Multiple Criteria Sorting Models with Category Size Constraints

Vincent Mousseau[*], Luís C. Dias[†], José Rui Figueira[‡§*]

August 22, 2007

[*]LAMSADE, Université Paris-Dauphine, Place du Maréchal De Lattre de Tassigny, 75775 Paris Cedex 16, France, e-mail: mousseau@lamsade.dauphine.fr

[†]INESC-Coimbra, R. Antero de Quental 199, 3000-033 Coimbra, Portugal, e-mail: ldias@inescc.pt

[‡]CEG-IST, Center for Management Studies - Instituto Superior Técnico, Technical University of Lisbon, Tagus Park, Av. Cavaco Silva, 2780-990 Porto Salvo, Portugal, email: figueira@ist.utl.pt

[§]Visiting Researcher at DIMACS, Rutgers University, CoRE Building, 96 Frelinghuysen Road Piscataway NJ 08854-8018 USA

Abstract

We consider the Multiple Criteria Sorting Problem, that aims at assigning each alternative from a finite set A to one of the predefined categories. Sorting problems usually refer to absolute evaluation (the assignment of an alternative does not depend on the remaining ones), as opposed to ranking and choice problems in which the very purpose is to compare alternatives against each other. However, we may identify decision situations where the Decision Maker has concerns or constraints about the number or the proportion of alternatives assigned to each category. We therefore introduce the concept of category size and the concept of Size Constrained Sorting Problems. In such problems, the Decision Maker intends to define standards for absolute evaluation, subject to a relative evaluation of the category sizes (which indirectly involves comparison among the alternatives). We propose how to conduct a preference elicitation process to define a sorting model taking into account the conciliation between concerns relative to category size and concerns relative to sorting examples. As an illustration, we propose a procedure to infer the values for preference parameters that accounts for specifications provided by a Decision Maker about the size of categories, in the context of the UTADIS sorting model.

Keywords: *Multiple Criteria Decision Analysis, Sorting problems, Category size, UTADIS*

Introduction

Modeling a real-world decision problem using multiple criteria decision aiding involves defining a set of n_{alt} alternatives $A = \{a_1, a_2, \ldots, a_{n_{alt}}\}$ evaluated on n_{crit} criteria $g_1, g_2, ..., g_{n_{crit}}$, each criterion, g_j, being associated to a scale X_j. Several problem statements (or problem formulations) can be considered. Roy [Roy85] distinguishes three basic problem statements: choice, sorting and ranking (see also [Ban96]).

Given a set A of alternatives, choice problems consist in determining a subset $A^* \subset A$, as small as possible, composed of alternatives being judged as the most satisficing. Optimization problems are particular cases of choice problems. Ranking problems consist in establishing a preference pre-order (partial or complete) in the set of alternatives A.

Sorting problems consist in formulating the decision problem in terms of a classification, in order to assign each alternative from A to one of the n_{cat} predefined categories $C_1, C_2, \ldots, C_{n_{cat}}$. The assignment of an alternative a to the appropriate category should rely on the intrinsic value of a (and not on the comparison of a to other alternatives from A).

Among these problem statements, a major distinction concerns *relative versus absolute* judgment of alternatives. This distinction refers to the way alternatives are considered and to the type of result expected from the analysis. In the first case, alternatives are directly compared one to each other and the results are expressed using the comparative notions of "*better*" and "*worse*". Choosing and ranking are typical examples of comparative judgments. The presence (or absence) of an alternative a in the set of the best alternatives A^* results from the comparison of a to the remaining alternatives. Similarly, the position of an alternative in a ranking depends on its comparison to the remaining ones.

In the case of *absolute* judgments, each alternative is considered independently from the others in order to determine its intrinsic value by means of comparisons to norms or references. Sorting problems refer to absolute judgments and consist of assigning each alternative to one of the pre-defined categories. The assignment of an alternative $a \in A$ results from its intrinsic evaluation on all criteria (the assignment of a to a specific category does not influence the category to which another alternative should be assigned). Various methods have been developed for such assignment problem (see for instance [MR77], [DGJL80], [MO91],[Yu92], [Per98], [Bel00] and [DMFC02], see also [ZD02] for a review).

This paper deals with sorting problems where the Decision Maker (DM) has some concerns or constraints about the number or the proportion of alternatives assigned to each category. More precisely, we are concerned with what we call Size Constrained Sorting Problems (SCSP), related to a new notion regarding the size of the categories, i.e., "*the proportion of alternatives assigned to the category C_k*". SCSPs arise when the DM has an idea about how the alternatives should be distributed among the categories. For instance, imagine a sorting model to assign students to grades A,B,C, and D. The DM is asked the following question "If all of this year's students fall into the same category (it does not matter which one), would you be compelled to change the sorting model?". If the answer is yes, then we would have a SCSP.

The purpose of this paper is to present this new type of problem statement, and to illustrate how it can be dealt with in the context of inferring the parameters of a multi-criteria decision aiding model. In such model construction processes (also known as disaggregation processes, see [DGJL80, ZD00, JLS01, DZZ01, DMFC02]), the model's parameters are inferred in order to satisfy the DM's requirements, in contrast with processes that ask the DM to directly provide the parameter values. The DM's requirements are usually sorting examples that the model should reconstitute. To these requirements, we consider adding requirements relative to category sizes, noting that trade-offs may be necessary.

In the next section, we present four illustrative examples that motivate the usefulness of the notion of category size for decision aiding, aiming to show that it is very common that a Decision Maker (DM) has some concerns about the proportion of alternatives falling into each category. We analyse, in Section 2, the consequences of using category size in the sorting problem statement and define the SCSP. We contrast this new type of problem statement against choice and ranking problem statements, and make a distinction between static SCSPs (alternatives are known when the model is built and divulged) and anticipatory SCSPs (alternatives are not known at that time). Section 3 formally defines the notion of category size. Two ways to consider category size in preference elicitation processes are proposed in Section 4. The first is to proceed by trial and error; the second one is to include constraints related to category sizes into parameter inference formulations. An illustrative example is provided in Section 5, where we adapted the UTADIS method to deal with category size constraints. A final section presents conclusions and suggests further research.

1 Motivating examples

In order to motivate the introduction of the notion of category size for decision aiding, let us consider several realistic illustrative decision problems in which this notion can play an important role in the modeling process. In each of the following examples, the DM provides specifications about the respective size of categories.

Example A

Consider a corporate distribution company composed of a large number of retail stores. The head of this company wants to sort the stores according to performance-related categories. This sorting is to be grounded on several criteria (*e.g.*, profit, customer complaints, market share, ...). The CEO formulates this problem as a sorting problem in which retail stores should be assigned to one of the following five categories:

C_1 Underperforming retail store (immediate corrective action required),

C_2 Retail store having a bad performance,

C_3 Retail store having an average performance,

C_4 Retail store having good results,

C_5 Retail store that outperform (exceptional results stemming from an outstanding management).

In this situation, the CEO might want to identify and analyze only the cases corresponding to exceptions, *i.e.*, retail stores that have either very positive or very negative results. Her idea is that most of the retail stores fall into the C_3 situation and that only very few of them correspond to C_1 or C_5. She views the distribution of the retail stores among the five categories as "*bell-shaped*" (symmetric unimodal), as illustrated in Figure 1 (left).

Example B

Consider a credit manager in a financial institution who decides whether or not to grant loans to clients. His role is to accept/reject loan files or possibly refer to his superior for difficult or ambiguous cases. His decision is grounded on the various elements documented in the file. This decision problem can be formulated through a multiple criteria trichotomic segmentation (accept / refer to superior / reject).

However, the credit manager does not want to send too many files (no more than 10% in average) to his superior. In such a case, it is natural to conceive a sorting model with three categories in which *the central class is smaller in size than the remaining ones* (when considering a set of "typical" loan files), as illustrated in Figure 1 (second from left).

Example C

Every year the director of a department of a firm wants to split the grant budget among her collaborators. She considers four levels of bonuses (A: high, B: medium, C: small, D: none) according to several performance criteria discussed in the beginning of the year with her collaborators (the bonuses packages A, B and C are also stated in the beginning of the year).

The bonus policy of the director is such that very few collaborators get an A-bonus, a little more get a B-bonus, a significant proportion get a C-bonus and a large proportion of them get no bonus. In this situation, the shape of the category size distribution can be considered as "*increasing*" (or "*decreasing*" according to the way the categories are labeled), as illustrated in Figure 1 (third and fourth from left).

Example D

Each year, the responsible of a University training program faces the same problem when defining the foreign language courses. He wants to split a group of students into three groups of different levels (beginners, intermediate, advanced). The assignment of a student to a specific class is grounded on his/her skills (oral expression, listening comprehension, grammar, written expression, ...). In this situation, it might be required that the three classes should not be "too different" in size. Such a decision problem can obviously be formulated as a multiple criteria sorting problem. One of the specificities of this problem consists of the "*uniform*" size of the categories representing the three classes, as illustrated in Figure 1 (right).

These four decision problems illustrate prototypical sorting decision situations in which category size somehow intervenes in the modeling process. An analyst designing a decision aiding model for these situations should consider in the modeling process the information on the shape of the category size distribution, as suggested by the DM.

The reader will easily imagine various other problems in which this notion is an important aspect of the decision model definition. Each of these examples suggest constraints about the proportion of alternatives assigned to each category. Figure 1 depicts typical category size distributions, which we may call "bell-shaped" (Example A), "Dichotomic" (Example B), "Increasing/Decreasing" (Example C), and "Uniform" (Example D).

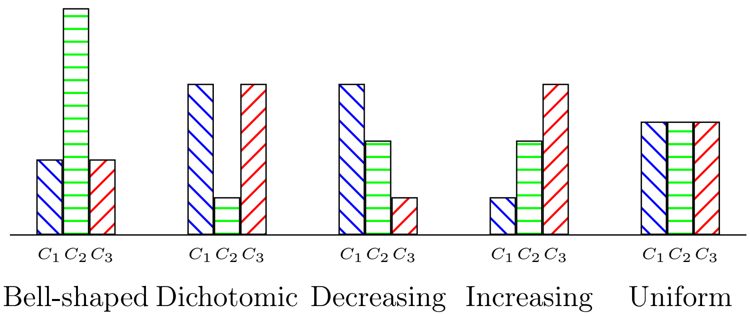

Figure 1: Typical category size distributions

2 Size Constrained Sorting Problems

We refer to Size Constrained Sorting Problem (SCSP) as decision situations formulated through a sorting model in which specifications about the size of the categories are introduced. These specifications take the form of implicit or explicit constraints on category size.

In example A for instance, these constraints specify that "the distribution of the retail stores among the five categories is *bell-shaped*". In example B, the credit manager does not want to refer to his superior more than 10% of the files. In the Example C, the director's view on the "relative proportion" of A, B, C and D-bonuses specifies constraints on the size of the corresponding categories. In example D, the constraints express the statement "the three groups should not be *too different* in size".

2.1 SCSP vs. "Standard" Sorting Problem

SCSPs differ from "standard" sorting problems, since constraints on the size of categories introduce implicit relative evaluation among alternatives. More precisely, in a "standard" sorting problem the assignment of an alternative a to a specific category only depends on the intrinsic characteristics of a (and the norms defining the categories), while in SCSPs its assignment depends not only on its intrinsic characteristics but also on the other alternatives in A. Let us note, however, that this does not mean that alternatives will be directly compared. Rather, in a given situation both the set A and the relative judgements about the category sizes contribute to shape the implicit definition of each category.

For instance, in Example D introduced above, a student that has average/low skills might be assigned to the "beginners" class when a large proportion of students have high skills, but he/she might have been assigned to the "intermediate" class if the average level of students were lower. As another example let us suppose, in Example B, that the credit manager faces an exceptional situation in which a majority of loan files are uncertain and ambiguous. In such a case, a standard sorting model would assign a large proportion of files to the "*central*" class (*i.e.*, refer to superior). The introduction of constraints about the size of this category would "force" the model to assign some alternatives previously assigned to the central class to the lower and higher classes, keeping in this *refer to superior* category only the most ambiguous cases.

Consequently, it follows that SCSPs possess some of the characteristics of the relative problem statements (choice and ranking) namely regarding the dependence of the result with respect to the set of alternatives. In SCSPs the assignment of an alternative depends on its intrinsic characteristics, but also on the assignment of the other alternatives. Such a problem statement is however distinct from relative problem statements and can be considered as an intermediate situation, since it deals both with relative and absolute issues. Absolute evaluation is present in the sorting model, and the very idea of sorting, while comparative evaluation stems from the inclusion of constraints concerning category size.

2.2 SCSP vs. choice and ranking problems

It is interesting to note that choice problems can be expressed through a SCSP, if the categories are ordered. Consider a sorting problem with two categories (C_1: select, C_2: reject). Let us impose that the size of C_1 is equal to n. If we pose $n = 1$, then this problem corresponds to a choice. Moreover, relaxing the constraint concerning the size of C_1 (*i.e.*, increasing the value for n) leads to interesting formulations with respect to the choice problem.

Conversely, some SCSPs can be formulated as choice or ranking problem statements, but only if the categories are ordered. If there exists a size constraint stating that the first category should contain k alternatives, this corresponds to the choice of the k best alternatives. If there exist size constraints stating that k_1 alternatives belong to C_1, k_2 alternatives belong to C_2, etc., this may be accomplished by ranking the alternatives from the best to the worst and breaking the ranking into n_{cat} segments.

However, there exist SCSPs that cannot be formulated using a relative formulation (choice or ranking). For instance, a situation like Example B cannot be solved by ranking the alternatives because we would not know which segment with 10% of the alternatives (in the middle of the ranking) should be selected. Indeed, to solve a SCSPs by ranking the alternatives is possible only if the DM states the required sizes for all the categories, and even then the size constraints might not be respected if the ranking was not a linear order (i.e. if the ranking has ex-aequo alternatives).

2.3 Anticipatory SCSPs

We can distinguish among situations in which the set of alternatives A is completely known or not before building the model. Suppose that we have a complete description of the set of alternatives A. In this case, constraints on category size may be specified during the model definition. The resulting model explicitly integrates these constraints, and it may be possible to guarantee that the resulting sorting respects these constraints. For instance, in Example D, the three groups of students are to be defined considering the actual students of the current year. This is what we could call a *"Static" SCSP*.

In other situations, a sorting model is being built to evaluate alternatives that will appear in the future or alternatives with performances that vary over time. In these situations, A is not known beforehand and we hence face uncertainty concerning the alternatives' performances. A sorting model is to be built taking into account constraints about the size of the categories, *given the alternatives that are realistically likely to appear*. We will refer to these problems as *"Anticipatory" SCSPs*.

In static SCSPs the sorting model is only a means to an end (the partitioning of set A), which as we have described might in some occasions be accomplished even without building a sorting model (namely, resorting to a ranking). In contrast, in anticipatory SCSPs, the sorting model is the objective of the decision aiding process.

In anticipatory SCSPs, the model is built (and may be divulged) before the actual alternatives are known, and constraints about category size may possibly be violated when the actual set of alternatives is considered. For instance, in example C, the director of the department would want to announce the criteria for granting bonuses, and if the employees work exceedingly well she may have to grant more A bonuses than she was expecting. anticipatory SCSPs may also correspond to situations where the alternatives arrive separately and have to be evaluated as soon as they arrive (rather than all at the same time), as for instance loan applications.

3 Definition of category size

We have referred to the size of category C_k resulting from a sorting model as "*the proportion of alternatives assigned to the category C_k*". Let us consider a specific sorting model that uses a set of preference parameters Ω (such as criteria weights, limits of categories, ...). Let $\mathcal{P}$ denote the domain of possible values for the parameters in Ω. Let $C(a_i, p)$ denote the index of the category to which $a_i \in A$ is assigned when the model parameters take values $p \in \mathcal{P}$. Let each alternative be defined by its evaluations on n_{crit} criteria. For the j^{th} criterion ($j = 1, \ldots, n_{crit}$), the evaluations may take values on a scale X_j. Let $K = \{1, \ldots, n_{cat}\}$ be the set of category indices.

Given a set of alternatives A and a vector of parameter values $p \in \mathcal{P}$, we define the size of each category C_k as the proportion of alternatives from A that are assigned to C_k, *i.e.*,

$$\mu_p(C_k) = \frac{|\{a_i \in A : C(a_i, p) = k\}|}{|A|} \tag{1}$$

The above definition satisfies the following desirable properties:

$$\begin{cases} \mu_p(C_k) \geq 0, \ \forall k \in K \\ \sum_{i=1}^{n_{cat}} \mu_p(C_k) = 1 \\ \mu_p(\cup_{k \in K'} C_k) = \sum_{k \in K'} \mu_p(C_k), \ \ \forall K' \subset K \end{cases} \tag{2}$$

In such situations, there is a complete knowledge about the alternatives, which excludes anticipatory SCSPs. In the case of anticipatory SCSPs, the set A would be a set representative of alternatives likely to appear in the future, based on a record of past alternatives and/or forecasts from experts. Hence, we should refer to the size of category C_k resulting from a sorting model as "*the proportion by which an evaluation vector corresponding to a typical distribution of alternatives (as provided by a representative set) is assigned to the category*".

An additional analysis is to find the maximum and minimum size of each category, given a domain for the vector of parameters p. Assuming that the information provided by the DMs allows to define a domain $P \subset \mathcal{P}$, we may compute:

$$\begin{array}{l} \mu_p^{min}(C_k) = \min_{p \in P}(\mu_p(C_k)) \\ \mu_p^{max}(C_k) = \max_{p \in P}(\mu_p(C_k)) \end{array} \tag{3}$$

This provides the DMs an idea of the interval for the size of each category given the lack of precise knowledge about the parameter values. These intervals become narrower as more information about the parameter values is added (*i.e.*, as P becomes smaller).

4 Considering category size in preference elicitation processes

In some situations the parameter values are not precisely known. The lack of a precise vector of preference-related parameter values may stem from various sources:

- the DMs find it hard to precisely answer some questions regarding their preferences,
- the DMs may not fully understand the role of all the parameters,
- the model may be used in the future and the DMs may not know how their preferences will evolve,
- etc.

The elicitation of a multiple criteria sorting model amounts at assigning precise values to the preference parameters used by the aggregation model, *i.e.*, to select an appropriate $p^* \in \mathcal{P}$. This task can be accomplished either,

- by a direct questioning procedure with the DM,
- or indirectly through the use of an inference program that induces parameter values that restore holistic judgments (*e.g.*, assignment examples) provided by the DM (see for instance [JS82] and [JLS01] for such a disaggregation approach).

In this section we discuss how the concept of category size may be exploited in such preference elicitation processes. As input, we consider a domain $P_0 \subset \mathcal{P}$ for possible parameter values and a set A of alternatives to assign to categories.

The concept of category size may be used to support an elicitation process by trial and error, where the DM chooses a combination of values for the parameters $p \in \mathcal{P}$ and observes the computed category sizes corresponding to it through pictures similar to those in Figure 1. If what the DM sees does not correspond to his/her intuition of what the distribution of category sizes should look like, then he/she may change the parameter values, by trial and error, until a satisfactory distribution is found. However, unless the DM is using a very simple assignment model (e.g. one that depends on a small number of parameters and such that the effects of changing each parameter are easy to predict), then a trial and error process may become cumbersome.

Therefore we consider the case where the sorting model is inferred. In such cases, the DM does not have to be an expert in the sorting method, and may simply provide assignment examples for a subset of alternatives $A^* \subset A$, *i.e.*, alternatives for which the DM defines a specific assignment. In addition, the DM can provide some constraints related to his/her intuitive view on the "*size of each category*", and may provide additional constraints on parameter values.

Constraints on category sizes can be expressed in various manners:

- exact values, e.g., "there should be 5% alternatives in C_1";
- bounds, e.g., "there should be at most 5% alternatives in C_1";
- intervals, e.g., "there should be between 5% and 10% alternatives in C_1";
- comparisons, e.g., "there should be more alternatives in category C_1 than in C_2".

This information can be translated into constraints on p (defining $P \subset \mathcal{P}$) in a mathematical program, because a set of alternatives A has already been fixed. The details of such a mathematical program will vary from a sorting method to another (the next section provides an example for a UTADIS-like method).

Since assignment examples and constraints on category size are expressed through constraints in the inference mathematical program, these two types of constraints might be conflicting. This implies that the inference program should specify how such potential conflicts among these constraints are to be solved (see Section 5, for further discussion in the UTADIS framework).

The set A used to build the model (set of representative actions) should be representative of the actual set of alternatives that will be evaluated, avoiding any kind of biases. For instance, if a model is being built to evaluate mortgage loan requests and typically 20% of the requests come from single-income families, then approximately 20% of the alternatives in A should reflect this characteristic. The set A will also need to contain enough elements for the precision envisaged. For the subset A^* of examples, the only requirement is that the DM feels confident in assigning those examples to categories.

If the DM is capable of holistically evaluating all the elements of the representative set A (i.e., $A^* = A$), for instance using a database of past decisions, then the size constraints might not be needed: the resulting category sizes might already reflect the requirements of the DM. However, the size constraints might be needed if the DM found the result did not fulfill those requirements, meaning that some examples might have to be assigned to different categories.

Using size constraints allows to work with a set of examples A^* much smaller than the representative set A, containing only the judgements in which the DM is more trustful. An important aspect to note in this approach is that the example set is not necessarily a representative sample in what concerns category size. For instance, referring to motivating example A (retail stores), the CEO could provide two examples for each category and still state that most alternatives should fall into category C3. This allows eliciting examples (e.g., "Please provide one example for each category") without linking the number of examples in each category to the size of the category (the example in Section 5.2 also illustrates this point). This is possible because category sizes will be measured considering the representative set A and not just the set of examples A^*.

Finally, let us note that in anticipatory SCSPs the actual set of alternatives will seldom be equal to the set A used to build the model. Therefore, the inferred model that satisfies all category size constraints when A is considered, may no longer satisfy some of them when the unknown future alternatives are considered instead. For this reason, when a DM places a constraint like "there should be 5% alternatives in the top category", he/she should expect that *around* 5% alternatives, and *not exactly* 5%, will appear in the top category when using the model in the future. The DM may even determine beforehand (e.g., using a Monte-Carlo simulation) the probability of violating each constraint.

This characteristic of anticipatory SCSPs should not be seen as a drawback. Indeed, it may be essential. For instance, in Example A (Section 1) the CEO may provide size constraints stating that less than 10% of the retail stores should fall into categories C_1 or C_5, whereas more than 50% should be sorted into C_3. This reflects the CEO's view of normality, and a model might be built using past data as a representative set. However, if for some reasons the performances of the actual retail stores become atypical (e.g. 30% of them fall into C_1), then the CEO would want to be alerted for that. Surely, she would not want a model that would automatically adjust its parameters to keep the underperformers under a 10% size barrier.

5 Illustration: a UTADIS-like method dealing with SCSPs

In order to illustrate the use of the category size concept, we propose a procedure to infer the preference parameters values that account both for assignment examples and specifications about the size of categories, in the context of the UTADIS sorting model ([DGJL80], [ZD00]).

5.1 A brief reminder of the UTADIS method

UTADIS is a multiple criteria sorting method that assigns alternatives $a_i \in A = \{a_1, a_2, \ldots, a_i, \ldots, a_{n_{alt}}\}$ to one of the predefined ordered categories $C_1, C_2, \ldots, C_k, \ldots, C_{n_{cat}}$ (C_1 being the worst category) on the basis of a set of criteria $\{g_1, g_2, \ldots, g_j, \ldots, g_{n_{crit}}\}$. In what follows, we will assume, without any loss of generality, that preferences increase with the value on each criterion. Let $F = \{1, \ldots, j, \ldots, n_{crit}\}$ denote the set of criteria indices, $I = \{1, \ldots, i, \ldots, n_{alt}\}$ denote the set of alternatives indices, and by $K = \{1, \ldots, k, \ldots, n_{cat}\}$ the set of categories indices. The UTADIS sorting method is an additive utility model of the form:

$$u(a_i) = \sum_{j \in F} u_j(g_j(a_i)) \;\in\; [0,1], \quad \forall\, i \in I, \tag{4}$$

where,

- $g_j(a_i)$ is the evaluation of alternative a_i on criterion g_j, $\forall i \in I$, $\forall j \in F$,
- u_j is a piecewise linear utility function for criterion g_j, $\forall j \in F$, each u_j ranges on the set $[0, w_j]$ (we pose without loss of generality $\sum_{j \in F} w_j = 1$).

UTADIS assigns each alternative $a_i \in A$ by comparing $u(a_i)$ to a set of category limits, $b_1, \ldots, b_k, \ldots, b_{n_{cat}-1}$ such that $b_k > b_{k-1},\; k = 2, ..., n_{cat} - 1$. These limits are used to assign alternatives in the following way (Figure 2 illustrates the assignment process described above):

$$\begin{cases} a_i \in C_1 & \Leftrightarrow \quad 0 \le u(a_i) < b_1 \\ & \vdots \\ a_i \in C_k & \Leftrightarrow \quad b_{k-1} \le u(a_i) < b_k \\ & \vdots \\ a_i \in C_{n_{cat}} & \Leftrightarrow \quad b_{n_{cat}-1} \le u(a_i) \le 1 \end{cases} \tag{5}$$

Let g_j^m (g_j^M, respectively) be the minimum (maximum, respectively) evaluation on criterion g_j, $\forall j \in F$. The interval $[g_j^m, g_j^M]$ is divided into L_j equal subintervals: $[g_j^0, g_j^1[, \ldots, [g_j^l, g_j^{l+1}[, \ldots, [g_j^{L_j-1}, g_j^{L_j}]$ ($g_j^0 = g_j^m$ and $g_j^{L_j} = g_j^M$), where g_j^l is computed as follows:

$$g_j^l = g_j^m + \frac{l}{L_j}(g_j^M - g_j^m), \quad l = 0, \ldots, L_j \;\text{ and }\; j \in F \tag{6}$$

Each piecewise linear function u_j is defined by the utilities of breakpoints $u_j(g_j^0) \le u_j(g_j^1) \le \ldots \le u_j(g_j^{L_j})$ (we recall that $u_j(g_j^0) = 0$ and $u_j(g_j^{L_j}) = w_j$). If $g_j(a_i) \in [g_j^l, g_j^{l+1}[$, then the partial utility is obtained by linear interpolation: $u_j(g_j(a_i)) = u_j(g_j^l) + \frac{g_j(a_i) - g_j^l}{g_j^{l+1} - g_j^l}(u_j(g_j^{l+1}) - u_j(g_j^l))$. Hence, the parameters of the UTADIS sorting model are the following:

- The utility of each breakpoint g_j^l, i.e., $u_j(g_j^l)$, for $j \in F$ and $l = 1, \ldots, L_j$.
- The category limits, b_k, for $k = 1, \ldots, n_{cat} - 1$.

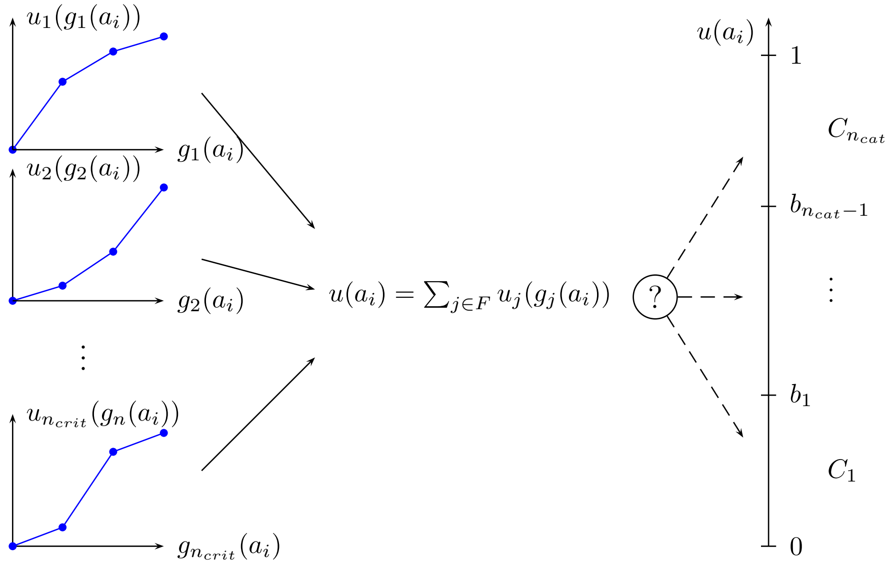

Figure 2: UTADIS sorting scheme

Let $A^* \subset A$ denote a subset of alternatives that the DM intuitively assigns to a specific category (A^* contains the assignment examples). UTADIS aims at inferring the parameters values that best match the assignment examples. Suppose the DM stated that alternative $a_i \in A^\star$ should be assigned to the category C_k ($a_i \to C_k$). This statement generates constraints on the parameters values: $b_{k-1} \leq u(a_i) < b_k$. In order to integrate these constraints in a mathematical program, two slack variables $\delta^-(a_i)$ and $\delta^+(a_i)$ are introduced as follows (ϵ is an arbitrarily small positive constant):

$$\begin{cases} u(a_i) - b_k - \delta^-(a_i) \leq \epsilon \\ u(a_i) - b_{k-1} + \delta^+(a_i) \geq 0 \end{cases} \tag{7}$$

The linear program (8)-(17) infers the parameter values that best restore a set of assignment examples:

$$\begin{aligned}
\min\ z \quad = \quad & \textstyle\sum_{a_i \in C_1} \delta^-(a_i) + \ldots + \sum_{a_i \in C_k} (\delta^-(a_i) + \delta^+(a_i)) + \ldots + \sum_{a_i \in C_{n_{cat}}} \delta^+(a_i) && (8)\\
s.t: \quad & \textstyle\sum_{j \in F} u_j(g_j(a_i)) - b_1 - \delta^-(a_i) \leq \epsilon, \quad \forall a_i \in C_1 && (9)\\
& \textstyle\sum_{j \in F} u_j(g_j(a_i)) - b_k - \delta^-(a_i) \leq \epsilon, \quad \forall a_i \in C_k, \quad k = 2, \ldots, n_{cat} - 1 && (10)\\
& \textstyle\sum_{j \in F} u_j(g_j(a_i)) - b_{k-1} + \delta^+(a_i) \geq 0, \quad \forall a_i \in C_k, \quad k = 2, \ldots, n_{cat} - 1 && (11)\\
& \textstyle\sum_{j \in F} u_j(g_j(a_i)) - b_{n_{cat}-1} + \delta^+(a_i) \geq 0, \quad \forall a_i \in C_{n_{cat}} && (12)\\
& u_j(g_j^{l+1}) - u_j(g_j^l) \geq 0 \ \ \forall j \in F, \ l = 1, \ldots, L_j - 1 && (13)\\
& u_j(g_j^0) = 0, \ \ \forall \ j \in F && (14)\\
& \textstyle\sum_{j \in F} u_j(g_j^{L_j}) = 1 && (15)\\
& b_k - b_{k-1} \geq \epsilon, \quad k = 2, \ldots, n_{cat} - 1 && (16)\\
& \delta^-(a_i), \delta^+(a_i) \geq 0, \quad \forall a_i \in A^\star && (17)
\end{aligned}$$

5.2 Considering category size constraints in UTADIS

Let us consider the data of a real-world application concerning a credit granting application [Yu92]. This application deals with 40 alternatives evaluated on 7 criteria to be assigned to 3 ordered categories: C_1: to reject, C_2: to analyze, C_3: to accept.

The knowledge of the credit manager leads to define intervals of variation for the parameters values as follows:

- the shape of the functions u_j is imprecisely known: partial utility functions u_j are defined as follows: $u_j(g_j^h) = w_j.u_j^h$ where $u_j^h \in [\underline{u}_j^h, \overline{u}_j^h]$, h=0..5 (intervals $[\underline{u}_j^h, \overline{u}_j^h]$ are defined in table 1 hereafter).
- the criteria weights are such that $w_j \in [0.1, 0.2]$, $\forall j \in F$, (note: $w_j = u_j(g_j^{L_j})$) and
- the categories profiles or limits are such that $b_1 \in [0.5, 0.6]$ and $b_2 \in [0.65, 0.7]$.

	g_1	g_2	g_3	g_4	g_5	g_6	g_7
$[\underline{u}_j^0, \overline{u}_j^0]$	0	0	0	0	0	0	0
$[\underline{u}_j^1, \overline{u}_j^1]$	$[0.25, 0.6]$	$[0, 0.18]$	$[0, 0.16]$	$[0, 0.2]$	$[0, 0.125]$	$[0, 0.175]$	$[0, 0.3]$
$[\underline{u}_j^2, \overline{u}_j^2]$	$[0.5, 0.65]$	$[0.22, 0.42]$	$[0.14, 0.34]$	$[0.2, 0.4]$	$[0.1, 0.3]$	$[0.18, 0.45]$	$[0.1, 0.5]$
$[\underline{u}_j^3, \overline{u}_j^3]$	$[0.65, 0.8]$	$[0.62, 0.82]$	$[0.44, 0.64]$	$[0.6, 0.8]$	$[0.7, 0.9]$	$[0.425, 0.7]$	$[0.45, 0.8]$
$[\underline{u}_j^4, \overline{u}_j^4]$	$[0.75, 0.99]$	$[0.8, 1]$	$[0.86, 0.99]$	$[0.8, 1]$	$[0.875, 1]$	$[0.7, 0.85]$	$[0.75, 0.9]$
$[\underline{u}_j^5, \overline{u}_j^5]$	1	1	1	1	1	1	1

Table 1: Shape of the partial utility functions

Moreover, the credit manager usually sends about 10% of the files to his/her superior for further analysis, which means that, C_2 should contain "approximately" 4 files out of 40. Hence, our purpose is to build a model taking into account the constraint about the size of category C_2.

As a first step, it may be helpful to compute the minimum and maximum size for each category (see (3)). This will give an idea of the possible category sizes before considering any size constraints or sorting examples. Let us define the decision variables y_{ik} such that:

$$y_{ik} = \begin{cases} 1, & \text{if the alternative } a_i \rightarrow C_k \\ 0, & \text{otherwise.} \end{cases} \quad \forall i : a_i \in A,\ k = 1, 2, 3 \tag{18}$$

These y_{ik} variables can be defined in a mathematical program by the following constraints where M is a large positive constant and ε a small positive constant.

$$\sum_{j=1}^{n_{crit}} u_j(g_j(a_i)) - b_k + My_{ik} \leq M - \varepsilon, \tag{19}$$

$$-\sum_{j=1}^{n_{crit}} u_j(g_j(a_i)) + b_{k-1} + My_{ik} \leq M, \tag{20}$$

Hence the expression $\sum_{i=1}^{n_{alt}} y_{ik}$ denotes the number of alternatives assigned to category C_k. The following program computes the maximum value for the size of C_k (the computation of the minimum is analogous). These computations lead to the results given in Table 2. It appears that, considering the imprecision of the data, the size of C_2 is in the interval $[0, 21]$ (see Table 2). However, the credit manager wants to send approximately 10% of the files (i.e,

±4 files) to his/her superior.

$$
\begin{aligned}
&\text{Max } \sum_{i=1}^{40} y_{ik} \\
&\quad \text{s.t.} \quad \sum_{j=1}^{7} u_j(g_j(a_i)) - b_k + My_{ik} \leq M - \varepsilon, \quad i = 1, \ldots, 40, \ k = 1, 2 && (21)\\
&\qquad\qquad - \sum_{j=1}^{7} u_j(g_j(a_i)) + b_{k-1} + My_{ik} \leq M, \quad i = 1, \ldots, 40, \ k = 1, 2 && (22)\\
&\qquad\qquad u_j(g_j^{l+1}) - u_j(g_j^l) \geq 0 \ \ \forall j \in F, \ l = 1, \ldots, L_j - 1 && (23)\\
&\qquad\qquad u_j(g_j^0) = 0, \ \ \forall \ j \in F && (24)\\
&\qquad\qquad u_j(g_j^h) \in [w_j \underline{u}_j^h, w_j \overline{u}_j^h], \ \ h = 1, ..., 4, \ \forall j \in F && (25)\\
&\qquad\qquad u_j(g_j^{L_j}) = w_j \in [w_j^m, w_j^M], \ j = 1, \ldots, 7, && (26)\\
&\qquad\qquad b_2 \geq b_1 + \varepsilon, && (27)\\
&\qquad\qquad b_k \in [b_k^m, b_k^M], \ k = 1, 2 && (28)\\
&\qquad\qquad y_{i1} + y_{i2} + y_{i3} = 1, \ i = 1, \ldots, 40 && (29)\\
&\qquad\qquad y_{ik} \in \{0, 1\}, \ i = 1, ..., 40, \ k = 1, 3 && (30)
\end{aligned}
$$

	$\mu_p^{min}(C_k)$	$\mu_p^{max}(C_k)$
C_1	3	22
C_2	0	21
C_3	12	30

Table 2: Intervals for the size of categories $\mu_p(C_k)$

Furthermore, as assignment examples, the credit manager has identified some files (a_{42}, a_{53}, a_{61}) for which he/she should refer to his/her superior, as well as some files to be rejected (a_{22}, a_{27}, a_{41}) and some to be accepted (a_{80}, a_{90}). Note that this set of examples does not have to reflect the relative proportion of alternatives that is envisaged for the second category (i.e, ±10% files).

A possible approach to take into account both the assignment examples and the category size constrains is a lexicographical one: first, one checks whether it is possible to satisfy the assignment examples; then, among the multiple models that reassign correctly the assignment examples, one tries to satisfy the category size constraints as much as possible. The linear program (8)-(17) allows to verify that the provided assignment examples fit the UTADIS additive model ($z^* = 0$). If it did not fit ($z^* > 0$) then, either z^* would be considered as an acceptable error, or the examples should be revised, or a suitable compromise between the different types of constraints ought to be found (more details on the latter option will be provided below).

In order to cope with the manager's requirements concerning the category size, we define a mathematical program to infer (in the domain of acceptable values of the parameters) a model as compatible as possible with this constraint on the size of C_2, while conforming to the DM's assignment examples.

As the expression $\sum_{i=1}^{n_{alt}} y_{ik}$ denotes the number of alternatives assigned to category C_k and can be used to define constraints on the size of C_k. In our case, the statement 'C_2 *should* *contain approximately 10 % of the files*" (i.e., 4 alternatives) can be formulated by using the two following constraints:

$$\begin{cases} \sum_{i=1}^{n_{alt}} y_{i2} \geq 4 - \sigma \\ \sum_{i=1}^{n_{alt}} y_{i2} \leq 4 + \sigma \end{cases} \tag{31}$$

where σ is a variable to be minimized. Moreover, the assignment examples can be easily integrated as constraints by setting the values of the corresponding y_{ik} variables ($a_i \rightarrow C_k \Leftrightarrow y_{ik} = 1$). In the following mathematical program, the eight assignment examples that were provided by the DM are enforced by the constraint (34).

$$\begin{aligned}
&\text{Min } \sigma \\
&\text{s.t.} \quad \sum_{j=1}^{7} u_j(g_j(a_i)) - b_k + My_{ik} \ \leq \ M - \varepsilon, \ \ i = 1, \ldots, 40, \ k = 1, 2 && (32)\\
&\quad -\sum_{j=1}^{7} u_j(g_j(a_i)) + b_{k-1} + My_{ik} \ \leq \ M, \ \ i = 1, \ldots, 40, \ k = 1, 2 && (33)\\
&\quad y_{42,2} = y_{53,2} = y_{61,2} = y_{22,1} = y_{27,1} = y_{41,1} = y_{80,3} = y_{90,3} = 1 && (34)\\
&\quad \sum_{i=1}^{40} y_{i2} \geq 4 - \sigma && (35)\\
&\quad \sum_{i=1}^{40} y_{i2} \leq 4 + \sigma && (36)\\
&\quad u_j(g_j^{l+1}) - u_j(g_j^{l}) \geq 0 \ \ \forall j \in F, \ l = 1, \ldots, L_j - 1 && (37)\\
&\quad u_j(g_j^{0}) = 0, \ \ \forall \ j \in F && (38)\\
&\quad u_j(g_j^{h}) \in [w_j \underline{u}_j^{h}, w_j \overline{u}_j^{h}], \ \ h = 1, ..., 4, \ \forall j \in F && (39)\\
&\quad u_j(g_j^{L_j}) = w_j \in [w_j^{m}, w_j^{M}], \ j = 1, \ldots, 7, && (40)\\
&\quad b_2 \geq b_1 + \varepsilon, && (41)\\
&\quad b_k \in [b_k^{m}, b_k^{M}], \ k = 1, 2 && (42)\\
&\quad y_{i1} + y_{i2} + y_{i3} = 1, \ i = 1, \ldots, 40 && (43)\\
&\quad y_{ik} \in \{0, 1\}, \ i = 1, ..., 40, \ k = 1, 2, 3, \ \sigma \geq 0 && (44)
\end{aligned}$$

After solving this mathematical programming model we have the following results:

- the optimal value of program (32)-(44) is $\sigma^* = 6$: there exists no combination of parameter values that satisfies the credit manager's request both in terms of assignment examples and requirement about the size of C_2. In other words, when the credit manager wants to restore his/her assignment examples, the minimal size for C_2 is 10 (6 alternatives more than desired size for C_2).
- The 10 files to analyze are: $\{a_{42}, a_{43}, a_{44}, a_{51}, a_{52}, a_{53}, a_{54}, a_{57}, a_{59}, a_{61}\}$,

On the basis of this first result, the credit manager may want to refine the analysis and get a deeper understanding on how the size constraint is conflicting with the assignment examples. To do so, one may use a bi-criteria formulation to find a compromise among the violation of size constraints and the violation of sorting examples. This might be needed if (32)-(44) did not manage to satisfactorily address the DM's concerns about the category size, which would be revealed by an optimal value for σ too high (according to the DM's perspective). To obtain the efficient frontier of this bi-criteria problem, we replace constraint (34) by a relaxed version of this constraint:

$$y_{42,2} + y_{53,2} + y_{61,2} + y_{22,1} + y_{27,1} + y_{41,1} + y_{80,3} + y_{90,3} \geq T \tag{45}$$

where T would be progressively decreased (T=8, 7, 6, and so on) until reaching an acceptable compromise between the number of assignment examples satisfied and the category sizes. This highlights which assignment examples were responsible for the high values of σ in the optimal solution of (32)-(44). Doing so, we obtain three efficient solutions to this bi-criteria problem (these solutions are represented in Figure 3):

A All assignment examples are correctly assigned but the size of C_2 is 10 (optimal solution of program (21)-(30)),

B One assignment example is incorrectly assigned (a_{42} is assigned to C_1, instead of C_2), and the size of C_2 is 5: $\{a_{53}, a_{56}, a_{61}, a_{62}, a_{63}\}$.

C Two assignment examples are incorrectly assigned (a_{53} and a_{61} are assigned to C_3, instead of C_2), and the size of C_2 is 4: $\{a_{28}, a_{42}, a_{51}, a_{58}\}$.

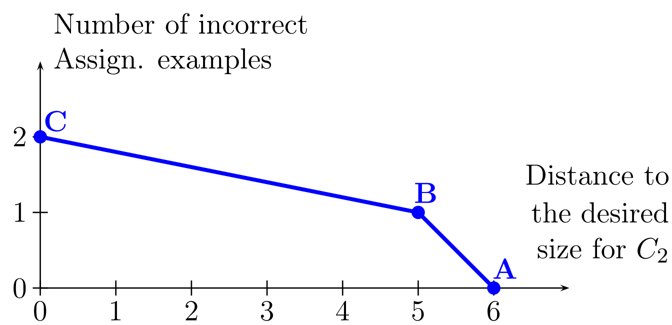

Figure 3: Bi-criteria representation of solutions

Conclusion

Sorting problems consist of formulating the decision problem in terms of a classification that assigns each alternative from A to one of the predefined categories $C_1, C_2, \ldots, C_{n_{cat}}$. The assignment of an alternative a to the appropriate category should rely on the intrinsic value of a (and not on the comparison of a to other alternatives from A). On the contrary, the very nature of ranking and choice problems is to compare alternatives one to another to determine a preference order or the subset of the best one(s). Hence ranking and choice refer to relative evaluation while sorting refers to absolute evaluation.

This paper introduces the notion of SCSP and its interest in decision aiding. We have motivated the use of the notion of category size in sorting problems and given a formal definition to this notion. We have shown that considering constraints on category size leads to define a new type of problem, the SCSP, that has both an absolute and relative evaluation aspects. It has aspects of an absolute evaluation, because alternatives are compared with the implicit standards of the DM and the alternatives are not directly compared. It has also aspects of a relative evaluation: the implicit standards defining the categories will depend on the set A of alternatives being sorted and the judgements about category sizes, therefore the category of an alternative may change if A changes.

We have proposed how to make the category size concept operational even in decision situations where the DM's preferences are imprecise. The UTADIS illustration shows an operational process to take into account category size constraints as well as constraints derived from sorting examples. The lexicographic sequence will be particularly useful when the set of examples is easily reproduced by a UTADIS model. Otherwise, the DM may withdraw some of the examples or use a more complex bi-criteria model to find a suitable compromise among his/her two objectives.

We deem the notions of category size and SCSP open a new research avenue that ought to be pursued. On the one hand, future research may work on the design of elicitation procedures that allow DMs to specify contraints on category size, thus integrating the notion of category size in the various existing sorting methods. On the other hand, new multicriteria sorting methods might be devised to deal specifically with SCSPs.

Acknowledgements:
This work has benefited from the luso-french grant n^o: 07863YE (GRICES/EGIDE) and FCT/FEDER grant POCI/EGE/58371/2004. The third author was also supported by the grant SFRH/BDP/6800/2001.

References

[Ban96] C.A. Bana e Costa. Les problematiques de l'aide a la decision: vers l'enrichissement de la trilogie choix-tri-rangement. *RAIRO/ Recherche Operationnelle*, 30(2):191–216, 1996.

[Bel00] N. Belacel. Multicriteria assignment method PROAFTN: methodology and medical application. *European Journal of Operational Research*, 125(1):175–183, August 2000.

[DGJL80] J.M. Devaud, G. Groussaud, and E. Jacquet-Lagreze. UTADIS: Une methode de construction de fonctions d'utilite additives rendant compte de jugements globaux. In *European working group on MCDA, Bochum , Germany*, 1980.

[DMFC02] L.C. Dias, V. Mousseau, J. Figueira, and J.N. Clímaco. An aggregation/disaggregation approach to obtain robust conclusions with ELECTRE TRI. *European Journal of Operational Research*, 138(2):332–348, April 2002.

[DZZ01] M. Doumpos, S.H. Zanakis, and C. Zopounidis. Multicriteria preference disaggregation for classification problems with an application to global investing risk. *Decision Sciences*, 32(2):333–385, 2001.

[JLS01] E. Jacquet-Lagrèze and Y. Siskos. Preference disaggregation: 20 years of MCDA experience. *European Journal of Operational Research*, 130(2):233–245, April 2001.

[JS82] E. Jacquet-Lagrèze and Y. Siskos. Assessing a set of additive utility functions for multicriteria decision making: the UTA method. *European Journal of Operational Research*, 10:151–164, 1982.

[MO91] R. Massaglia and A. Ostanello. N-tomic: a support system for multicriteria segmentation problems. In P. Korhonen, A. Lewandowski, and J. Wallenius, editors, *Multiple Criteria Decision Support*, pages 167–174. Springer Verlag, LNEMS 356, Berlin, 1991.

[MR77] J. Moscarola and B. Roy. Procédure automatique d'examen de dossiers fondée sur une segmentation trichotomique en présence de critères multiples. *RAIRO Recherche Opérationnelle*, 11(2):145–173, 1977.

[Per98] P. Perny. Multicriteria filtering methods based on concordance/non-discordance principles. *Annals of Operations Research*, 80:137–167, 1998.

[Roy85] B. Roy. *Méthodologie multicritère d'aide à la décision.* Economica, Paris, 1985.

[Yu92] W. Yu. *Aide multicritère à la décision dans le cadre de la problématique du tri: méthodes et applications.* PhD thesis, LAMSADE, Université Paris Dauphine, Paris, 1992.

[ZD00] C. Zopounidis and M. Doumpos. PREFDIS: a multicriteria decision support system for sorting decision problems. *Computers & Operations Research*, 27(7-8):779–797, June 2000.

[ZD02] C. Zopounidis and M. Doumpos. Multicriteria classification and sorting methods: A literature review. *European Journal of Operational Research*, 138:229–246, 2002.