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

Table of Contents..............................................................................................................2

Abstract, Résumé...............................................................................................................3

1. Introduction....................................................................................................................4

2. Two different methods to cope with inconsistent systems..........................................5
   2.1. A method based on the \{0,1\} linear programming model......................................6
   2.2. An algorithm to propose solutions for inconsistency using LP.............................9
   2.3. Comparative analysis of the algorithms.................................................................12

3. An application in the context of the inference of Electre Tri weights.........................12

4. Conclusion and further research..................................................................................14

References........................................................................................................................14
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 imprecise (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.

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 ag-
gregation of the performances. Providing precise figures for all parameter values is of-
ten difficult, to the extent that there may exist some imprecision, contradiction, arbit-
rariness, 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 &
Climaco, 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 $t_2$ is larger than parameter
$t_3$), 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 con-
straints to infer parameter values are often called aggregation/disaggregation procedures

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 se-
  lection 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 inter-
active 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 pre-
vious ones. To resolve the inconsistency that appeared in the linear system of con-
straints, 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, e.g., Chinneck, 1992) 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:
\[
\begin{align*}
\sum_{j=1}^{n} \alpha_{1j} x_j & \geq \beta_1 \\
& \vdots \\
\sum_{j=1}^{n} \alpha_{(m-1)j} x_j & \geq \beta_{(m-1)}
\end{align*}
\]  
\begin{equation}
(\alpha_{11}, \ldots, \alpha_{(m-1)n} \in \mathbb{R}, \beta_1, \ldots, \beta_{(m-1)} \in \mathbb{R}) \tag{1}
\end{equation}

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:

\[
\begin{align*}
\sum_{j=1}^{n} \alpha_{1j} x_j & \geq \beta_1 \\
& \vdots \\
\sum_{j=1}^{n} \alpha_{(m-1)j} x_j & \geq \beta_{(m-1)} \\
\sum_{j=1}^{n} \alpha_{mj} x_j & \geq \beta_m
\end{align*}
\]  
\begin{equation}
(\alpha_{11}, \ldots, \alpha_{m} \in \mathbb{R}, \beta_1, \ldots, \beta_{m} \in \mathbb{R}) \tag{2}
\end{equation}

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,

\[ T_k = \left\{ x \in \mathbb{R}^n : \sum_{j=1}^{n} \alpha_{ij} x_j \geq \beta_i, \forall i \in I \right\} = \emptyset \]

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 \( S_1, \ldots, S_p \) (if they exist) such that:

i. \( S_i \) resolves (2), \( i \in \{1, \ldots, p\} \);  

ii. \( S_i \nsubseteq S_j, i,j \in \{1, \ldots, p\}, i \neq j \);  

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 \( S \nsubseteq S_i (i \in \{1, \ldots, p\}) \), 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 \( S_2, \ldots, S_p \) 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.
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
\[
\max \left\{ \sum_{j=1}^{n} \alpha_{mj} x_j : \sum_{j=1}^{n} \alpha_{ij} x_j \geq \beta_i, \forall i \in B(S) \right\}, \text{ otherwise it would not be optimal to } \operatorname{LP}(S).
\]
Since the polyhedron \( \{ x \in \mathbb{R}^n : \sum_{j=1}^{n} \alpha_{ij} x_j \geq \beta_i, \forall i \in I \setminus (S \cup \{m\}) \} \) is contained in the cone \( \{ x \in \mathbb{R}^n : \sum_{j=1}^{n} \alpha_{ij} x_j \geq \beta_i, \forall i \in B(S) \} \), then the value of the objective function of \( \operatorname{LP}(S) \) 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 \( I(S \cap \{m\}) \) is replaced by \( I(R \cap \{m\}) \) by removing constraints in \( R \cup S \)), implies that at least one of the constraints in \( R \cup S \) belongs to \( B(S) \).

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

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

\begin{algorithm}
\textbf{Begin}
\begin{algorithmic}
\State Create an empty FIFO \textit{Candidates};
\State Create an empty FIFO \textit{Solutions};
\State SolutionIndex \leftarrow 2;
\State Solve \( \operatorname{LP}(\emptyset) \);
\For{each constraint index} \( i \in B(\emptyset) \) \do
\State \text{InsertInFIFO(\textit{Candidates}, \{i\})}
\EndFor
\While{\text{NotEmpty(\textit{Candidates}) and SolutionIndex } \leq p}
\State \( S := \text{RemoveFromFIFO(\textit{Candidates})} \)
\State Solve \( \operatorname{LP}(S) \)
\If{\( \operatorname{LP}(S) \) is unbounded or has an optimal \( \geq \beta_m \)}
\State \text{InsertInFIFO(\textit{Solutions}, S)}
\State SolutionIndex \leftarrow \text{SolutionIndex } + 1
\Else
\For{each constraint index} \( i \in B(S) \) \do
\State \text{InsertInFIFO(\textit{Candidates}, \( S \cup \{i\} \))}
\EndFor
\EndIf
\EndWhile
\State Show elements in \textit{Solutions} to the Decision Makers;
\end{algorithmic}
\textbf{End}
\end{algorithm}

\textbf{Justification for the algorithm:}
1) \textit{This algorithm presents the solutions \( S_2 \) to \( S_p \) by non-decreasing order of cardinality.}
Before the \textit{while} loop, the algorithm considers as candidates sets of cardinality equal to 1. In the \textit{while} loop, when the set \( S \) at the head of the \textit{Candidates} FIFO is tested (by solving \( \operatorname{LP}(S) \)) 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) \textit{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 \textit{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 \( s_{(l)} \in R \) that enters the FIFO \textit{Candidates}. 

10
During the while loop, at a given moment, the candidates of cardinality $k$ ($1 \leq k < |R|$) will start to appear at the head of the Candidates FIFO. If one of these elements $S$ is such that $S \subseteq R$, then solving LP$(S)$ yields an optimal value which is less than $\beta_n$. From Proposition 2, $B(S) \cap R \neq \emptyset$, i.e., there exists an element $s_{(\emptyset)}$ that belongs to $R$ and does not belong to $S$. This element is appended to $S$ to constitute a set $\{s_{(\emptyset)}, \ldots, s_{(\emptyset)}\} \in R$ 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 $\beta_n$.

On the computer implementation of the algorithm

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$(\emptyset)$ 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.

Return to the example

0. Initially, Candidates and Solutions are empty FIFOs.

   LP$(\emptyset)$ amounts to maximize $0.6x_1 + x_2$, subject to constraints [1] to [7].

   The solution of LP$(\emptyset)$ 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 deal 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


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.
