Eliciting Electre Tri category limits for a group of decision makers

Olivier Cailloux[*] Patrick Meyer[†] Vincent Mousseau[*‡]

January 20, 2012

Abstract

Multiple criteria sorting aims at assigning alternatives evaluated on several criteria to predefined ordered categories. In this paper, we consider a well known multiple criteria sorting method, Electre Tri, which involves three types of preference parameters: (1) category limits defining the frontiers between consecutive categories, (2) weights and majority level specifying which coalitions form a majority, and (3) veto thresholds characterizing discordance effects. We propose an elicitation procedure to infer category limits from assignment examples provided by multiple decision makers. The procedure computes a set of category limits common to all decision makers, with variable weights and vetoes for each decision maker. Hence, the method helps reaching a consensus among decision makers on the category limits, whereas finding a consensus on weights and vetoes is left aside. The inference procedure is based on mixed integer linear programming and performs well even for datasets corresponding to real-world decision problems. We provide an illustrative example of the use of the method and analyze the performance of the proposed algorithms.

[*]Laboratoire Génie Industriel, École Centrale Paris, Grande Voie des Vignes, 92295 Châtenay-Malabry Cedex, France. `olivier.cailloux@ecp.fr`, `vincent.mousseau@ecp.fr`

[†]Institut TELECOM, TELECOM Bretagne, UMR CNRS 3192 Lab-STICC, Technopôle Brest Iroise CS 83818, 29238 Brest Cedex 3, Université européenne de Bretagne, France. `patrick.meyer@telecom-bretagne.eu`.

[‡]Corresponding author.

Keywords: Multiple criteria analysis; Sorting methods; Preference elicitation; Multiple decision makers.

1 Introduction

In this paper, we are interested in decision problems formulated as multicriteria sorting problems, i.e., when a finite number of alternatives from a set A evaluated on a set of criteria $\{g_j, j \in J\}$ are to be assigned to one of k predefined ordered categories $c_1 \ll c_2 \ll ... \ll c_h \ll ... \ll c_k$ (c_1 being the worst category, and c_k the best one). The assignment is done based on the comparison of the alternatives to external norms, rather than by comparison of the alternatives to each other.

Several approaches have been proposed to address such multicriteria sorting problem [22, 14]. We consider a well know multiple criteria sorting method, Electre Tri [13, 20, 23]. More precisely, we consider a variant of the Electre Tri method in line with the axiomatic work of Bouyssou and Marchant [1, 2]. This variant assigns alternatives using the alternatives' performances and preferential parameters of three types: profiles defining the category limits, weights specifying the importance of each criterion, and veto thresholds. To support specifying their preferences, we suppose that the DMs are able to provide assignment examples, i.e. alternatives (fictitious or real) associated to the categories the DMs think they belong to. Such assignment examples can correspond to past decision records or be expressed directly by DMs.

Most of the work on preference elicitation in Multicriteria Decision Aid focuses on representing the preferences of a single decision maker (DM). We are interested in elicitation procedures for multiple DMs that make it possible for each DM to provide individual preference information in order to build a multiple criteria sorting model accepted by each DM as representing the group preferences. We present linear programs able to find, on the basis of assignment examples provided by DMs, common profiles, shared among all the DMs, but allowing their weights (criteria importance factors) to vary individually. This can be used as a first step towards reaching an agreement on a preference model.

It is important to note that choosing a set of profiles may have different impacts on the different DMs, in terms of the remaining latitude on the weights they each may chose from. We therefore also propose a measure of the latitude that a DM has in setting her weights when a profile has been fixed.

In this paper, three linear and mixed integer programs solving the following problems are described.

ICL, or Infer Category Limits, finds, if possible, a set of profiles such that it is possible to satisfy the assignment examples of each DM using individual weights and majority threshold parameters without using veto thresholds.

ICLV, or Infer Category Limits with Vetoes, finds, if possible, a set of profiles such that it is possible to satisfy the assignment examples of each DM using individual weights and majority threshold parameters and using veto thresholds if necessary. This is a generalization of the first program but they are presented in order of increasing complexity.

CWR, or Compute Weights Restrictions, having fixed a set of shared profiles, computes a measure indicating the remaining latitude for each DM regarding their possibility of choosing the weights ordering on the criteria.

These inference programs may be used in a process aiming to build a consensus by progressively reaching a common preferential model. These tools should be combined with other decision aiding tools, therefore allowing to build a comprehensive decision aiding process.

The paper is structured as follows. Section 2 presents the sorting procedure used in this article and reviews related elicitation procedures. A decision aiding process illustrating the use of the proposed tools and their relation with existing ones is described in Section 3. The mathematical programs implementing the suggested tools are detailed in Sections 4 to 6. Section 7 examplifies the usage of the tools on an illustrative example. The algorithms performances are studied in Section 8, and conclusions and future perspectives are presented in Section 9.

2 Electre Tri and related elicitation issues

The inference procedures presented in this paper are based on a variant of the ELECTRE TRI sorting method. It uses the pessimistic assignment rule, without indifference or preference thresholds attached to criteria. Only a binary discordance condition is considered, i.e. a veto forbids an outranking in any possible concordance situation, or not. These simplifications, as compared to the original ELECTRE TRI procedure, permit to model the assignments of the procedure as mixed integer constraints and are in line with the axiomatized model of ELECTRE TRI proposed by Bouyssou and Marchant [1, 2].

2.1 Electre Tri method

We consider a finite set of alternatives A, a set of profiles $B = \{b_0, \dots b_k\}$, and a finite set of criteria $\{g_j, j \in J\}$. A criterion $g_j, j \in J$, is a function from $A \cup B$ to $\mathbb{R}$ where $g_j(a)$ denotes the performance of the alternative a on the criterion g_j. The alternatives have to be sorted in k categories $c_1, \dots, c_k$, ordered by their desirability (c_1 is the worst category, and c_k is the best one). Each category c_h is defined by the performances of its lower profile b_{h-1} and its upper profile b_h, with $b_{h-1}, b_h \in B$. The performances are supposed to be such that a higher value denotes a better performance, and the performances on the profiles are supposed to be non-decreasing, i.e. $\forall j \in J, 1 \leq h \leq k : g_j(b_{h-1}) \leq g_j(b_h)$.

To sort the alternatives, ELECTRE TRI defines an outranking relation $\succeq$ on the set of alternatives such that an alternative a outranks an alternative b if and only if a is considered at least as good as b. The pessimistic assignment rule assigns an alternative a to the highest possible category c_h such that the alternative outranks the category's lower profile b_{h-1}. An alternative a outranks a profile b_{h-1} if and only if the coalition of criteria in favor of the assertion "a is at least as good as b_{h-1}" forms a majority and no criterion strongly opposes (has a veto against) that assertion. The coalition of criteria in favor of $a \succeq b_{h-1}$, $\forall a \in A, 1 \leq h \leq k$, forms a majority iff

$$\sum_{j \in J} w_j C_j(a, b_{h-1}) \geq \lambda,$$

where w_j is the weight of criterion g_j (with $\sum_{j\in J} w_j = 1$), $C_j(a, b_{h-1}) \in \{0,1\}$, and $C_j(a, b_{h-1}) = 1 \Leftrightarrow g_j(a) \geq g_j(b_{h-1})$, 0 otherwise. The result of the sum of the weights of the criteria in support of the outranking is compared to a majority threshold $\lambda \in [0.5, 1]$ defined by the decision maker along with the weights. If the coalition is not a sufficient coalition, the alternative does not outrank the profile b_{h-1} and will therefore be assigned in a category below c_h.

Even when the coalition is strong enough, some criterion may veto the outranking situation. This happens if $g_j(a) \leq v_j^{h-1}$, where veto threshold v_j^{h-1} represents the performance below which alternative a is forbidden to outrank profile b_{h-1}, and thus is forbidden to be assigned to the category c_h. To summarize, the alternative a outranks the profile b_{h-1} (and therefore is assigned to at least the category c_h) if and only if $\sum_{j\in J} w_j C_j(a, b_{h-1}) \geq \lambda$ and $\forall j \in J : g_j(a) > v_j^{h-1}$.

In a case involving a single DM, the weights and majority thresholds (defining the sufficient coalitions) and the category limits (the profiles) may be given directly by her. However, this requires that the DM understands how these values will be used. It is moreover a difficult process to directly ask the DM for these parameters. The approach used here supposes that she provides assignment examples which are used to infer the preferential parameters.

The situation is even more complex when several DMs are involved. As is classical, it is assumed that the order of the categories, the criteria to use, the performances of the alternatives are consensual. There is no reason however to suppose that all DMs a priori agree on the importance of the criteria and the majority threshold to use or on the limits of the categories. Starting from assignment examples (not necessarily consensual) provided by each DM, this paper presents algorithms to support a group of DMs to reach a consensus concerning these preferential parameters.

2.2 Review of Electre Tri elicitation procedures

Previous works aiming to infer preferential parameters for the Electre Tri procedure on the basis of assignment examples exist, but they often involve a single DM. Mousseau and Slowinski [19] suggest to find the Electre Tri preference model parameters on the basis of assignment examples given by a decision maker, using non linear optimization. A linear program has been proposed in order to find the importance coefficients only, the other parameters being supposedly known [18]. In Ngo The and Mousseau [21], the parameters to be found by the model are the category limits, considering other parameters as fixed. This is done using a mixed integer formulation as well. In Dias et al. [11], the synergy between two approaches is discussed: the first approach suggests to infer the preference model parameters from assignment examples; the second one consists in computing robust assignments from a set of given constraints [12, 10]. When the DM provides an inconsistent set of assignment examples (i.e. assignment examples that do not match Electre Tri), Mousseau et al. [17] propose algorithms to compute which subset of assignment examples should be removed to restore consistency. This method has then been extended to relax assignments examples instead of removing them [16].

While the above approaches target a unique DM, Damart et al. [8] propose a method involving a group of DMs. This is done by iteratively constructing both individual preference models and a collective preference model representing the group consensus using informations given by the DMs on some assignment examples.

A summary of the main features of elicitation procedures proposed in the literature is proposed in Table 1. For each article, the second column indicates the expected input of the main tool proposed in the article, the last one shows its output. i designates assignment examples from a single DM, i^* designates possibly inconsistent assignment examples from a single DM, g is a group of DMs' assignment examples, $\mathcal{P}$ is a set of profiles evaluations, W is a set of weights. The computations are based on linear (or mixed integer and linear) programming, except for the first one.

3 Use of the proposed tools in a decision aiding process

The tools developed in this paper constitue only a part of the toolbox at an analyst's disposal to support a group of decision makers. Such toolbox includes other research results such as those summarized in Table 1. This section presents an example decision aiding process in order to illustrate where our algorithms apply and how they combine

Article	input	output
MS98 [19]	i	$\mathcal{P}$, W (non-linear)
MFN01 [18]	i, $\mathcal{P}$	W
NM00 [21]	i, W	$\mathcal{P}$
DMFC02 [11]	i	robust model ($\mathcal{P}$, W), robust assignments
MDFGC03 [17]	i^*	examples to remove to restore consistency
MDF06 [16]	i^*	examples to relax to restore consistency
DDM07 [8]	g, $\mathcal{P}$	progressive collective model (W)
ICL (this article)	g	collective model ($\mathcal{P}$)

Table 1: inference procedures for Electre Tri

with these other tools. This process targets obtaining a consensual group sorting model.

Our example process comprises two main parts. The first part deals with obtaining consensual group profiles, matching individual assignment examples when associated with individual weights and majority threshold. As a second part, starting from these profiles, consensual weights and majority threshold must be obtained in order to reach a group preference model.

The following steps may be followed to obtain consensual group profiles.

- Obtain individual assignment examples.
- Search for shared profiles with individual weights via the ICL program of this article.
- If no solution is found, possibly allow the search algorithm to return solutions including veto thresholds through the use of the ICLV program of this article. An other possibility is to use MDFGC03 (cf. Table 1) to suggest a change of assignment examples in order to remove inconsistencies, considering the group as an individual.
- When a solution is found, compute the remaining lattitude of each DM regarding the setting of the weights, i.e. to exclude a possibly too restrictive or too unfair solution, thanks to the CWR algorithm of this paper.

- Finally, ask the DMs if the choice of the profiles and the resulting constraints on their weights is satisfactory. If not, run the ICL or ICLV program again with supplementary constraints.

Once a solution has been found for consensual profiles, consensual weights and majority threshold must be obtained. Note that at this stage it is possible that the choice of profiles implies that no consensual weights may be found matching all assignment examples for all DMs. In this case, DDM07 could be applied (cf. Table 1). This method proposes to progressively build a consensual preference model representing a group of DMs preferences starting from consensual profiles values and assignment examples. This is done by progressively integrating some assignment examples into a "group" assignment examples set, possibly by asking some DMs to change their assignment examples when necessary. Describing the method in details is outside the scope of this paper and we refer the reader to the relevant article [8].

Note that it is also possible to search, as a single step, for shared profiles and consensual weights and majority thresholds respecting all assignment examples and other a priori constraints (the veto thresholds). This can be done using a simple modification of the ICL program, or of the ICLV program, which would use shared weights and majority thresholds instead of individual ones. If a consensual solution is found at this stage which satisfies all DMs, the problem is solved and there is no need to use the divide and conquer approach suggested here. However, it should be noted that it is possible that no consensual preference model exists able to reproduce all assignment examples, whereas finding consensual profiles with individual weights and majority thresholds is less constraining. Moreover, concentrating on first obtaining consensual profiles, then searching for the rest of the model parameters, simplifies the problem by dividing it into two easier problems. Finally, it should be remarked that no proposal for group decision aiding process using an Electre Tri model aiming at finding at the same time all the model parameters for the whole group on the basis of assignment examples has ever been published, to the best of our knowledge.

4 Inferring category limits

Having a set of alternatives A^* used as assignment examples, a set of criteria indices J, the evaluations of the alternatives $g_j(a), \forall a \in A^*, j \in J$, the number of categories k, a set of DMs $\mathcal{L}$, assignment examples E^l, the Infer Category Limits (ICL) program determines the performances of profiles $g_j(b_h), \forall j \in J, 1 \leq h \leq k-1$ shared among the DMs, together with individual weights w_j^l and majority thresholds λ^l, matching all assignment examples. For each DM $l \in \mathcal{L}$, the set of examples E^l is a set of pairs $(a,h), a \in A^*, 1 \leq h \leq k$ specifying that the alternative a is assigned to the category c_h by l. Supplementary to the three main types of decision variables, namely profiles, weights, majority thresholds, the MIP also defines the following variables. The binary variables $C_j(a,b_h), \forall a \in A^*, j \in J, 1 \leq h \leq k-1$ represent the partial concordance indices such that $C_j(a,b_h) = 1$ if and only if the performance of the alternative a on the criterion j is at least as good as the performance of the profile b_h. The continuous variables $\sigma_j(a,b_h)$ represent the weighted partial concordance indices, they are such that $\sigma_j(a,b_h) = w_j$ if $C_j(a,b_h) = 1$, $\sigma_j(a,b_h) = 0$ otherwise. Finally, a slack variable $s \geq 0$ is used in the objective function.

Hereafter we present a mathematical program finding adequate profiles without using veto thresholds. The case where veto thresholds are allowed is considered in Section 5.

4.1 Constraints

To make sure the solution matches the assignment examples, we must ensure the following is satisfied:

$$\forall l \in \mathcal{L} : \forall (a,h) \in E^l, h > 1 : \sum_{j \in J : g_j(a) \geq g_j(b_{h-1})} w_j^l \geq \lambda^l, \text{ and} \tag{1}$$

$$\forall l \in \mathcal{L} : \forall (a,h) \in E^l, h < k : \sum_{j \in J : g_j(a) \geq g_j(b_h)} w_j^l < \lambda^l. \tag{2}$$

Equations (1) ensure that the example alternative is assigned to a category at least as good as c_h, and (2) make sure that it is assigned to a category not better than c_h. In order to ensure this, variables $C_j(a,b_h)$ and $\sigma_j^l(a,b_h)$ are defined. Binary variables $C_j(a, b_h)$, indicating whether $g_j(a) \geq g_j(b_h)$ holds, are defined with the following constraints, $\forall j \in J, a \in A^*, 1 \leq h \leq k-1$.

$$\frac{1}{M}(g_j(a) - g_j(b_h)) < C_j(a, b_h) \leq \frac{1}{M}(g_j(a) - g_j(b_h)) + 1.$$

The constant M is an arbitrary big value ensuring that $-1 < \frac{1}{M}(g_j(a) - g_j(b_h)) < 1$. These constraints ensure that $g_j(a) \geq g_j(b_h) \Rightarrow C_j(a, b_h) = 1$, and that $g_j(a) < g_j(b_h) \Rightarrow C_j(a, b_h) = 0$. Indeed, if $g_j(a) - g_j(b_h) \geq 0$, $\frac{1}{M}(g_j(a) - g_j(b_h)) + 1 \geq 1$, thus $C_j(a, b_h) \leq \frac{1}{M}(g_j(a) - g_j(b_h)) + 1$ is necessarily satisfied (independently of the value of $C_j(a, b_h)$), and $0 \leq \frac{1}{M}(g_j(a) - g_j(b_h)) < 1$, thus $C_j(a, b_h)$ is constrained to one. Similarily, if $g_j(a) - g_j(b_h) < 0$, $-1 < \frac{1}{M}(g_j(a) - g_j(b_h)) < 0$ thus $\frac{1}{M}(g_j(a) - g_j(b_h)) < C_j(a, b_h)$ is necessarily satisfied, and $0 < \frac{1}{M}(g_j(a) - g_j(b_h)) + 1 < 1$ thus $C_j(a, b_h)$ is constrained to zero.

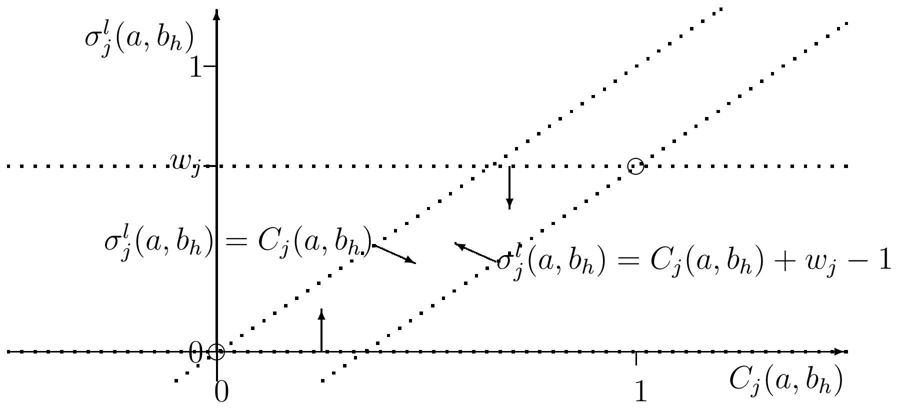
$$\forall l \in \mathcal{L}, j \in J, a \in A^*, 1 \leq h \leq k-1 : \begin{cases} \sigma_j^l(a, b_h) \leq w_j^l \\ \sigma_j^l(a, b_h) \geq 0 \\ \sigma_j^l(a, b_h) \leq C_j(a, b_h) \\ \sigma_j^l(a, b_h) \geq C_j(a, b_h) + w_j^l - 1. \end{cases} \tag{3}$$

Constraints (3) are used to define the variables $\sigma_j^l(a, b_h)$ representing the sum of the support for saying that a is at least as good as b_h, i.e. the value $\sigma_j^l(a, b_h) = w_j^l C_j(a, b_h)$, while avoiding using a non-linear expression [15]. The dotted parallelogram in Fig. 1 represents the possible values for $\sigma_j^l(a, b_h)$ depending on the value of $C_j(a, b_h)$ as defined by the constraints above. As $C_j(a, b_h)$ can only take values zero or one, the figure shows that $\sigma_j^l(a, b_h)$ is constrained to the appropriate value, namely zero or w_j.

At this point, referring to Equations (1) and (2), we have that $\sum_{j \in J} \sigma_j^l(a, b_h) = \sum_{j \in J : g_j(a) \geq g_j(b_h)} w_j^l$. To ensure satisfaction of the assignment examples, suffices thus to add the following constraints.

$$\forall l \in \mathcal{L}, \forall (a, h) \in E^l, h > 1 : \sum_{j \in J} \sigma_j^l(a, b_{h-1}) \geq \lambda^l, \tag{4}$$

$$\forall l \in \mathcal{L}, \forall (a, h) \in E^l, h < k : \sum_{j \in J} \sigma_j^l(a, b_h) < \lambda^l. \tag{5}$$



$\sigma_j^l(a,b_h)$
1
w_j
$\sigma_j^l(a,b_h) = C_j(a,b_h)$
$\sigma_j^l(a,b_h) = C_j(a,b_h) + w_j - 1$
0
0
1
$C_j(a,b_h)$

Figure 1: constraining $\sigma_j^l(a,b_h)$ to the appropriate value

Finally, the constraints $\sum_{j \in J} w_j = 1$, $\forall l \in \mathcal{L}$, are added, and the following constraints are used to ensure a correct ordering of the profiles defining the categories, $\forall j \in J, 2 \leq h \leq k-1 : g_j(b_{h-1}) \leq g_j(b_h)$.

4.2 Objective function

In order to maximize the separation between the sum of support and the majority thresholds, a slack variable s is introduced in Constraints (4) and (5) to be maximized as an objective function (see Program 1).

Having a set of alternatives A^*, a set of criteria indices J, the evaluations of the alternatives $g_j(a), \forall a \in A^*, j \in J$, the number of categories k, a set of DMs $\mathcal{L}$, assignment examples E^l, determine the performances of the profiles $g_j(b_h), \forall j \in J, 1 \leq h \leq k-1$, together with individual weights w_j^l and majority thresholds λ^l, maximizing s subject to the constraints provided in Program 1. Strict inequalities have been transformed to large inequalities using a constant value ε defined as an arbitrary small positive value. The constant M is an arbitrary big value. The variables $C_j(a,b_h), \forall j \in J, a \in A^*, 1 \leq h \leq k-1$, are binaries and the other ones are real.

5 Inferring category limits with vetoes

Suppose now that the DMs are ready to accept veto thresholds in the individual preference models, i.e. when searching for common profiles, it is deemed acceptable

$Max\ s$ s.t.

$$\left\{\begin{array}{ll} \forall l \in \mathcal{L}: & \left.\begin{array}{r} \sum_{j\in J} w_j^l = 1. \\ g_j(b_{h-1}) \leq g_j(b_h). \\ \frac{1}{M}((g_j(a) - g_j(b_h)) + \varepsilon) \leq C_j(a, b_h). \\ C_j(a,b_h) \leq \frac{1}{M}(g_j(a) - g_j(b_h)) + 1. \\ \left\{\begin{array}{l} \sigma_j^l(a,b_h) \leq w_j^l \\ \sigma_j^l(a,b_h) \geq 0 \\ \sigma_j^l(a,b_h) \leq C_j(a,b_h) \\ \sigma_j^l(a,b_h) \geq C_j(a,b_h) + w_j^l - 1. \end{array}\right. \end{array}\right\} (E) \\ \forall j \in J, 2 \leq h \leq k-1: & \\ \forall j \in J, a \in A^*, 1 \leq h \leq k-1: & \\ \forall j \in J, a \in A^*, 1 \leq h \leq k-1: & \\ \forall l \in \mathcal{L}, j \in J, a \in A^*, 1 \leq h \leq k-1: & \\ \forall l \in \mathcal{L}, \forall (a,h) \in E^l, h < k: & \sum_{j\in J} \sigma_j^l(a,b_h) + s \leq \lambda^l - \varepsilon. \\ \forall l \in \mathcal{L}, \forall (a,h) \in E^l, h > 1: & \sum_{j\in J} \sigma_j^l(a,b_{h-1}) \geq \lambda^l + s. \end{array}\right.$$

Program 1: ICL

to use shared veto thresholds to satisfy the individual assignment examples. This can enable to find common profiles in situations where it would not be possible to satisfy all examples without vetoes. This is the aim of the Infer Category Limits with Vetoes (ICLV) program.

This ICLV program can be seen as a generalization of the previous one, except for the change in the objective function, as it tries to find solutions having zero vetoes. However, we chose to present the programs in an increasing order of complexity. For simplicity, we also consider that the DMs share the vetoes. The same approach would be applicable for searching individual vetoes, with an objective function that could e.g. minimize the number of individual vetoes used or minimize the number of DMs using some vetoes. Sharing the vetoes also reduces the number of binary variables and reduces the risk that the resulting preference model would overfit the provided data.

The mathematical program is based on the previous one, with a few additions and changes. The veto situations are modeled as follows: a veto threshold $v_j^{b_h}$ (a variable in our problem) is associated with each criterion $j \in J$ and profile index $1 \leq h \leq k-1$. We also need binary variables $V_j(a, b_h), \forall j \in J, a \in A^*, 1 \leq h \leq k-1$, equal to one iff there is a veto situation between a and b_h. When for any criterion and profile the evaluation $g_j(a)$ is lower than or equal to the corresponding $v_j^{b_h}$ veto threshold, the alternative may not outrank the profile.

5.1 Additional constraints

Constraints are used to keep the vetoes lower than the corresponding profile: $\forall j \in J, 1 \leq h \leq k-1, v_j^{b_h} \leq g_j(b_h)$. The veto thresholds also have to be correctly ordered: $\forall j \in J, 2 \leq h \leq k-1, v_j^{b_{h-1}} \leq v_j^{b_h}$. Binary variables $V_j(a, b_h)$ are defined so that $V_j(a, b_h)$ equals one iff $g_j(a) < v_j^{b_h}$, $\forall j \in J, a \in A^*, 1 \leq h \leq k-1:$

$$\frac{v_j^{b_h} - g_j(a) + \varepsilon}{M} \leq V_j(a, b_h) \leq \frac{v_j^{b_h} - g_j(a)}{M} + 1.$$

Constraints (4), (5) must be redefined to take the vetoes into account when matching assignment examples.

$$\forall l \in \mathcal{L}, \forall (a, h) \in E^l, h < k : \sum_{j \in J} \sigma_j^l(a, b_h) < \lambda^l + \sum_{j \in J} V_j(a, b_h), \tag{6}$$

$$\forall l \in \mathcal{L}, \forall (a, h) \in E^l, h > 1 : \sum_{j \in J} \sigma_j^l(a, b_{h-1}) \geq \lambda^l + \sum_{j \in J} V_j(a, b_{h-1}), \tag{7}$$

where $\sum_{j \in J} V_j(a, b_h)$ accounts for the existence of a veto situation between a and b_h. Indeed in constraint (6), the term $\sum_{j \in J} V_j(a, b_h)$ guarantees that a does not outrank b_h when a veto intervenes, whatever the coalition of criteria in favor of such outranking. A similar reasoning holds for constraint (7).

To minimize the number of vetoes used, a binary variable V_j is defined for each criterion. The variable equals one iff a veto is used for this criterion:

$$\forall j \in J, a \in A^*, 1 \leq h \leq k-1 : V_j \geq V_j(a, b_h).$$

$Min \sum_{j \in J} V_j$ s.t.

$$\begin{cases} \text{set } (E) \text{ of constraints from Program 1} & \\ \forall l \in \mathcal{L}, \forall (a,h) \in E^l, h < k: & \sum_{j \in J} \sigma_j^l(a, b_h) \leq \lambda^l + \sum_{j \in J} V_j(a, b_h) - \varepsilon. \\ \forall l \in \mathcal{L}, \forall (a,h) \in E^l, h > 1: & \sum_{j \in J} \sigma_j^l(a, b_{h-1}) \geq \lambda^l + \sum_{j \in J} V_j(a, b_{h-1}). \\ \forall j \in J, 1 \leq h \leq k-1: & v_j^{b_h} \leq g_j(b_h). \\ \forall j \in J, 2 \leq h \leq k-1: & v_j^{b_{h-1}} \leq v_j^{b_h}. \\ \forall j \in J, a \in A^*, 1 \leq h \leq k-1: & V_j(a, b_h) \geq \frac{v_j^{b_h} - g_j(a) + \varepsilon}{M}. \\ \forall j \in J, a \in A^*, 1 \leq h \leq k-1: & V_j(a, b_h) \leq \frac{v_j^{b_h} - g_j(a)}{M} + 1. \\ \forall j \in J, a \in A^*, 1 \leq h \leq k-1: & V_j \geq V_j(a, b_h). \end{cases}$$

Program 2: ICLV

5.2 Objective function

In most situations it is reasonable to find a solution involving the least possible number of vetoes: allowing too many veto thresholds to be used may lead to an over-fitting of the model with ad-hoc veto thresholds. The objective function should then be to minimize the sum of the V_j variables:

$$Min \sum_{j \in J} V_j.$$

An alternative could be e.g. to minimize the number of situations where a veto is used.

Program 2 presents a synthesis of the ICLV program.

6 Computing weights restriction

For each DM, fixing the common profiles induces constraints on her weights. Indeed, once the profiles are inferred by ICL (or ICLV), her assignment examples express a set

W^l of constraints on the weights w^l and majority threshold λ.

$$W^l = \{(w_j, j \in J, \lambda)\} \mid \forall (a, h) \in E^l : a \succeq b_{h-1} \wedge \neg a \succeq b_h\}.$$

It is possible to define, on the basis of W^l, an importance relation $\rhd^l$ on J as follows, $\forall j_1, j_2 \in J$:

$$j_1 \rhd^l j_2 \Leftrightarrow w_{j_1} > w_{j_2}, \forall (w_j, j \in J, \lambda) \in W^l.$$

This relation $\rhd^l$ expresses, for a given DM l, the comparisons of criteria weights which hold for all weight vectors compatible with her assignment examples, considering the common profiles. It results from its definition that $\rhd^l$ is a partial order. The incompleteness of this relation should be understood in the following way: pairs of criteria (j_1, j_2) that are not in this relation correspond to criteria such that there exists $(w_j, j \in J, \lambda) \in W^l$ with $w_{j_1} > w_{j_2}$ and $(w'_j, j \in J, \lambda') \in W^l$ with $w'_{j_2} > w'_{j_1}$. In such a case, W^l does not specify how w_{j_1} and w_{j_2} compare.

The Compute Weights Restriction (CWR) program aims at computing $\rhd^l$ for each DM $l \in \mathcal{L}$. In order to compute the relation $\rhd^l$, we search, for each $j_1, j_2 \in J$, for a feasible solution to a linear program matching all assignment examples of DM l and with a supplementary constraint $w^l_{j_1} \leq w^l_{j_2}$. This is a linear program involving no binary variables because at this stage the profiles values are known. If that linear program has no feasible solution, criterion j_1 is more important than criterion j_2, $\forall (w_j, j \in J, \lambda) \in W^l$, thus $j_1 \rhd^l j_2$. On the contrary, if the linear program has a feasible solution, $j_1 \rhd^l j_2$ does not hold.

Once the common profiles have been computed using ICL or ICLV programs, it is necessary that each DM accepts the weight restrictions induced by the choice of the common profiles. The relation $\rhd^l$ is presented to each DM l for validation. Although the validation of $\rhd^l$ by the DM l does not guarantee the acceptance of W^l, we consider that, if $\rhd^l$ is validated by the DM, she accepts the constraints on weights W^l imposed by the choice of the profiles. Indeed, it would be illusory to ask the DM to validate W^l per se.

If for a pair of criteria j_1, j_2 such that $j_1 \rhd^l j_2$, the DM considers that criterion j_2 is more important than j_1, it is necessary to reconsider the choice of the profiles and recompute new profiles using ICL or ICLV programs including a new constraint: $w^l_{j_2} > w^l_{j_1}$.

7 Illustrative example

Let us illustrate the method on the following hypothetical scenario. (Another illustrative example in a risk setting context can be found in [7].) A government board has the responsibility to choose which research project to finance among a list of research proposals. This board wants to establish a systematic procedure to be able to assign research proposals into three categories: those projects that are considered very good and should be funded (category *Good*); projects that are good and should be funded if supplementary budget can be found (category *Average*); projects that are of low quality and should not be funded (category *Bad*). The four members of the board agree to use the following six criteria.

sq The project's scientific quality, evaluated on a 5-point ordinal scale.

wq The proposal's writing quality, evaluated on a 5-point ordinal scale.

ad The proposal's adequacy with respect to the government priorities, evaluated on a 3-point ordinal scale.

te The experience of the researcher teams submitting the project, evaluated on a 5-point ordinal scale.

ic Whether the proposal includes international collaboration, a binary assessment.

ps The researchers' publication score evaluated by an aggregate measure of the total quality of publications of the researchers involved in the proposal (evaluated on a [0,100] scale).

We suppose that each DM provides 30 assignment examples (that could correspond to existing projects). These examples have been generated using Electre Tri with profiles shared by the four DMs, each of them having an individual set of weights and cutting level, and no veto was involved. The Electre Tri parameters used to generate the assignment examples are provided in Tables 2 and 3 and the complete list of assignment examples is provided in [6].

Using the assignment examples, the ICL program is used to find profiles shared by the DMs which match their individual assignment examples. The results are provided

Profile	sq	wq	ad	te	ic	ps
b_1	3	3	2	3	1	40
b_2	4	4	3	4	1	80

Table 2: Profiles used to generate assignment examples.

DM	sq	wq	ad	te	ic	ps	λ
dm1	0.4	0.2	0.1	0.1	0.1	0.1	0.61
dm2	0.3	0	0.3	0	0.4	0	0.71
dm3	0.1	0	0.1	0.2	0.3	0.3	0.51
dm4	0.4	0	0	0.1	0.1	0.4	0.81

Table 3: Weights used to generate assignment examples, per DM.

in Table 4, where the profiles performances values corresponding to criteria using an integer scale have been rounded up. Because the alternatives also have integer performance values on these criteria, doing so does not impact the way each alternative compares to these profiles. Table 5 shows for each DM a possible set of weights matching the assignment examples with the common profiles.

Note that the inferred common profiles are very similar to the ones used for the data generation. This shows that the inference program is able to reproduce the profiles from which the assignment examples were derived.

It is now important to compute, for each DM, the constraints on the weights imposed by the choice of the common profiles. These restrictions as found by the CWR program are provided in Fig. 2.

These importance relations are to be discussed and accepted by all DMs for these shared profiles to be accepted. If one DM disagree on some importance comparisons,

Profile	sq	wq	ad	te	ic	ps
b_1	3	2	2	3	1	40.7
b_2	4	4	3	4	1	81.8

Table 4: Inferred profiles.

DM	sq	wq	ad	te	ic	ps	λ
dm1	0.375	0.25	0	0	0.25	0.125	0.688
dm2	0.75	0	0.125	0	0.125	0	0.938
dm3	0.125	0	0.125	0.25	0.25	0.25	0.563
dm4	0.875	0	0	0	0	0.125	0.938

Table 5: A set of weights, as found by the ICL program, matching assignment examples of each DM when used together with the inferred profiles.

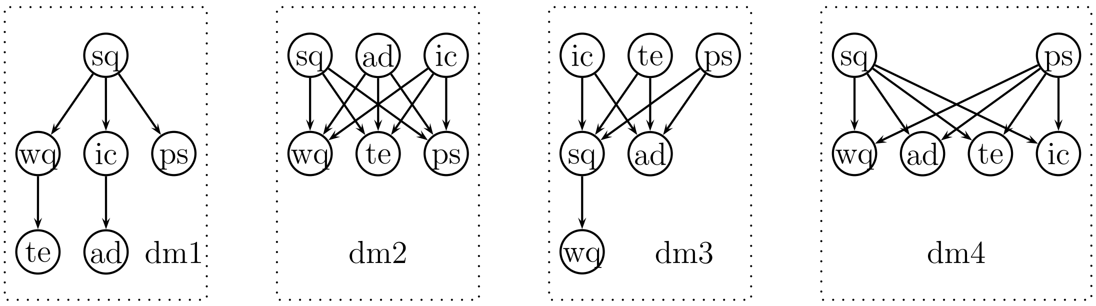


Figure 2: The restrictions on the weights imposed by the choice of the common profiles, for each DM. An arrow from a criterion j to a criterion j' states that the choice of these profiles implies a greater weight for j than for j' for this DM.

the ICL program should be run again with some additional constraints concerning unacceptable weights comparisons.

It may be seen that a consensus on the weights cannot be readily found using these profiles as e.g. DM 3 has criterion ic necessarily more important than criterion sq while DMs 1 and 4 have the inverse preference. To get closer to a consensus, the third DM might be asked if he could consider criterion sq to be more important than criterion ic.

Had no solution been found at some point, the ICLV program could be used instead, as allowing vetoes increase the chances that a shared profile set be found. Once shared profiles are found, even facing an impossibility of an immediate consensus on the weights as illustrated here, the method proposed by Damart et al. [8] may be used to iteratively build consensual weights among the group of DMs.

8 Scalability of the algorithms

The proposed algorithms have been implemented on top of the Java J-MCDA libraries [3, 4] and use the JLP library [5] to define the mathematical programs, which are then solved using the Ilog CPLEX library version 12.2. The Java wrapper makes a future integration into the Decision Deck project [9] easy and involves no performance impact as the mathematical programs are solved in pure C thanks to the Ilog concert technology. The tests have been run on an Intel Core2 Duo E8400 3 GHz PC with 2 GB RAM running a GNU/Linux Debian Lenny system.

In order to assess the performance and scalability of the proposed mathematical programs, the ICL and CWR programs have been run on a series of test problems. So as to get consistent results and observe the behaviour of the programs according to the problem size, about 2000 problems have been generated of various sizes chosen in the following ranges: number of criteria between three and ten, number of DMs between one and four, number of categories between two and five (implying that one to four category limits have to be found). The number of examples given by each DM vary between 1 and 700. Each criterion has an integer scale between 0 and 99.

Once the problem size is fixed, we generate category limits which we suppose all DMs share. The category limits performances divide evenly the space of possible parameters into the categories, e.g. with four categories the profiles will have a performance on every criterion of 25 for the worst profile, 50 for the middle one, and 75 for the best one. Each DM is also assigned a random set of weights. We also generate random alternatives, i.e. alternatives having their performances on each criterion set randomly with an uniform distribution. The assignment examples are obtained by computing, for each DM, the assignment of the alternatives into the defined categories. Then the ICL and CWR programs are used with the resulting data. Note that by construction, there exists a set of shared profiles satisfying assignment examples of all DMs.

We observe the proportion of ICL programs that are solved within a time limit of 90 minutes (a duration compatible with an off-line use of the program) and a disk space usage limit of 3 GB (representing the size of the tree structure used by the solver). The proportion of problems that can be solved within these limits obviously depends on the size of the problem. Reported in Table 6 is the proportion of problems solved

Number of binary variables	Number of instances	Solved
[0, 399]	477	100%
[400, 799]	441	87%
[800, 1199]	362	80%
[1200, 1599]	290	78%
[1600, 1999]	268	75%
[2000, 2199]	121	69%

Table 6: The proportion of problems solved within the given resource limits (less than 90 minutes of computation, max 3 GB disk space usage) according to the number of binary variables contained in those problems. The number of instances indicates the number of problems generated, i.e. that we tried to solve.

using the ICL program, according to the number of binary variables involved.

Recall that the number of binary variables in the ICL problem corresponds to the product of the number of criteria, the number of profiles to be found, and the number of alternatives in the assignment examples set. Data sets involving a MIP formulation with less than 400 binary variables to solve ICL are all solved within 90 minutes. Such data sets correspond to relatively small problems, e.g. with 3 categories (2 profiles), 5 criteria and 40 assignment examples (10 examples for each DM if 4 DMs are involved). Moreover, a large majority of instances are solved within 90 minutes when the number of binary variables does not exceed 1200. It is likely that most practical sized problems will contain a number of binary variables below 1200, this number corresponding to a problem containing 6 criteria, 3 categories, 100 alternatives in the examples set.

Furthermore, we observed that the ability to solve the problem also depends on the problem structure, and in particular on the number of criteria, for a given number of binary variables. Indeed, all problems with no more than 5 criteria have been solved in our experiment, even those containing the biggest number of binary variables (in the range [2000, 2199]). On the opposite, when the number of criteria is between 8 and 10, less than 50% of the instances involving 1200 binary variables can be solved within 90 minutes. Table 7 shows the proportion of problems solved according to the number of criteria.

Nb of criteria	Nb of binary variables	Nb of instances	Solved
[3, 5]	[0, 2199]	714	100%
[6, 7]	[0, 399]	112	100%
[6, 7]	[400, 799]	108	96%
[6, 7]	[800, 1199]	93	96%
[6, 7]	[1200, 1599]	84	96%
[6, 7]	[1600, 1999]	80	95%
[6, 7]	[2000, 2199]	24	96%
[8, 10]	[0, 399]	184	99%
[8, 10]	[400, 799]	179	70%
[8, 10]	[800, 1199]	132	48%
[8, 10]	[1200, 1599]	100	39%
[8, 10]	[1600, 1999]	97	36%
[8, 10]	[2000, 2199]	52	29%

Table 7: The proportion of problems solved within the given resource limits (less than 90 minutes of computation, max 3 GB disk space usage) according to the number of binary variables and criteria contained in those problems. The number of instances indicates the number of problems generated, i.e. that we tried to solve.

The CWR program performances were also assessed by running it for each test where profiles have been found. The time to compute all the weights restrictions has been calculated. At most a few minutes are needed in the most difficult cases to compute all the restrictions, and in most cases even only a few seconds. This is so because, although CWR involves solving a lot of linear programs, these LPs are solved almost instantaneously.

These data show that the ICL program is able to solve most problems having a reasonable size within 90 minutes (using less than 3 GB of disk space). As the approach we propose is typically used in an off-line mode where the analyst analyses the preferences indicated by the DMs in the interval between two meetings with them, a bound of 90 minutes might be considered a reasonable time to solve such complex problems. Bigger problems (involving about 10 criteria) might solve if given more time or space, although this has not yet been investigated. A probably better approach to solve very big problems would consist in developing appropriate resolution methods taking into account the structure of the problem. This is an interesting area to explore for further applicability of these algorithms.

9 Conclusions and perspectives

This paper deals with group preference elicitation. We aim at eliciting an Electre Tri sorting model for a group of DMs on the basis of assignment examples provided by the various DMs. More precisely, we propose algorithms to elicit Electre Tri category limits shared by all DMs from their assignment examples. The proposed algorithms consider the case with or without vetoes. We infer a part of the sorting model only (category limits, but with variable weights): such partial inference provides a better control of the inference process, in particular when multiple DMs provide assignment examples. We illustrate the use of these algorithms on a practical example. Numerical tests of these algorithms show that a majority of instances with a size corresponding to real-world problems can be solved in a reasonable time.

This work opens new research challenges and issues. Instances involving more than eight criteria remain difficult to solve. Specific resolution algorithms should be investigated to be able to solve all large instances. Our algorithm should be adapted in order to select a set of profiles which minimizes the restriction on the weights imposed to the DMs. When the assignment examples provided by the DMs are not compatible with an Electre Tri model using shared profiles, an interesting issue amounts at identifying ways to modify the assignment examples in order to find common profiles [17, 16]. In such case, it would also be interesting to identify subsets of DMs who are closer to a consensus. Another interesting extension to the proposed algorithms involve considering confidence levels attached to the assignment examples provided by the DMs. From a practical point of view an important question deals with the choice of the alternatives involved in the assignment examples: how to choose alternatives that will lead to informative assignment examples? Our algorithms provide one single solution defining the profiles. It would be interesting to compute all sets of category limits compatible with the assignment examples (rather than one solution) using a robust disaggregation approach [11].

Finally, an implementation of the tools described here should be made available in the Decision Deck framework as open source software. That would permit to test the suggested approach on real cases, which would certainly raise further new ideas and suggestions for improvements.

References

[1] Bouyssou, D. and Marchant, T. (2007a). An axiomatic approach to noncompensatory sorting methods in MCDM, I: the case of two categories. *European Journal of Operational Research*, 178(1):217–245.

[2] Bouyssou, D. and Marchant, T. (2007b). An axiomatic approach to noncompensatory sorting methods in MCDM, II: more than two categories. *European Journal of Operational Research*, 178(1):246–276.

[3] Cailloux, O. (2010). ELECTRE and PROMETHEE MCDA methods as reusable software components. In *Proceedings of the 25th Mini-EURO Conference on Uncertainty and Robustness in Planning and Decision Making (URPDM 2010)*, Coimbra, Portugal. University of Coimbra, Portugal.

[4] Cailloux, O. (2011). J-MCDA: free/libre java libraries for MCDA. `http://sourceforge.net/projects/j-mcda/`.

[5] Cailloux, O. and Lukasiewycz, M. (2011). JLP: a Java interface to integer linear programming solvers. `http://code.google.com/p/jlp-interface/`.

[6] Cailloux, O., Meyer, P., and Mousseau, V. (2011). Eliciting electre tri category limits for a group of decision makers. Technical report, Laboratoire Génie Industriel, Ecole Centrale Paris. Cahiers de recherche 2011-09.

[7] Cailloux, O. and Mousseau, V. (2011). Parameterize a territorial risk evaluation scale using multiple experts knowledge through risk assessment examples. In Bérenguer, C., Grall, A., and Guedes Soares, C., editors, *ESREL, Conference of the European Safety and Reliability Association*, pages 2331–2340. Sept. 18-22, 2011, Troyes, France.

[8] Damart, S., Dias, L., and Mousseau, V. (2007). Supporting groups in sorting decisions: Methodology and use of a multi-criteria aggregation/disaggregation DSS. *Decision Support Systems*, 43(4):1464–1475.

[9] Decision Deck Consortium (2011). The Decision Deck project. `http://www.decision-deck.org/`.

[10] Dias, L. and Clímaco, J. (2000). ELECTRE TRI for groups with imprecise information on parameter values. *Group Decision and Negotiation*, 9(5):355–377.

[11] Dias, L., Mousseau, V., Figueira, J., and Clímaco, J. (2002). An aggregation/disaggregation approach to obtain robust conclusions with ELECTRE TRI. *European Journal of Operational Research*, 138(2):332–348.

[12] Dias, L. C. and Climaco, J. (1999). On computing ELECTRE's credibility indices under partial information. *Journal of Multi-Criteria Decision Analysis*, 8(2):74–92.

[13] Figueira, J., Mousseau, V., and Roy, B. (2005). ELECTRE methods. In Figueira, J., Greco, S., and Ehrgott, M., editors, *Multiple Criteria Decision Analysis: State of the Art Surveys*, pages 133–162. Springer Verlag, Boston, Dordrecht, London. Chapter 4.

[14] Greco, S., Matarazzo, B., and Slowinski, R. (2002). Rough sets methodology for sorting problems in presence of multiple attributes and criteria. *European Journal of Operational Research*, 138:247–259.

[15] Meyer, P., Marichal, J., and Bisdorff, R. (2008). Disaggregation of bipolar-valued outranking relations. In An, L. T. H., Bouvry, P., and Tao, P. D., editors, *Proc. of MCO'08 conference*, pages 204–213, Metz, France. Springer.

[16] Mousseau, V., Dias, L., and Figueira, J. (2006). Dealing with inconsistent judgments in multiple criteria sorting models. *4OR*, 4(3):145–158.

[17] Mousseau, V., Dias, L., Figueira, J., Gomes, C., and Clímaco, J. (2003). Resolving inconsistencies among constraints on the parameters of an MCDA model. *European Journal of Operational Research*, 147(1):72–93.

[18] Mousseau, V., Figueira, J., and Naux, J. (2001). Using assignment examples to infer weights for ELECTRE TRI method: Some experimental results. *European Journal of Operational Research*, 130(2):263–275.

[19] Mousseau, V. and Slowinski, R. (1998). Inferring an ELECTRE TRI model from assignment examples. *Journal of Global Optimization*, 12(2):157–174.

[20] Mousseau, V., Slowinski, R., and Zielniewicz, P. (2000). A user-oriented implementation of the ELECTRE TRI method integrating preference elicitation support. *Computers & Operations Research*, 27(7-8):757–777.

[21] Ngo The, A. and Mousseau, V. (2002). Using assignment examples to infer category limits for the ELECTRE TRI method. *JMCDA*, 11(1):29–43.

[22] Perny, P. (1998). Multicriteria filtering methods based on Concordance/Non-Discordance principles. *Annals of Operations Research*, 80:137–167.

[23] Roy, B. (1991). The outranking approach and the foundations of ELECTRE methods. *Theory and Decision*, 31:49–73.