CAHIER DU LAMSADE

Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision
(Université Paris-Dauphine)
Unité de Recherche Associée au CNRS ESA 7024

RESOLVING INCONSISTENCIES AMONG CONSTRAINTS ON THE PARAMETERS OF AN MCDA MODEL

CAHIER N° 178
avril 2001

João CLÍMACO [1 2]
Luis DIAS [1 2]
José FIGUEIRA [2 4]
Carlos GOMES da SILVA [3]
Vincent MOUSSEAU [4]

received: January 2001.

[1] *Instituto de Engenharia de Sistemas e Computadores-Coimbra, Rua Antero de Quental, 199, 3000-033 Coimbra, Portugal ({ldial,jclimaco}@inescc.pt).*

[2] *Faculdade de Economia, Universidade de Coimbra, Av. Dias da Silva, 165, 3004-512 Coimbra, Portugal.*

[3] *Escola Superior de Tecnologia e Gestão, Instituto Politécnico de Leiria, 2401-951 Leiria, Portugal (cgsilva@estg.iplei.pt).*

[4] *LAMSADE, Université Paris-Dauphine, Place du Maréchal De Lattre de Tassigny, 75775 Paris Cedex 16, France ({mousseau,figueira}@lamsade.dauphine.fr).*

Table of Contents

Resolving inconsistencies among constraints on the parameters of an MCDA model

ABSTRACT: We consider a framework where Decision Makers (DMs) interactively define a multicriteria evaluation model by providing imprecise information (i.e., a linear system of constraints to the model's parameters) and by analyzing the consequences of the information provided. DMs may introduce new constraints explicitly or implicitly (results that the model should yield). If a new constraint is incompatible with the previous ones, then the system becomes inconsistent and the DMs must choose between removing the new constraint and removing some of the older ones. We address the problem of identifying subsets of constraints which, when removed, lead to a consistent system. The identification of such subsets is extremely useful because they state the reason for the inconsistent information given by DMs. There may exist several possibilities for the DMs to choose from, in order to resolve the inconsistency. We present some approaches based on mathematical programming to identify such possibilities and an application to an aggregation/disaggregation procedure for the ELECTRE TRI method.

Keywords: Inconsistent Linear Systems, Multiple Criteria Analysis, ELECTRE TRI, Aggregation/Disaggregation Approach, Imprecise Information.

Résolution des incohérences dans un système de contraintes sur les paramètres d'un modèle multicritère

RESUME: Nous considérons un contexte dans lequel le décideur définit de façon interactive un modèle d'évaluation multicritère en spécifiant une information imprécise (i.e., un système de contraintes linéaires sur paramètres du modèle) et en analysant les conséquences de l'information fournie. Le décideur peut introduire de nouvelles contraintes explicitement ou implicitement (sous la forme de résultats que le modèle doit reproduire). Si une nouvelle contrainte est incompatible avec les précédentes, le système devient incohérent et le décideur doit soit retirer la nouvelle contrainte, soit certaines parmi les anciennes. Nous nous intéressons au problème de l'identification de sous-ensembles de contraintes dont la suppression rend le système cohérent. L'identification de tels sous-ensembles est extrêmement utile car elle permet d'identifier la source de l'incohérence dans l'information fournie par le décideur. Il peut exister plusieurs sous-ensembles de ce type ; le décideur doit choisir parmi eux une manière de résoudre l'incohérence. Nous présentons des approches basées sur la programmation mathématique pour identifier ces sous-ensembles ainsi qu'une application dans le cadre d'une procédure d'agrégation-désagrégation pour la méthode ELECTRE TRI.

Mots clés: Systèmes linéaires incohérents, Analyse Multicritère, ELECTRE TRI, Approche par agrégation/désagrégation, Information imprécise.

1. Introduction

Multicriteria decision aiding models usually have many preferential parameters that the decision makers (DMs) must set. These parameters influence, namely, the manner in which differences in performances are valued, and the role of each criterion in the aggregation of the performances. Providing precise figures for all parameter values is often difficult, to the extent that there may exist some imprecision, contradiction, arbitrariness, and/or lack of consensus concerning the value of the parameters (Roy & Bouyssou, 1989).

We will consider an imprecise information context (see, e.g., Weber, 1987; Dias & Clímaco, 2000), where the DMs may indicate some constraints on the acceptable combinations of parameter values. Such information may be provided in an explicit manner (e.g., parameter t_1 belongs to $[0.2, 0.3]$, or parameter t_1 is larger than parameter t_2), or in an implicit manner (indicating a result that the model should restore, e.g., alternative a_1 should be better ranked than a_2). Methods that accept the latter type of constraints to infer parameter values are often called aggregation/disaggregation procedures (Jacquet-Lagrèze & Siskos, 1982; Mousseau & Slowinski, 1998; Nadeau et al., 1991).

In the course of an interactive process, DMs may progressively add constraints on the parameter values. Let T_k denote the set of parameter values that are acceptable to the DMs (according to the constraints they provided) at the k-th iteration. Given this set, it is possible to provide some output to support the DMs in revising T_k:

- robust conclusions – the results that are valid for all the combinations $t \in T_k$ (Roy, 1998; Vincke, 1999); for instance, "a_1 is never contained in the choice set" in a selection problem; "a_1 is always better ranked than a_2" in a ranking problem; or "a_1 can only be assigned to category "good" or "very good" in a sorting problem;
- variability information – the results that vary more, according to the combination chosen; for instance, "the position of a_1 in the ranking is very unstable: for some input values it may be the best, whereas for other combinations it is one of the worst" in a ranking problem (Kämpke, 1996);
- inferred parameter values procedures (Jacquet-Lagrèze & Siskos, 1982; Mousseau & Slowinski, 1998) – a "central" combination $t \in T_k$ that satisfies all the constraints, hence able to restore the results that were demanded.

We consider interactive processes in which DMs start the first iteration with very little information. Each iteration will provide an opportunity to add, delete or modify a specific supplementary constraint. Adding a single piece of information at each iteration facilitates the control of the information supplied by the DMs. This interactive process stops when DMs are satisfied and the set T_k as well as the results of the model match their view of the decision problem.

We will consider that all the constraints are linear (T_k is a polyhedron) and that the polyhedron T_{k+1} that corresponds to the next iteration is obtained by adding a single constraint, i.e., by intersecting T_k with a half-space or a hyperplane. A difficulty occurs when T_{k+1} becomes empty, meaning that the new constraint contradicts some of the previous ones. To resolve the inconsistency that appeared in the linear system of constraints, one must either drop the new constraint, or some of the older ones. The choice should belong to the DMs, after they learn which are the sets of constraints that lead to a non-empty T_{k+1} if removed. Notice that when we refer to the removal of one or more constraints, the DMs may choose to relax these constraints instead (e.g., increasing the right-hand side of an $Ax \leq b$ system).

Many authors have previously addressed the subject of infeasibility analysis in linear programming (see, J.W. Chinneck, (1997) for a complete summary of the state of the art in infeasibility analysis algorithms) according to different perspectives, namely:

1. Some authors (Loon, 1981; Chinneck, 1994; Tamiz *et al.*, 1996) are interested in determining an Irreducibly Inconsistent System (IIS). An IIS is a subset of constraints that corresponds to an inconsistent system, which is minimal, in the sense that any proper subset of an IIS is a consistent system. Let us remark that the inconsistency in an IIS can be removed by deleting any constraint, but if there are other IISs, then the initial system of constraints can remain inconsistent.
2. A different problem is to determine the minimum number of constraints that has to be removed to restore the consistency in the initial system, which is equivalent to solve the minimum-cardinality IIS set-covering problem (Chinneck, 1996; Murty *et al.*, 2000).
3. Finally, we can mention the problem of determining the minimum weight (or cost) alternative to restore the consistency in a system, which is equivalent to determine a minimum-weight IIS set-cover (Chinneck, 1996; Murty *et al.*, 2000).

The perspective we are interested in is close to problem 2, with the following specificities:

- we are also interested in sets of constraints that restore the consistency if removed that are *not* of minimum cardinality, since the DMs may rather drop two constraints they consider unimportant than drop a single important one;
- we know that one of the constraints caused the inconsistency, hence removing that constraint is a trivial manner to resolve the inconsistency; the question here is what other alternatives exist.

Hence, we may formulate the problem we are addressing as: to determine the p "smallest" sets of constraints (in terms of cardinality) that, if removed, restore the consistency to the initial system.

In the context of the interactive processes we are considering, solving such problems will allow us to propose alternative ways to resolve an inconsistency that appeared at a given iteration. This helps the DMs to understand how their inputs are conflicting and to question previously expressed judgments. Analyzing and confronting the alternative solutions of such problems provide opportunities for the DMs to learn about their preferences as the interactive process evolves.

In the next section we define our problem formally and propose two techniques to solve it. One of the techniques consists in solving a succession of mixed-integer linear programs, while the second one uses only linear programming. Section 3 presents an application to the aggregation/disaggregation approach for ELECTRE TRI, reviewing this approach and including a numerical example. Section 4 indicates some extensions and concludes the paper.

2. Two different methods to cope with inconsistent systems

Consider a problem in which the DM has interactively specified constraints on the parameters by defining a polyhedron of acceptable values denoted by T_{k-1} (at iteration k-1). This polyhedron is defined by the following (general) consistent system of $m-1$ linear constraints on n variables $x_1, ..., x_n$:

$$\begin{cases} \sum_{j=1}^{n} \alpha_{1j} x_j \geq \beta_1 \\ \vdots \\ \sum_{j=1}^{n} \alpha_{(m-1)j} x_j \geq \beta_{(m-1)} \end{cases} \qquad (\alpha_{11},\ldots,\alpha_{(m-1)n} \in \mathrm{R}, \beta_1,\ldots,\beta_{(m-1)} \in \mathrm{R}) \qquad (1)$$

Let $\sum_{j=1}^{n} \alpha_{mj} x_j \geq \beta_m$ be a new constraint that, when added to the system (1), originates a system (2) of m linear constraints, which is now inconsistent:

$$\begin{cases} \sum_{j=1}^{n} \alpha_{1j} x_j \geq \beta_1 \\ \vdots \\ \sum_{j=1}^{n} \alpha_{(m-1)j} x_j \geq \beta_{(m-1)} \\ \sum_{j=1}^{n} \alpha_{mj} x_j \geq \beta_m \end{cases} \qquad (\alpha_{11},\ldots,\alpha_{mn} \in \mathrm{R}, \beta_1,\ldots,\beta_m \in \mathrm{R}) \qquad (2)$$

Let $I=\{1,\ldots,m\}$ be the set of indices of the constraints defining T_k (at iteration k, i.e., with the new constraint that makes T_k empty). Hence,

$$T_k = \left\{ x \in \mathrm{R}^n : \sum_{j=1}^{n} \alpha_{ij} x_j \geq \beta_i, \forall i \in I \right\} = \varnothing$$

Let $S \subseteq I$ denote a subset of indices of constraints. We will say that S resolves (2) if and only if the system $\sum_{j=1}^{n} \alpha_{ij} x_j \geq \beta_i, \forall i \in I \setminus S$, is consistent. Let $|S|$ denote the cardinality of the set S. Formally, the problem we are addressing is to determine p distinct sets S_1, ..., S_p (if they exist) such that:

i. S_i resolves (2), $i \in \{1,\ldots,p\}$;
ii. $S_i \not\subset S_j$, $i,j \in \{1,\ldots,p\}$, $i \neq j$;
iii. $|S_i| \leq |S_j|$, $i,j \in \{1,\ldots,p\}$, $i<j$;
iv. If there exists a set S such that S resolves (2) and $S \not\subset S_i$ ($i \in \{1,\ldots,p\}$), then $|S| \geq |S_p|$.

Since we already know that the system (1) is consistent and the system (2) is inconsistent, we can obviously set $S_1=\{m\}$. From condition (ii), this implies that the remaining sets S_2, ..., S_p will not include $\{m\}$. In section 2.1 and 2.2, two alternative methods to solve this problem are proposed.

2.1. A method based on the {0,1} linear programming model

This first method is based on {0,1} linear programming techniques. It allows us to identify p subsets of constraints that, when removed, make the polyhedron T_k feasible.

A similar approach can be found in Kim & Ahn (1999): this is done through $p-1$ successive optimizations (PM_2, PM_3, …, PM_p).

The program PM_2 minimizes the number of constraints to be removed in order to make T_k feasible. The subset $S_1=\{m\}$ is obviously the smallest subset verifying *(i)* to *(iv)*. The first problem PM_2 to be solved has the following form:

PM_2:

$$
\begin{aligned}
&min \quad \sum_{i=1}^{m-1} y_i \\
&s.t.: \quad \sum_{j=1}^{n} a_{ij}x_j + My_i \geq \beta_i, \quad for\ i \in I \setminus \{m\} \\
&\qquad\quad \sum_{j=1}^{n} a_{ij}x_j \geq \beta_i, \quad for\ i = m \\
&\qquad\quad x_j \geq 0 \quad j=1,\dots,m, \ y_i \in \{0,1\} \quad i=1,\dots,m-1
\end{aligned}
$$

where M is a positive large number. The variables y_i, $i=1, \dots, m-1$, are binary variables assigned to each constraint. The indices of constraints for which $y_i^*=1$ (at the optimum of PM_2) constitute the subset S_2.

PM_3 is defined in order to compute S_3. This new program is derived from PM_2 by adding a single constraint of the following form:

$$\sum_{i \in S_2} y_i \leq |S_2| - 1$$

This constraint prevents PM_3 from finding an optimal solution that corresponds to (or includes) S_2. The indices of constraints for which $y_i^*=1$ (at the optimum of PM_3) constitute the subset S_3. In order to compute S_4, …, S_p, we proceed similarly: each new program is formed by adding one constraint to the previous program.

An outline of the algorithm is the following:

Let *Solutions* be a FIFO whose elements are subsets of $\{1,\dots,m-1\}$ that resolve (2).

```
Begin
   For k=2 to p do
   Solve PMk
      Sk←{i∈I\{m}:yi*=1}
   Add constraint ∑i∈Sk yi ≤ |Sk|−1 to PMk in order to define PMk+1
   End for
End
```

Let us consider the following example:

$$
\begin{array}{rcrcrl}
-0.05x_1 & - & x_2 & \geq & -40 & [1] \\
-0.5x_1 & - & x_2 & \geq & -50 & [2] \\
-1.2x_1 & - & x_2 & \geq & -70 & [3] \\
-4.5x_1 & - & x_2 & \geq & -179 & [4] \\
-x_1 & & & \geq & -35 & [5] \\
-1.3x_1 & + & x_2 & \geq & -30 & [6] \\
-0.75x_1 & + & x_2 & \geq & -14 & [7] \\
0.6x_1 & + & x_2 & \geq & 55 & [8]
\end{array}
$$

Here, system [1]-[7] is consistent, but when [8] is added the system [1]-[8] becomes inconsistent.

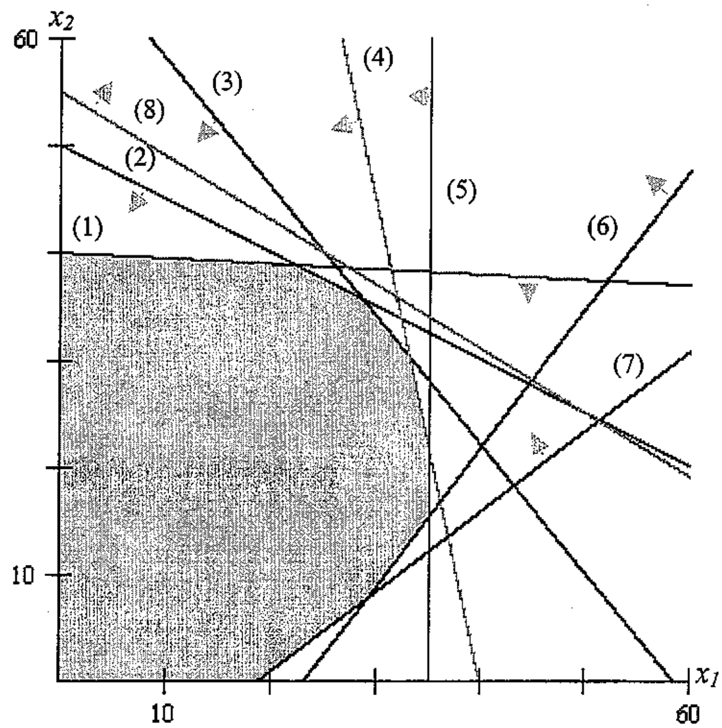

Figure 1: Feasible set.

We then build the following PM model:

$$
\begin{array}{llcccccccl}
\min & \sum_{i=1}^{7} y_i & & & & & & & & \\
s.t.: & -0.05x_1 & - & x_2 & \geq & -40 & - & My_1 & & [1] \\
 & -0.5x_1 & - & x_2 & \geq & -50 & - & My_2 & & [2] \\
 & -1.2x_1 & - & x_2 & \geq & -70 & - & My_3 & & [3] \\
 & -4.5x_1 & - & x_2 & \geq & -179 & - & My_4 & & [4] \\
 & -x_1 & & & \geq & -35 & - & My_5 & & [5] \\
 & -1.3x_1 & + & x_2 & \geq & -30 & - & My_6 & & [6] \\
 & -0.75x_1 & + & x_2 & \geq & -14 & - & My_7 & & [7] \\
 & 0.6x_1 & + & x_2 & \geq & 55 & & & & [8]
\end{array}
$$

where $y_i \in \{0,1\}$ for $i = 1,...,7$.

Let us compute, step by step, all the feasible solutions for the above example as follows,

0. $S_1=\{8\}$ (trivial solution).
1. In the first stage, we obtain $S_2=\{1,2\}$, i.e., $y_1^* = y_2^* = 1$. The constraint $y_1 + y_2 \leq 1$ is then added to the constraint set.
2. In the second stage, after optimizing PM the solution is $S_3=\{2,3\}$, i.e., $y_2^* = y_3^* = 1$. We add the constraint $y_2 + y_3 \leq 1$ to the model.
3. In the third stage, we obtain $S_4=\{3,4,5,6\}$, i.e., $y_3^* = y_4^* = {}_5^* = y_6^* = 1$. We add the constraint $y_3 + y_4 + y_5 + y_6 \leq 3$.

4. The problem becomes infeasible, meaning that there are no more alternatives to solve our problem.

2.2. An algorithm to propose solutions for inconsistency using LP

In this section we propose a second algorithm to solve the problem we are addressing, i.e., to find the sets $S_2, \ldots, S_p$ (since we are considering $S_1=\{m\}$). The algorithm is based on the results presented below:

Proposition 1. Let $S \subseteq \{1,\ldots,m-1\}$ be a set of indices of constraints. Let LP(S) denote the linear program to maximize $\sum_{j=1}^{n} \alpha_{mj} x_j$, subject to constraints of system (1), excluding the constraints in S:

$$\text{LP}(S): \qquad max\left\{\sum_{j=1}^{n} \alpha_{mj} x_j : \sum_{j=1}^{n} \alpha_{ij} x_j \geq \beta_i, \forall i \in I \setminus (S \cup \{m\})\right\}$$

Then,
a) LP(S) is always feasible, and
b) S resolves (2) $\Leftrightarrow$ LP(S) is unbounded or its optimal value is not less than β_m.

Proof:
a) Since the system (1) is consistent, it remains consistent after removing some of its constraints; hence, the linear program is feasible.
b) When LP(S) is unbounded or when the optimal value of LP(S) is greater than, or equal to, β_m, then all the constraints except the ones in S are satisfied, including the last constraint in system (2), i.e., S resolves (2). Otherwise, the last constraint in system (2) is violated and S does not resolve (2).

□

Let $x^*(S) = arg\ max\left\{\sum_{j=1}^{n} \alpha_{mj} x_j : \sum_{j=1}^{n} \alpha_{ij} x_j \geq \beta_i, \forall i \in I \setminus (S \cup \{m\})\right\}$ denote an optimal solution for the linear program above.

Let $B(S)$ be the set of indices of the constraints that are active (binding) at the solution $x^*(S)$:

$$B(S) = \left\{ i \in I \setminus (S \cup \{m\}) : \sum_{j=1}^{n} \alpha_{ij} x_j^*(S) = \beta_i \right\}.$$

Proposition 2. Let R and S be any two subsets of $I\setminus\{m\}$, such that:

- $S \subset R$;
- S does not resolve (2);
- R resolves (2).

Then, $B(S) \cap R\setminus S \neq \emptyset$.

Proof:
From Proposition 1,
R resolves (2) $\Rightarrow$ LP(R) is unbounded or its optimal value is not less than β_m, and
S does not resolve (2) $\Rightarrow$ the optimal value of LP(S) is less than β_m.
Let $x^*(S)$ denote the optimal solution of the linear program LP(S). Now, note that $x^*(S)$ is also an optimal solution of the linear program that would be formed by deleting all the constraints that are not binding at $x^*(S)$,

$max\left\{\sum_{j=1}^{n}\alpha_{mj}x_j : \sum_{j=1}^{n}\alpha_{ij}x_j \geq \beta_i, \forall i \in B(S)\right\}$, otherwise it would not be optimal to LP(*S*).

Since the polyhedron $\{x \in \mathrm{R}^n : \sum_{j=1}^{n}\alpha_{ij}x_j \geq \beta_i, \forall i \in I \setminus (S \cup \{m\})\}$ is contained in the cone $\{x \in \mathrm{R}^n : \sum_{j=1}^{n}\alpha_{ij}x_j \geq \beta_i, \forall i \in B(S))\}$, then the value of the objective function of LP(*S*) can increase only if (at least) one of the constraints in *B(S)* is removed. Thus, the fact that the objective function does increase when *S* is replaced by *R* (i.e., when $I\setminus(S\cup\{m\})$ is replaced by $I\setminus(R\cup\{m\})$ by removing constraints in $R\setminus S$), implies that at least one of the constraints in $R\setminus S$ belongs to *B(S)*.

□

Based on this result, an alternative manner to solve our problem is the following algorithm:

Let *Candidates* be a FIFO whose elements are subsets of *{1,...,m-1}*.
Let *Solutions* be a FIFO whose elements are subsets of *{1,...,m-1}* that resolve (2).

```
Begin
  Create an empty FIFO Candidates;
  Create an empty FIFO Solutions;
  SolutionIndex ← 2;
  Solve LP(∅);
  For each constraint index i∈B(∅) do
     InsertInFIFO(Candidates, {i})
  While NotEmpty(Candidates) and SolutionIndex ≤ p do
    S := RemoveFromFIFO(Candidates)
    Solve LP(S)
    If  LP(S) is unbounded or has an optimal ≥ βm then
      InsertInFIFO(Solutions, S)
      SolutionIndex ← SolutionIndex + 1
    Else
      For each constraint index i∈B(S) do
           InsertInFIFO(Candidates, S∪{i})
    End if
  End while
  Show elements in Solutions to the Decision Makers;
End
```

Justification for the algorithm:

1) *This algorithm presents the solutions* S_2 *to* S_p *by non-decreasing order of cardinality.* Before the *while* loop, the algorithm considers as candidates sets of cardinality equal to 1. In the *while* loop, when the set *S* at the head of the *Candidates* FIFO is tested (by solving LP(*S*)) and fails, the algorithm places at the tail of that FIFO other potential solutions whose cardinality equals $|S| + 1$. All candidates of cardinality $|S|$ are tested before those of higher cardinality.
2) *If a set* R *resolves (2) and no subset of* R *resolves (2), then the algorithm will find* R *for a sufficiently large value of* p.
   We will show that this is true by induction. Suppose that there exists a set $R=\{s_1, s_2, ..., s_{|R|}\}$ that resolves (2). Before the *while* loop, the elements in $B(\emptyset)$ are considered as candidates. From Proposition 2, $B(\emptyset)\cap R\neq\emptyset$, i.e., there exists an element $s_{(1)}\in R$ that enters the FIFO *Candidates*.

During the *while* loop, at a given moment, the candidates of cardinality k $(1 \le k < |R|)$ will start to appear at the head of the *Candidates* FIFO. If one of these elements S is such that $S \subset R$, then solving LP(S) yields an optimal value which is less than β_m. From Proposition 2, $B(S) \cap R \backslash S \neq \emptyset$, i.e., there exists an element $s_{(k)}$ that belongs to R and does not belong to S. This element is appended to S to constitute a set $\{s_{(1)}, ..., s_{(k)}\} \in R$ that enters the FIFO *Candidates*.

Some iterations later, the candidates of cardinality $|R|$ will start to appear at the head of the *Candidates* FIFO. One of these candidates is R, which will be declared a solution since LP(R) will either be unbounded, or have an optimal value that is not less than β_m.

<u>On the computer implementation of the algorithm</u>

One way to implement this algorithm is to solve the linear program before the *while* loop, and then another linear program for each iteration. An alternative way is to solve the initial LP(∅) and to save the simplex tableau corresponding to the removal of each constraint put in *Candidates*. In the *while* loop, solving LP(S) will amount to perform a single simplex iteration from the corresponding saved tableau. Then, for each set put in *Candidates* a new tableau must be saved. This would be faster, but would require more memory usage.

The operation InsertInFIFO() should not be performed whenever the set to insert is equal to, or includes, a set previously inserted. The elements of the set in the FIFO should be ordered by their indices to facilitate the search for this condition.

<u>Return to the example</u>

0. Initially, *Candidates* and *Solutions* are empty FIFOs.
   LP(∅) amounts to maximize $0.6x_1+x_2$, subject to constraints [1] to [7].
   The solution of LP(∅) yields an optimum value of 52.857, meaning that the constraint $0.6x_1+x_2 \geq 55$ [8] cannot be satisfied. The set of indices of the constraints that are active (binding) at the optimal solution is $B(\emptyset)=\{2,3\}$. This implies that at least one of these two constraints must be dropped to satisfy the constraint [8]. The sets {2} and {3} are added to *Candidates*.
1. The set {2} is removed from *Candidates*.
   The solution of LP({2}) yields an optimum value of 54.348 < 55.
   $B(\{2\})=\{1,3\}$. The sets {1,2} and {2,3} are added to *Candidates*.
2. The set {3} is removed from *Candidates*.
   The solution of LP({3}) yields an optimum value of 53.5 < 55.
   $B(\{3\})=\{2,5\}$. The set {3,5} is added to *Candidates* (the other set {2,3} is already in the FIFO).
3. The set {1,2} is removed from *Candidates*.
   The solution of LP({1,2}) yields an optimum value of 70 ≥ 55. Therefore, {1,2} enters the FIFO *Solutions*: if the DM removes these two constraints, the consistency is restored. No element is added to *Candidates*.
4. The set {2,3} is removed from *Candidates*.
   The solution of LP({2,3}) yields an optimum value of 59.25 ≥ 55. Therefore, {2,3} enters the FIFO *Solutions*.
5. The set {3,5} is removed from *Candidates*.
   The solution of LP({3,5}) yields an optimum value of 53.675 < 55.

The algorithms provide the information (and the corresponding constraints) to remove in order to retrieve consistency. In this case, the DM should choose to delete one of the three following constraints: [6], [7] or [10].

4. Conclusion and further research

In this paper, we have proposed two alternative algorithms to de///al with inconsistencies among constraints on the parameters of a MCDA model. The inconsistencies considered here correspond to situations in which the DM specifies a list of linear constraints on preferential parameters value that originate an empty polyhedron. More specifically, these algorithms allow us to compute subsets of constraints that, when removed, yield a non-empty polyhedron of acceptable values for preferential parameters.

The algorithms presented in section 2 are particularly useful within the context of preference elicitation through an aggregation/disaggregation process. Section 3 described and illustrated how these algorithms can be used when inferring the weights in the Electre Tri method from assignment examples.

The results presented in this paper suggest further research. First, it is obvious that the proposed algorithms can be used to solve inconsistencies on preferential parameters in various aggregation models.

Second, if some ordinal confidence index is attached to each constraint provided by the DM, it might be interesting to find the "smallest" subsets of constraints in which the DM has the least confidence. This problem leads to complex ordinal optimization programs.

Third, the proposed algorithms are specifically designed for the case in which the last constraint added causes infeasibility. Further research should be performed so as to figure out whether a variation of the algorithms could be used in the general case in which a set of constraints which is infeasible is given to begin with.

Lastly, we consider the reduction of inconsistencies through the deletion of subsets of constraints. It might be very interesting to try to relax some constraints (rather than to delete them) in order to restore consistency.

Acknowledgment: This work was supported by the Luso-French grants n° 328J4 and 500BA (ICCTI/Ambassade de France au Portugal). The authors are thankful to John W. Chinneck for comments on early versions of this paper.

References

Chinneck, J. (1994): "MINOS(IIS): Infeasibility analysis using MINOS", *Computers & Operations Research*, 21(1), 1-9.

Chinneck, J. (1996): "An effective polynomial-time heuristic for the minimum-cardinality IIS set-covering problem", *Annals of Mathematics and Artificial Intelligence*, 17, 127-144.

J.W. Chinneck, (1997), "Feasibility and Viability" *in Advances in Sensitivity Analysis and Parametric Programming*, T. Gal and H.J. Greenberg (eds.), Kluwer Academic Publishers, International Series in Operations Research and Management Science, Vol. 6.

Dias, L. & Clímaco, J. (2000): "ELECTRE TRI for groups with imprecise information on parameter values", *Group Decision and Negotiation* 9, 355-377.

Dias, L., Mousseau, V., Figueira, J. & Clímaco, J. (2000): "An aggregation/disaggregation approach to obtain robust conclusions with ELECTRE TRI", to appear in *European Journal of Operational Research.*

Dimitras, A., Zopounidis, C. & Hurson, C. (1995): "Multicriteria decision aid method for the assessment of business failure risk", *Foundations of Computing and Decision Sciences*, 20(2), 99-112.

Jacquet-Lagrèze E. & Siskos J. (1982): "Assessing a set of additive utility functions for multicriteria decision making, the UTA method", *European Journal of Operational Research*, 10(2), 151-164.

Kämpke (1996): "Sensitivity analysis for assessing preferentially independent order relations", *Computers & Operations Research,* 23, 1119-1130.

Kim, S.H., & Ahn, B.S. (1999): "Interactive group decision making procedure under incomplete information", *European Journal of Operational Research,* 116, 498-508.

Loon, J. (1981): "Irreducibly inconsistent systems of linear inequalities", *European Journal of Operational Research*, 8, 283-288.

Mousseau, V., Figueira J., & Naux, J-Ph. (2001): "Using assignment examples to infer weights for ELECTRE TRI method: Some experimental results", *European Journal of Operational Research*, 130 (2), 90-102.

Mousseau, V. & Slowinski, R. (1998): "Inferring an ELECTRE TRI Model from Assignment Examples ", *Journal of Global Optimization*, vol. 12, 157-174.

Mousseau, V., Slowinski, R. & Zielniewicz, P. (1999): "ELECTRE TRI 2.0a Methodological Guide and User's Manual", *Document N° 111*, LAMSADE, Université Paris-Dauphine.

Mousseau, V., Slowinski, R. & Zielniewicz, P. (2000): "A User-oriented Implementation of the ELECTRE-TRI Method Integrating Preference Elicitation Support", *Computers & Operations Research*, 27(7-8), 757-777.

Murty, K., Kabadi, S. & Chandrasekaran, R. (2000): "Infeasibility analysis for linear systems, a survey", *Working Paper*, Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, USA, 21 p.

Nadeau, R., Martel, J.-M. & Kiss, L. (1991): "Eleccalc: modelling the decision maker's preferences with ELECTRE II", In Glückaufova, D., Loula, D. & Cerny, M. (eds), *Proceedings of the International Workshop on Multicriteria Decision Making: Methods, Algorithms and Applicatins*, Liblice, Czechoslovakia, 109-119.

Roy, B. (1998): "A Missing Link in OR/DA: Robustness Analysis", *Foundations of Computing and Decision Sciences*, 23(3), 141-160.

Roy, B. & Bouyssou, D. (1989): "Main sources of inaccurate determination, uncertainty and imprecision in decision models", *Mathematical and Computer Modelling,* 12, 1245-1254.

Roy, B. & Bouyssou, D. (1993): *Aide multicritère à la décision : Méthodes et Cas,* Economica, Paris.

Roy, B. & Vincke, Ph. (1984): "Relational systems of preferences with one or more pseudo-criteria: some new concepts and results", *Management Science*, 30(11), 1323-1334.

Tamiz, M., Mardle, S. & Jones, D. (1996): "Detecting IIS in infeasible linear programmes using techniques from goal programming", *Computers & Operations Research*, 23(2), 113-119.

Vincke, Ph. (1999): "Robust solutions and methods in decision aid", *Journal of Multi-Criteria Decision Analysis*, 8, 181-187.

Weber, M (1987): "Decision making with incomplete information" *European Journal of Operational Research*, 28, 44-57.