } \mu(A) = \sum_{B \subseteq A} m(B) \end{aligned} \tag{6}$$

Computing the value of $\mu(A)$ with the Möbius transform induces the evaluation of $2^{|A|}$ parameters. In a model composed of n criteria, it 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. In the literature [12], for the ranking problematic, it has been shown experimentally that a 2-additive model allows to improve the representation capabilities. However using a 3-additive capacity instead of a 2-additive one does not significantly improve the accuracy of the model. 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:

$$\mu(F_{a,h-1}) \geq \lambda \quad \text{and} \quad \mu(F_{a,h}) < \lambda \tag{7}$$

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

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 [14, 13, 15, 6, 7, 11, 4, 17, 20]. In this section, we recall how to learn the set of parameters of an MR-Sort using respectively an exact method [11] and a metaheuristic [17].

3.1 Mixed Integer Programming

Learning the parameters of an MR-Sort model using linear programming techniques has been proposed in [11]. 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 an MR-Sort model such that a majority of the examples are restored by the inferred model. We recall in this subsection the main steps to obtain the MIP formulation proposed in [11].

The definition of an outranking relation (1) can be rewritten as follows:

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

To linearize this constraint, we introduce for each value $c_{a,j}^h$, a binary variable $\delta_{a,j}^l$ that it is equal to 1 when the performance of the object a is at least equal 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:

$$M(\delta_{a,j}^l - 1) \leq a_j - b_{l,j} < M \cdot \delta_{a,j}^l \tag{8}$$

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

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

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, we introduce new binary variables γ_a, equal to 1 if object a is assigned in the expected category, i.e. the category it has been assigned in the learning set, and equal to 0 otherwise. To deduce the value of γ_a variables, two additional constraints are added:

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

Finally, the combination of all the constraints leads to the MIP given in (9).

3.2 Metaheuristic

The MIP presented in the previous section is not suitable for large datasets 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 datasets, a metaheuristic has been proposed in [17]. As in the MIP, the metaheuristic takes as input a set of assignment examples and their vector of performances and returns the parameters of an MR-Sort model.

The metaheuristic proposed in [17] 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:

$$\left\{\begin{array}{rcll}
\max & & \sum_{a \in A} \gamma_a & \\
\sum_{j=1}^{n} c_{a,j}^{h-1} & \geq & \lambda + M(\gamma_a - 1) & \forall a \in A_h, h = \{2, ..., p\} \\
\sum_{j=1}^{n} c_{a,j}^{h} & < & \lambda - M(\gamma_a - 1) & \forall a \in A_h, h = \{1, ..., p-1\} \\
a_j - b_{l,j} & < & M \cdot \delta_{a,j}^{l} & \forall j \in F, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} \\
a_j - b_{l,j} & \geq & M(\delta_{a,j}^{l} - 1) & \forall j \in F, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} \\
c_{a,j}^{l} & \leq & \delta_{a,j}^{l} & \forall j \in F, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} \\
c_{a,j}^{l} & \leq & w_j & \forall j \in F, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} \\
c_{a,j}^{l} & \geq & \delta_{a,j}^{l} - 1 + w_j & \forall j \in F, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} \\
b_{h,j} & \geq & b_{h-1,j} & \forall j \in F, h = \{2, ..., p-1\} \\
\sum_{j=1}^{n} w_j & = & 1 & \\
\delta_{a,j}^{l} & \in & \{0, 1\} & \forall j \in F, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} \\
c_{a,j}^{l} & \in & [0, 1] & \forall j \in F, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} \\
b_{h,j} & \in & \mathbb{R} & \forall j \in F, \forall h \in P \\
\gamma_a & \in & \{0, 1\} & \forall a \in X \\
w_j & \in & [0, 1] & \forall j \in F \\
\lambda & \in & [0.5, 1] &
\end{array}\right. \tag{9}$$

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

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

The linear program designed to learn the weights and the majority threshold is given by (10). 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 in the category defined in the input dataset. We remark that the objective function of the linear program does not explicitly minimize the 0/1 loss but a sum of slacks. It implies that compensatory effects might appears to the detriment of the 0/1 loss. However in this metaheuristic, we consider that this effects are acceptable. This linear program doesn't contain 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 which correspond to the performances of the examples on criterion j located between the 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 classified 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.

In order to formally define these sets we introduce the following notation. A_h^l denotes the subset of misclassified objects that are assigned in category C_l by the model while in the dataset, they are assigned in category C_h. $A_{<h}^{>l}$ denotes the subset of misclassified objects that are assigned in category higher than C_l by the model while in the dataset it is assigned in a category below C_h. We denote by $\sigma(a, b_h) =$

$$\begin{cases} \min & \sum_{a \in A}(x'_a + y'_a) & \\ \sum_{j:a_j \geq b_{h-1,j}} w_j - x_a + x'_a = \lambda & \forall a \in A_h, \forall h \in P\backslash\{1\} \\ \sum_{j:a_j \geq b_{h,j}} w_j + y_a - y'_a = \lambda - \delta & \forall a \in A_h, \forall h \in P\backslash\{p-1\} \\ \sum_{j=1}^{n} w_j = 1 & \\ w_j \in [0;1] & \forall j \in F \\ \lambda \in [0.5;1] & \\ x_a, y_a, x'_a, y'_a \in \mathbb{R}_0^+ & \end{cases} \tag{10}$$

$\sum_{j:a_j \geq b_{h,j}} w_j$, the sum of criteria weights in favor of object a against profile b_h. We have, for any h, j and positive δ:

$$\begin{aligned}
V_{h,j}^{+\delta} &= \left\{a \in A_h^{h+1} : b_{h,j} + \delta > a_j \geq b_{h,j} \text{ and } \sigma(a, b_h) - w_j < \lambda\right\} \\
V_{h,j}^{-\delta} &= \left\{a \in A_{h+1}^{h} : b_{h,j} - \delta < a_j < b_{h,j} \text{ and } \sigma(a, b_h) + w_j \geq \lambda\right\} \\
W_{h,j}^{+\delta} &= \left\{a \in A_h^{h+1} : b_{h,j} + \delta > a_j \geq b_{h,j} \text{ and } \sigma(a, b_h) - w_j \geq \lambda\right\} \\
W_{h,j}^{-\delta} &= \left\{a \in A_{h+1}^{h} : b_{h,j} - \delta < a_j < b_{h,j} \text{ and } \sigma(a, b_h) + w_j < \lambda\right\} \\
Q_{h,j}^{+\delta} &= \left\{a \in A_{h+1}^{h+1} : b_{h,j} + \delta > a_j \geq b_{h,j} \text{ and } \sigma(a, b_h) - w_j < \lambda\right\} \\
Q_{h,j}^{-\delta} &= \left\{a \in A_h^{h} : b_{h,j} - \delta < a_j < b_{h,j} \text{ and } \sigma(a, b_h) + w_j \geq \lambda\right\} \\
R_{h,j}^{+\delta} &= \left\{a \in A_{h+1}^{h} : b_{h,j} + \delta > a_j \geq b_{h,j}\right\} \\
R_{h,j}^{-\delta} &= \left\{a \in A_h^{h+1} : b_{h,j} - \delta < a_j < b_{h,j}\right\} \\
T_{h,j}^{+\delta} &= \left\{a \in A_{<h}^{>h} : b_{h,j} + \delta > a_j \geq b_{h,j}\right\} \\
T_{h,j}^{-\delta} &= \left\{a \in A_{>h+1}^{<h+1} : b_{h,j} - \delta < a_j \leq b_{h,j}\right\}
\end{aligned}$$

The evaluation of the candidate move 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 Capacitive-MR-Sort model

As compared to a MR-Sort with additive weights, a MR-Sort model with capacities implies more parameters. In a standard MR-Sort model, a weight is associated to each criterion, which makes overall n parameters to elicit. With an MR-Sort model limited to two-additive capacities, the computation of the weights 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 $n + \frac{n(n-1)}{2} - 1 = \frac{n(n+1)}{2} - 1$ 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 (5) on the interaction read:

$$m_j + \sum_{k \in J} m_{j,k} \geq 0 \qquad \forall j \in F, \forall J \subseteq F\backslash\{j\} \tag{11}$$

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 [10], 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 introduces n^2 extra variables and reduces the number of constraints to $n^2 + 1$ without introducing non linearities.

With a 2-additive MR-Sort model, the constraints for an alternative a to be assigned in a category h (7) can also be expressed as follows:

$$\begin{cases} \sum_{j=1}^{n} c_{a,j}^{h-1} + \sum_{j=1}^{n} \sum_{k=1}^{j} c_{a,j,k}^{h-1} & \geq \lambda + M(\gamma_a - 1) \\ \sum_{j=1}^{n} c_{a,j}^{h} + \sum_{j=1}^{n} \sum_{k=1}^{j} c_{a,j,k}^{h} & < \lambda - M(\gamma_a - 1) \end{cases} \tag{12}$$

with:

- $c_{a,j}^{h-1}$ (resp. $c_{a,j}^{h}$) equals m_j if 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 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 $a \in X$, $j \in F$ and $l \in P$, constraints (11) 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 $c_{a,j}^{h}$ can be 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 coefficient (13).

$$\begin{cases} c_{a,j}^{l} & \leq \delta_{a,j}^{l} \\ c_{a,j}^{l} & \leq m_j \\ c_{a,j}^{l} & \geq \delta_{a,j}^{l} - 1 + m_j \end{cases} \tag{13}$$

However it is not the case for the variables $c_{a,j,k}^{h-1}$ and $c_{a,j,k}^{h}$, because they imply two criteria. To linearize the formulation, we introduce new binary variables, $\Delta^l_{a,j,k}$ 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 $\Delta^l_{a,j,k}$ thanks to the conjunction of constraints given at (8) and the following constraints:

$$2\Delta^l_{a,j,k} \leq \delta^l_{a,j} + \delta^k_{a,j} \leq \Delta^l_{a,j,k} + 1$$

In order to deduce the value of $c^l_{a,j,k}$, which can be either positive or negative, for all $l \in P$, we decompose the variable in two parts, $\alpha^l_{a,j,k}$ and $\beta^l_{a,j,k}$ such that $c^l_{a,j,k} = \alpha^l_{a,j,k} - \beta^l_{a,j,k}$ with $\alpha^l_{a,j,k} \geq 0$ and $\beta^l_{a,j,k} \geq 0$. 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 value of $\alpha^l_{a,j,k}$ and $\beta^l_{a,j,k}$ are finally obtained thanks to the following constraints:

$$\begin{cases} \alpha^l_{a,j,k} & \leq \Delta^l_{a,j,k} \\ \alpha^l_{a,j,k} & \leq m^+_{j,k} \\ \alpha^l_{a,j,k} & \geq \Delta^l_{a,j,k} - 1 + m^+_{j,k} \end{cases} \qquad \begin{cases} \beta^l_{a,j,k} & \leq \Delta^l_{a,j,k} \\ \beta^l_{a,j,k} & \leq m^-_{j,k} \\ \beta^l_{a,j,k} & \geq \Delta^l_{a,j,k} - 1 + m^-_{j,k} \end{cases}$$

Finally, we obtain the MIP given in (14).

5 Metaheuristic to learn a Capacitive-MR-Sort model

The MIP described in the previous section requires a lot of binary variables and is therefore unsuitable for large problems. In subsection 3.2, we described the principle of a metaheuristic designed to learn the parameters of an MR-Sort model. In this section, we describe an adaptation of the metaheuristic in view of learning the parameters of a Capacitive-MR-Sort model. Like for the MIP described in the previous section, we limit the model to 2-additive capacities in order to reduce the number of coefficient in comparison to a model with a general capacity.

The main component that needs to be adapted in the metaheuristic in order to be able to learn a Capacitive-MR-Sort model is the linear program that infers the weights and the majority threshold (10). 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 given in (15).

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 the object a is assigned to the right 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, it is needed to adapt the formal definition of the sets in which objects are classified for computing the candidate move evaluation. The semantic of the sets, described in Section 3.2 remains the same, only the formal definitions of the sets are adapted as follows.

$$\begin{aligned}
V^{+\delta}_{h,j} &= \Big\{a \in A^{h+1}_h : b_{h,j} + \delta > a_j \geq b_{h,j} \text{ and } \mu(F_{a,h} \setminus \{j\}) < \lambda\Big\} \\
V^{-\delta}_{h,j} &= \Big\{a \in A^h_{h+1} : b_{h,j} - \delta < a_j < b_{h,j} \text{ and } \mu(F_{a,h} \cup \{j\}) \geq \lambda\Big\} \\
W^{+\delta}_{h,j} &= \Big\{a \in A^{h+1}_h : b_{h,j} + \delta > a_j \geq b_{h,j} \text{ and } \mu(F_{a,h} \setminus \{j\}) \geq \lambda\Big\} \\
W^{-\delta}_{h,j} &= \Big\{a \in A^h_{h+1} : b_{h,j} - \delta < a_j < b_{h,j} \text{ and } \mu(F_{a,h} \cup \{j\}) < \lambda\Big\} \\
Q^{+\delta}_{h,j} &= \Big\{a \in A^{h+1}_{h+1} : b_{h,j} + \delta > a_j \geq b_{h,j} \text{ and } \mu(F_{a,h} \setminus \{j\}) < \lambda\Big\} \\
Q^{-\delta}_{h,j} &= \Big\{a \in A^h_h : b_{h,j} - \delta < a_j < b_{h,j} \text{ and } \mu(F_{a,h} \cup \{j\}) \geq \lambda\Big\}
\end{aligned}$$

The formal definitions of the sets $R^{+\delta}_{h,j}$, $R^{-\delta}_{h,j}$, $T^{+\delta}_{h,j}$ remain the same as for the simple additive MR-Sort model as well as function computing the evaluations taking into account the size of the sets.

6 Experimentations

The use of the MIP for learning a Capacitive-MR-Sort model is limited because of the high number of binary variables it involves. It contains more binary variables than the MIP learning the parameters of a simple additive MR-Sort model. In [11], experiments 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 learning Capacitive-MR-Sort models.

In view of assessing the performances of the metaheuristic designed for learning the parameters of a Capacitive-MR-Sort model, we used it to learn Capacitive-MR-Sort models from several real datasets presented in Table 2. These datasets have been found in the UCI machine learning repository [1] and in WEKA [9]. They have been already used to assess the learning performances of other algorithms, like in [18] and [17]. The dataset presented in Table 2 contains from 120 to 1728 instances, 4 to 8 criteria (criteria) and 2 to 36 categories. In the experimentations, the categories have been binarized by thresholding at the median (like in [18, 17]). All the input criteria of the datasets are considered as monotone.

Dataset	#instances	#criteria	#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	9
LEV	1000	4	5
CEV	1728	6	4

Table 2. Datasets

In a first experiment, we used 50% of the alternatives contained in the datasets as learning set and the rest as test

$$
\left\{\begin{array}{rcll}
\max & & \displaystyle\sum_{a \in A} \gamma_a & \\
\displaystyle\sum_{j=1}^{n} c_{a,j}^{h-1} + \sum_{j=1}^{n}\sum_{k=1}^{j} \alpha_{a,j,k}^{h-1} - \sum_{j=1}^{n}\sum_{k=1}^{j} \beta_{a,j,k}^{h-1} & \geq & \lambda + M(\gamma_a - 1) & \forall a \in A_h, h = 2, ..., p \\
\displaystyle\sum_{j=1}^{n} c_{a,j}^{h} + \sum_{j=1}^{n}\sum_{k=1}^{j} \alpha_{a,j,k}^{h} - \sum_{j=1}^{n}\sum_{k=1}^{j} \beta_{a,j,k}^{h} & < & \lambda - M(\gamma_a - 1) & \forall a \in A_h, \forall h \in 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} & \geq & \delta_{a,j}^{l} - 1 + m_j & \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 \in F, \forall 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 \in F, \forall 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 \in F, \forall 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 \in F, \forall k \in F, k < j, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} \\
\alpha_{a,j,k}^{l} & \geq & \Delta_{a,j,k}^{l} - 1 + m_{j,k}^{+} & \forall j \in F, \forall 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 \in F, \forall 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 \in F, \forall k \in F, k < j, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} \\
\beta_{a,j,k}^{l} & \geq & \Delta_{a,j,k}^{l} - 1 + m_{j,k}^{-} & \forall j \in F, \forall k \in F, k < j, \forall a \in A_h, \forall h \in P, l = \{h-1, h\} \\
m_j + \displaystyle\sum_{k \in J} (m_{j,k}^{+} - m_{j,k}^{-}) & \geq & 0 & \forall j \in F, \forall J \subseteq F \backslash \{j\} \\
b_{h,j} & \geq & b_{h-1,j} & \forall j \in F, h = \{2, ..., p-1\} \\
\displaystyle\sum_{j=1}^{n} m_j + \sum_{j=1}^{n}\sum_{k=1}^{j} (m_{j,k}^{+} - m_{j,k}^{-}) & = & 1 & \\
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 \in F, \forall 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 \in F, \forall 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 X \\
\lambda & \in & [0,1] &
\end{array}\right.
\tag{14}
$$

set. From the examples of the learning set, we learned MR-Sort and Capacitive-MR-Sort models with the metaheuristic. We repeated the operation for 100 random splittings of the datasets in learning and test sets. The results are given in Table 3. We see that the average classification accuracy obtained with the Capacitive-MR-Sort metaheuristic is in average comparable to the one obtained with the MR-Sort metaheuristic. For some datasets, the Capacitive-MR-Sort metaheuristic gives better results but sometimes it is the contrary. The use of a more descriptive model does not help to improve the classification accuracy of the test set.

The second experiment we did consisted in using all the instances of the datasets as learning set. As in the first experiment, for each dataset, we run the two metaheuristic with 100 different seeds. The average classification accuracy and the standard deviation of the learning set of each dataset is

Dataset	META MR-Sort	META Capa-MR-Sort
DBS	0.8400 ± 0.0456	0.8306 ± 0.0466
CPU	0.9270 ± 0.0294	0.9203 ± 0.0315
BCC	0.7271 ± 0.0379	0.7262 ± 0.0377
MPG	0.8174 ± 0.0290	0.8167 ± 0.0468
ESL	0.8992 ± 0.0195	0.9018 ± 0.0172
MMG	0.8303 ± 0.0154	0.8318 ± 0.0121
ERA	0.6905 ± 0.0192	0.6927 ± 0.0165
LEV	0.8454 ± 0.0221	0.8445 ± 0.0223
CEV	0.9217 ± 0.0067	0.9187 ± 0.0153

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

$$\left\{\begin{array}{rcll}
\min & & \sum_{a \in A}(x'_a + y'_a) & \\
\sum_{j:a_j \geq b_{h-1,j}}^{n}\left(m_j + \sum_{k:a_k \geq b_{h-1,k}}^{j} m_{j,k}\right) - x_a + x'_a & = & \lambda & \forall a \in A_h, \forall h \in P \backslash \{1\} \\
\sum_{j:a_j \geq b_{h,j}}^{n}\left(m_j + \sum_{k:a_k \geq b_{h,k}}^{j} m_{j,k}\right) + y_a - y'_a & = & \lambda - \varepsilon & \forall a \in A_h, \forall h \in P \backslash \{p-1\} \\
\sum_{j=1}^{n} m_j + \sum_{j=1}^{n}\sum_{k=1}^{j} m_{j,k} & = & 1 & \\
m_j + \sum_{k \in J} m_{j,k} & \geq & 0 & \forall j \in F, \forall J \subseteq F \backslash \{j\} \\
\lambda & \in & [0.5; 1] & \\
m_j & \in & [0,1] & \forall j \in F \\
m_{j,k} & \in & [-1,1] & \forall j \in F, \forall k \in F, k < j \\
x_a, y_a, x'_a, y'_a & \in & \mathbb{R}_0^+ & a \in A.
\end{array}\right. \tag{15}$$

given in Table 4. The Capacitive-MR-Sort metaheuristic does not always give better results than the MR-Sort one.

Dataset	META MR-Sort	META Capa-MR-Sort
DBS	0.9318 ± 0.0036	0.9247 ± 0.0099
CPU	0.9761 ± 0.0000	0.9694 ± 0.0072
BCC	0.7737 ± 0.0013	0.7700 ± 0.0077
MPG	0.8418 ± 0.0000	0.8418 ± 0.0000
ESL	0.9180 ± 0.0000	0.9180 ± 0.0000
MMG	0.8491 ± 0.0011	0.8508 ± 0.0005
ERA	0.7142 ± 0.0028	0.7158 ± 0.0004
LEV	0.8650 ± 0.0000	0.8650 ± 0.0000
CEV	0.9225 ± 0.0000	0.9225 ± 0.0000

Table 4. Average and standard deviation of the classification accuracy of the learning set when using the MR-Sort and Capacitive-MR-Sort models when using all the dataset as learning set

The average computing time required to obtain the results presented in Table 4 is given in Table 5. We observe that learning a Capacitive-MR-Sort model can take up to almost 3 times the time required to learn the parameters of a simple MR-Sort model.

Dataset	META MR-Sort	META Capa-MR-Sort
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

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

The two experiments show that using a more expressive model does not always result in a better classification accuracy. This observation raises two questions. Firstly, in view of the results obtained, one may doubt that the Capacitive-MR-Sort extends much the original MR-Sort. For what type of assignment data is the new model more flexible? Secondly, is the metaheuristic well-adapted to learn Capacitive-MR-Sort models? To answer these questions, more experimentations have to be done.

7 Comments

We observe that using 2-additive weights instead of simple additive weights in MR-Sort does not result in significant improvement of the 0/1 loss. It is somewhat surprising because the model is more flexible when 2-additive weights are used.

In view of understanding better how the representation capabilities of an MR-Sort model can be improved by using 2-additive weights, we do the following experimentation. We modify the MIP presented in section 3.1 to learn only the weights and the majority threshold of an MR-Sort model on basis of fixed profiles and assignment examples. The objective function of the MIP remains the minimization of the 0/1 loss. The MIP is used to learn the parameters of an MR-Sort model composed of 2 categories, $C_1 \succ C_2$, 4 to 6 criteria, and a fixed profile equals to 0.5 on all the criteria. Each of this learning sets contains 2^n alternatives, with n being the number of criteria of the model that is learnt. Performances of the alternatives of the learning are either equal to 0 or 1 on each criterion and the learning set is built such that each vector of performances is represented once and only once. Alternatives in the learning set are assigned either in C_1 or C_2 such that monotonicity is guaranteed in assignments, i.e. an alternative, x, which has at least equal or better performances on each criterion than another one, y, is never assigned in a least preferred category than the category in which y is assigned. In the experiment, we consider all the non-additive learning sets, i.e. all the learning sets which are not fully compatible with a simple additive MR-Sort model composed of n criteria.

Results of the experimentation are presented in Table 7.

Each row of the table contains the results for a given number of criteria, n. The second column contains the percentage of learning sets that are not compatible with a simple additive MR-Sort model composed of n criteria, among all the learning sets combinations. The last three columns contain the min, max and average percentage of 2^n examples that cannot be restored by a simple additive model among the non-additive learning sets. We observe that a MR-Sort model composed of 4 criteria is, in worst case, not able to restore 6.2% of the examples of the learning set (1 example on 16). With 5 and 6 criteria, the maximum 0/1 loss increases respectively to 9.4% and 12.4%. We see that the proportion of the alternatives that cannot be restored with a simple MR-Sort model is small. This observation might explain the poor gain observed with the Capacitive-MR-Sort metaheuristic compared to the MR-Sort one.

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 %

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

8 Conclusion

In this paper, we proposed an extension of the MR-Sort model by adding capacitive weights to the model. We called it Capacitive-MR-Sort. We also modified the MIP presented in [11] and the metaheuristic described in [17] in view of being able to learn Capacitive-MR-Sort models. The MIP formulation induces a lot of binary variables and is unsuitable for problems involving large datasets. As we want to be able to deal with real datasets, which are often large, we made experiments with the metaheuristic. Tests have been done on well-known datasets and showed that a more flexible model, the Capacitive-MR-Sort, does not guarantee to get a better classification accuracy. More experiments have to be done in view of being able to better measure and compare the representation ability of MR-Sort and Capacitive-MR-Sort models.

REFERENCES

[1] Bache, K., Lichman, M.: UCI machine learning repository (2013), http://archive.ics.uci.edu/ml

[2] Bouyssou, D., Pirlot, M.: A characterization of concordance relations. European Journal of Operational Research 167(2), 427–443 (2005)

[3] Bouyssou, D., Pirlot, M.: Further results on concordance relations. European Journal of Operational Research 181, 505–514 (2007)

[4] Cailloux, O., Meyer, P., Mousseau, V.: Eliciting ELECTRE TRI category limits for a group of decision makers. European Journal of Operational Research 223(1), 133–140 (2012)

[5] Chateauneuf, A., Jaffray, J.: Derivation of some results on monotone capacities by Möbius inversion. In: Bouchon-Meunier, B., Yager, R.R. (eds.) Uncertainty in Knowledge-Based Systems, International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU '86, Paris, France, June 30 - July 4, 1986, Selected and Extended Contributions. Lecture Notes in Computer Science, vol. 286, pp. 95–102. Springer (1986), http://dx.doi.org/10.1007/3-540-18579-8_8

[6] Dias, L., Mousseau, V., Figueira, J., Clímaco, J.: An aggregation/disaggregation approach to obtain robust conclusions with ELECTRE TRI. European Journal of Operational Research 138(1), 332–348 (2002)

[7] 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. European Journal of Operational Research 199(2), 496–505 (2009)

[8] Grabisch, M.: The application of fuzzy integrals in multicriteria decision making. European Journal of Operational Research 89(3), 445 – 456 (1996), http://www.sciencedirect.com/science/article/pii/037722179500176X

[9] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: An update. SIGKDD Exploration Newsletter 11(1), 10–18 (Nov 2009), http://doi.acm.org/10.1145/1656274.1656278

[10] Hüllermeier, E., Tehrani, A.: Efficient learning of classifiers based on the 2-additive Choquet integral. In: Moewes, C., Nürnberger, A. (eds.) Computational Intelligence in Intelligent Data Analysis, Studies in Computational Intelligence, vol. 445, pp. 17–29. Springer Berlin Heidelberg (2013), http://dx.doi.org/10.1007/978-3-642-32378-2_2

[11] Leroy, A., Mousseau, V., Pirlot, M.: Learning the parameters of a multiple criteria sorting method. In: Brafman, R., Roberts, F., Tsoukiàs, A. (eds.) Algorithmic Decision Theory, Lecture Notes in Computer Science, vol. 6992, pp. 219–233. Springer Berlin / Heidelberg (2011)

[12] Meyer, P., Pirlot, M.: On the expressiveness of the additive value function and the choquet integral models. In: DA2PL 2012 Workshop From Multiple Criteria Decision Aid to Preference Learning. pp. 48–56 (2012)

[13] Mousseau, V., Figueira, J., Naux, J.P.: Using assignment examples to infer weights for ELECTRE TRI method: Some experimental results. European Journal of Operational Research 130(1), 263–275 (2001)

[14] Mousseau, V., Słowiński, R.: Inferring an ELECTRE TRI model from assignment examples. Journal of Global Optimization 12(1), 157–174 (1998)

[15] Ngo The, A., Mousseau, V.: Using assignment examples to infer category limits for the ELECTRE TRI method. Journal of Multi-criteria Decision Analysis 11(1), 29–43 (2002)

[16] Roy, B., Bouyssou, D.: Aide multicritère à la décision: méthodes et cas. Economica Paris (1993)

[17] Sobrie, O., Mousseau, V., Pirlot, M.: Learning a majority rule model from large sets of assignment examples. In: Perny, P., Pirlot, M., Tsoukiás, A. (eds.) Algorithmic Decision Theory. pp. 336–350. Springer (2013)

[18] Tehrani, A.F., Cheng, W., Dembczynski, K., Hüllermeier, E.: Learning monotone nonlinear models using the Choquet integral. Machine Learning 89(1-2), 183–211 (2012)

[19] 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)

[20] Zheng, J., Metchebon, S., Mousseau, V., Pirlot, M.: Learning criteria weights of an optimistic Electre Tri sorting rule. Computers & OR 49(0), 28 – 40 (2014), http://www.sciencedirect.com/science/article/pii/S0305054814000677