Computers & Operations Research 66 (2016) 67–80

Contents lists available at ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

Approaches for inequity-averse sorting

Özlem Karsu

Department of Industrial Engineering, Bilkent University, Ankara, Turkey

ARTICLE INFO

Available online 24 August 2015

ABSTRACT

In this paper we consider multi-criteria sorting problems where the decision maker (DM) has equity concerns. In such problems each alternative represents an allocation of an outcome (e.g. income, service level, health outputs) over multiple indistinguishable entities. We propose three sorting algorithms that are different from the ones in the current literature in the sense that they apply to cases where the DM's preference relation satisfies anonymity and convexity properties. The first two algorithms are based on additive utility function assumption and the third one is based on the symmetric Choquet integral concept. We illustrate their use by sorting countries into groups based on their income distributions using real-life data. To the best of our knowledge our work is the first attempt to solve sorting problems in a symmetric setting.

1. Introduction

Many practical problems involve the assignment of alternatives into predefined homogeneous groups. From a multicriteria point of view, this problem can be handled using multicriteria sorting or classification techniques. Multicriteria sorting refers to the cases where the groups are defined in an ordinal way starting from the ones including the most preferred alternatives to the ones including the least preferred alternatives while classification refers to the cases where these groups are defined in a nominal way [1,2]. Classification/sorting problems have applications in many areas including but not limited to medicine, pattern recognition, human resources management and financial management and economics [1].

In this paper we consider multi-criteria sorting problems where the criteria are like. In such problems each alternative corresponds to an allocation of an outcome over multiple entities and the decision maker (DM) has equity concerns. Considering the outcome level allocated to each entity as a criterion makes the problem a multicriteria decision making problem yet such problems differ from the classical MCDM problems discussed in the literature. First of all, in problems with equity concerns, we assume that the entities are indistinguishable to the DM, that is, the identities of the entities do not affect the decision. We call this property *anonymity*, *impartiality* or *symmetry*. The equity concerns should be incorporated into the preference model and this is achieved using a well-known axiom called the Pigou–Dalton principle of transfers from the inequality measurement literature. Such equity concerns arise in many real life decision making settings (see [3] for a recent review of inequity averse optimization in operational research). Potential applications of sorting problems with equity concerns include policy decision making and grouping countries with respect to their welfare, which is defined as a function of income distribution. For example, in health care policy decision making, the policy makers may consider a set of health care resource allocation policies each of which is associated with a distribution of the health outcome over different population groups. They may want to sort the feasible policies into groups such as *acceptable policies*, *intermediate policies that need further discussion*, and *to be rejected policies*

We consider three approaches to sort a given set of alternatives into (ordinal) classes. These approaches consider a set of utility functions in line with the preference information provided by the DM and sort the alternatives accordingly, taking equity concerns into account. Our work is related to two main disciplines in the literature, namely the economics literature on inequality measurement and the operational research literature on multi-criteria decision making as we explain below.

The economics literature on (income) inequality measurement deals with identifying desirable axioms that appropriate social welfare functions and inequality measures should satisfy. The axioms we use for the inequity-averse preference model are introduced in this literature. Also, the inequity-averse utility function forms we assume are the ones that have been discussed as appropriate inequity-averse social welfare function forms. There are some attempts in this literature to compare and rank a given set of income distributions (see e.g. [4]) based on a unanimity rule (an alternative is better than another if it has a higher functional value for all the functions in a predefined set of functions).

E-mail address: ozlemkarsu@bilkent.edu.tr

http://dx.doi.org/10.1016/j.cor.2015.08.004
0305-0548/

However, these attempts do not take any preference information into account. We extend these studies by designing sorting algorithms which take preference information from the DM into account and provide sorting conforming to the given preference information.

The initial discussions on multicriteria decision making literature consider problems which do not involve equity concerns and hence they are mostly based on rational preference models. Recently, equitable preference relations have been introduced by [5] and are further discussed by [6] in the multicriteria decision making framework. These works discuss incorporating equity concerns into the preference model in the context of multiobjective optimization. The equitable preference models and the underlying axioms that we use in this work are introduced in [5]. However, these works assume a multiobjective optimization setting and do not include any discussions on multicriteria sorting environments. To the best of our knowledge, our work is the first attempt to incorporate equity concerns in a multicriteria sorting environment, in that regard, it extends the current literature on equitable preferences.

The multicriteria sorting literature has so far focused on sorting problems with rational preference models. We based our first two approaches on two models discussed in this literature; however we extend them such that equitable preferences are considered as we explain below.

The three sorting approaches we use consider a set of utility functions in line with the preference information provided by the DM and sort the alternatives accordingly, assuming that the underlying preference model of the DM is equitable. These approaches differ from each other in terms of how the DM's utility (aggregation, social welfare) function is modeled. The first approach draws on a model suggested by [7] and assumes that the preferences of the DM can be represented by an additive utility function. This function is basically the sum of marginal utilities and the marginal utility functions are taken as piecewise linear functions. The second approach is an extension of the work suggested by [8], which is similar to the first one and assumes additive utility function. However, this approach is more general in the sense that it assumes nondecreasing marginal utility functions rather than piecewise linear ones. We extend the usage of these two models suggested by [7,8] to symmetrical settings by making the necessary modifications to the algorithms assuming that the DM has an equitable preference relation. As we will discuss later in detail, the original versions of these algorithms are designed and used for DMs with rational reference relations. We, however, consider equitable preferences, and hence will also assume symmetry and convexity properties for the preference model. These properties will restrict the set of utility functions we will consider. Specifically, we will assume that the marginal utility functions are concave and hence use piecewise linear concave marginal utility functions in the first approach and use nondecreasing concave marginal utility functions in the second approach. The third approach uses an ordered weighted averaging (OWA) method [5,6], which relates to the symmetric Choquet integral concept. These approaches assume utility functions that are equitable yet easy to use in a mathematical modeling setting. They have the potential to provide sufficient analysis while avoiding computational difficulty of other approaches that include more complex (e.g. nonlinear) utility function forms.

Our contributions can be summarized as follows:

- We propose multicriteria sorting methods for the case where the DM has equity concerns hence there is symmetry. To the best of our knowledge, this study is the first attempt to provide sorting mechanisms for multicriteria decision making (MCDM) problems with equity concerns.
- We propose variations of additive utility function based sorting approaches so that they can be used in symmetrical settings where equity is of concern.
- We propose another algorithm based on the symmetric Choquet integral concept, which draws on insights from the economics literature.
- We extend the current theory on equitable preferences by discussing them within a multicriteria sorting framework.

The outline of the paper is as follows: in the next section we briefly discuss the current literature in multicriteria sorting where there is no anonymity. In Section 3 we discuss sorting environments with equity concerns and discuss the term *equitable aggregation*. We propose sorting approaches based on three different equitable aggregation function forms. The first two are based on the well-known UTADIS method, which assumes an additive utility function and the third one is based on the ordered weighted averaging (OWA) method. We illustrate the use of these approaches by implementing them on a medium scale example problem. In Section 4 we provide the results of our computational experiments. We conclude the discussion and mention some future research directions in Section 5.

2. Sorting in classical MCDM problems

The multicriteria sorting problem is as follows:

A finite set of alternatives $A=\{a_1,a_2,\ldots,a_m\}$ is evaluated on a family of $g=\{g_1,g_2,\ldots,g_n\}$ n criteria. Let the index set of the alternatives be $I=\{1,2,\ldots,m\}$ and the criteria index set be $J=\{1,2,\ldots,n\}$. Given an alternative a_i, $g_j(a_i)$ shows the performance of alternative a_i in criterion j. The DM wants to sort the alternatives into q classes. Let C_k denote class k where C_1 is the most preferred and C_q the least preferred. Let the index set of the classes be $K=\{1,2,\ldots,q\}$.

The above problem is *the classical sorting problem*. There is a vast amount of literature on (classical) multicriteria sorting. We refer the interested reader to [1] for a review on multicriteria classification and sorting methods.

The sorting problems considered in this paper are different in the sense that they include like criteria, i.e. the criteria are measured using the same scale. Examples of such problems arise in settings where each alternative corresponds to an allocation profile of an output over multiple entities which are indistinguishable. For example in public service facility location problems, each alternative location corresponds to a distribution that shows the distances that customers travel to the service facility. Assuming that the customers are indistinguishable to the DM, in a two-customer setting he would be indifferent between the following two alternatives:

- A location that results in the distance vector (6,2) in which customer 1 travels 6 units of distance and customer 2 travels 2 units of distance.
- Another location that results in the distance vector (2,6) in which customers 1 and 2 travel 2 and 6 units of distance, respectively.

We call the MCDM problems that involve like criteria and hence involve symmetry *MCDM problems with like criteria*. In these problems all criteria are measured on the same scale (e.g. the distance scale in our location example).

Two main issues that every sorting methodology involves are the following [1]: the form of the criteria aggregation model and the methodology employed to define the parameters of the model.

Some of the criteria aggregation models are *Outranking relations, decision rules* and *utility function approach* [1] . Outranking relations method compares alternatives based on their "concordance" and "discordance" measures. We conclude that an alternative a_i outranks alternative a_j if there are enough arguments to confirm that a_i is at least as good as a_j (concordance), while there is no significant reason to refute this statement (discordance) [1]. In the sorting environment, the DM is asked to determine reference profiles that represent different classes. The alternatives are compared to these reference profiles and assigned to classes accordingly. For example, if an alternative's concordance and discordance measures are above and below the corresponding threshold values respectively, the alternative is concluded to outrank the reference profile and hence can be assigned to the class represented by the reference profile or a better class (see [1] for details). In decision rules method, a preference model is constructed based on a set of decision rules. The alternatives are sorted based on this preference model (see [9–11] for more information).

Utility function approaches assume that the DM's preferences can be represented by implicit utility (value) functions. Different forms are assumed for these utility functions.

One of the most common forms assumed is the additive utility function form $U(g(a_i)) = \sum_{j=1}^{n} u_j(g_j(a_i)) \in [0,1]$, where $u_j(.)$ is the marginal utility function for criterion j (see [12,8] for some recent discussions). The family of UTADIS methods is based on this additivity assumption. In its simplest form, the sorting is based on comparing the utility values of alternatives to the utility thresholds that define the (lower) bounds for each class. The methods that assume piecewise linear marginal utility functions solve a linear model with decision variables corresponding to the utility threshold values and the weights (or marginal utility intervals) for partitions. The objective is minimizing the classification errors on the reference assignments made by the DM. These errors can be defined in various ways such as the number of misclassifications. The optimal parameter values obtained from this model are used to estimate utilities of the whole set of alternatives.

Other utility function forms such as linear [13], quasiconcave [14], and Tchebycheff [15] are also considered.

In this paper we consider utility function approaches as the criteria aggregation method. The sorting approaches we use involve the following two steps:

1. Decision maker's providing some information on preferences.
2. Assignment of alternatives to their classes based on the DM's given judgements.

We now discuss these two steps in detail.

1. Decision maker's providing some information on preferences: in our algorithms we consider preference information that involves holistic judgements of the decision maker. The DM assigns a set of reference alternatives to their classes. This method is called *preference disaggregation* or *indirect elicitation* [16] method. In terms of the timing of interaction, we use prior articulation of judgements. That is, the DM is given the reference set at the beginning and asked to sort the alternatives in this set. We also consider the case where the DM is requested to make a predetermined number of reference assignments to each class. We do not consider an interactive setting but it is straightforward to design the corresponding interactive approaches that gather example assignments throughout the solution process.

2. Assignment of alternatives to their classes based on DM's given judgements: once the DM provides information, our sorting approaches find the worst and the best class that an alternative can belong to, which is consistent with the provided information. Note that most of the early UTADIS methods apply a different method. They predict the model parameters by finding the optimal values of a mathematical model, which minimizes a predefined optimality criterion. They then use these optimal parameters to sort the other alternatives into classes. The criterion can be defined in various ways such as the number of misplaced alternatives by the model (sorting error) or the magnitude of violations. We note here that, most of these models have alternative optimal solutions, i.e. different parameter settings minimize the criterion. Moreover, if a set of "optimal" parameters based on restricted preference information are chosen and applied to get a final sorting for the whole set of alternatives, there is no guarantee that this setting would be the "correct" setting. Hence we follow the idea used in [13,7,8,17] and provide worst and best classes that alternatives can belong to given the preference information.

3. Multicriteria sorting in environments with equity concerns

In this part we discuss the multicriteria sorting problems where the DM has equity concerns. To the best of our knowledge there are no sorting approaches discussed in the literature which are specifically designed for such cases where equity is of concern.

We consider a finite set of explicitly given alternatives and use the same notation as before: a_i denotes alternative i and $g_j(a_i)$ denotes the performance of alternative a_i in criterion (outcome) j. We denote the vector of criteria (outcome) values for alternative a_i with $g(a_i)$. That is, $g(a_i) \in G$ ($G \subset \mathbb{R}^n$) is the image of a_i in the criteria space. Note that $g(a_i)$ (or g^i) denotes the distribution over which we want to ensure equity, i.e. jth entity in distribution i gets $g_j(a_i)$ (or g^i_j).

Unlike a classical MCDM problem we consider a single outcome type, hence all the criteria are measured in the same scale with the same unit. The allocation of this outcome over multiple entities makes the problem a multi-criteria problem. To illustrate the data setting assumed in this paper consider a health care decision making problem in which there are five potential health care projects, each of which is associated with a distribution of benefits to three different population groups (for simplicity assume that the groups are of equal size). These groups may be constructed based on geographical location of the users of the healthcare system (e.g. different neighborhoods), or based on some other demographic factor (e.g. age, income level). In this small example $m=5$, $n=3$, $g(a_1)=g^1=(10,30,40)$, $g(a_2)=g^2=(25,15,25)$ and so on (see Table 1).

We assume that the DM has a preference model in which the weak preference relation $\preceq$ (with the corresponding strict preference and indifference relations denoted as $\prec$ and $\sim$, respectively) satisfies the following axioms (see [5,6] for a more detailed discussion of these axioms):

For any vector (alternative in the criteria space) $g \in G$

1. *Reflexivity* (R) $g \preceq g$ for all $g \in G$.

2. *Transitivity* (T) If $g^1 \preceq g^2$ and $g^2 \preceq g^3$ then $g^1 \preceq g^3$ for all $g^1, g^2, g^3 \in G$.

Table 1
Illustrative example.

Project (alternative a_i)	Benefits to groups		
	G1	G2	G3
1	10	30	40
2	25	15	25
3	5	50	50
4	15	15	35
5	30	40	10

3. *Strict monotonicity* (SM) $g \prec g+\varepsilon \mathbf{e}_k$ for $\varepsilon > 0$, where $\mathbf{e}_k$, is the kth unit vector in $\mathbb{R}^n$.

4. *Impartiality* (*IM*): $(g) \sim \Pi^k(g)$ for all $k = 1, \ldots, n!$, for all $g \in G$, where $\Pi^k(g)$ stands for an arbitrary permutation of the g vector.

This axiom ensures that the identities of the entities are not important and do not affect the decision. In our small example $g^1 \sim g^5$ (that is, $g(a_1) \sim g(a_5)$) due to symmetry.

5. *Pigou–Dalton principle of transfers* (*PT*): $g_j > g_k \Rightarrow g \prec g-\varepsilon \mathbf{e}_j+\varepsilon \mathbf{e}_k$, for all $g(a_i) \in G$, where $0 < \varepsilon < g_j - g_k$, where $\mathbf{e}_j$, $\mathbf{e}_k$ are the jth and kth unit vectors in $\mathbb{R}^n$.

This axiom ensures that any transfer from a relatively well-off entity to a worse-off one (without changing the relative positions of these two entities with respect to each other) results in a more preferred allocation. In our example, $g^4 \prec g^2$ due to PT since g^2 could be obtained by transferring 10 units of benefit from G3 to G1 in g^4 .

The preference relations that satisfy axioms R, T, SM, IM, and PT are called *equitable rational preference relations*. Using equitable rational preference relations, the relations of *equitable dominance*, *equitable indifference* and *equitable weak dominance* can be defined as follows [5]:

Definition 1. For any two criteria vectors g^1 and g^2,

$g^1 \prec_e (/\preceq_e/\sim_e) g^2$ (g^2 equitably dominates/equitably weakly dominates/is equitably indifferent to g^1) iff $g^1 \prec (/\preceq/\sim) g^2$ for all equitable rational preference relations $\preceq$. Equitable dominance is also called generalized Lorenz dominance (see [4]).

Definition 2. An alternative is equitably efficient if there is no alternative that equitably dominates it.

Following [5], we can introduce the ordered vector and cumulative ordered vector for an alternative g as follows:

Definition 3. Given $g \in \mathbb{R}^n$, let $\overrightarrow{g}$ denote the permutation of g such that $\overrightarrow{g}_p$: $\overrightarrow{g}_1 \le \overrightarrow{g}_2 \le \cdots \le \overrightarrow{g}_n$. $\overrightarrow{g}$ is called the ordered vector of g and $\overrightarrow{\mathbb{R}} = \{\overrightarrow{g} : g \in \mathbb{R}^n\}$ is called the ordered space.

Definition 4. Given $g \in \mathbb{R}^n$, let $\Theta : \mathbb{R}^n \to \mathbb{R}^n$ be the cumulative ordering map defined as follows:

$\Theta(g) = (\theta_1(g), \theta_2(g), \ldots, \theta_n(g))$ where $\theta_j(g) = \sum_{i=1}^{j} \overrightarrow{g}_i$ for $j = 1, 2, \ldots, n$. $\Theta(g)$ is called the cumulative ordered vector of g.

Theorem 5 (*Kostreva and Ogryczak [5]*). *For any two alternatives g^1 and g^2,*

$g^1 \prec_e g^2 \iff \Theta(g^1) \le \Theta(g^2)$ for all $j \in J$ where at least one strict inequality holds.

$g^1 \preceq_e g^2 \iff \Theta(g^1) \le \Theta(g^2)$ for all $j \in J$.

This theorem shows the relation between rational (vector) dominance that is used in the classical MCDM literature and the equitable dominance. An alternative equitably dominated another one if and only if its cumulative ordered vector rationally dominates the latter's cumulative ordered vector.

We will refer to the aggregations that respect axioms R, T, SM, IM, and PT as *equitable aggregations*.

Definition 6. An equitable aggregation function is a function $U : \mathbb{R}^n \to \mathbb{R}$ for which the following holds: $g^1 \prec_e (/\preceq_e/\sim_e) g^2 \Longrightarrow U(g^1) < (/\le/=) U(g^2)$.

An equitable aggregation function should be strictly increasing (due to SM), symmetric (due to IM) and should satisfy PT. All equitable aggregation functions are Schur-concave functions, which are symmetric by definition [6,3]. Schur-concavity relates to more familiar concavity concepts in the following way: all symmetric (strictly) quasi-concave and symmetric (strictly) concave functions are (strictly) Schur-concave [3].

Different Schur-concave utility functions such as symmetric additive concave (that is $U(g(a_i)) = \sum_{j=1}^{n} u_j(g_j(a_i))$ where $u_j(g_j(a_i)) = u(g_j(a_i))$ for all j and $u(.)$ is concave), symmetric concave and symmetric quasi-concave functions have been discussed as appropriate forms of inequity averse social welfare (aggregation) functions in the economics literature (see e.g. [4,18–20]). Moreover, [6] note that increasing functions of cumulative ordered outcomes can be used to obtain equitably efficient solutions in a multi-objective optimization context (based on Theorem 5, the alternatives whose cumulative ordered vectors are (rationally) nondominated will be equitably efficient). Specifically, the weighted sum function $(\sum_{j=1}^{n} w_j \theta_j(g))$ provides a family of linear aggregations over the cumulative ordered vectors, which can be converted to Schur-concave functions of the original vectors (namely the OWA functions), as we will show in Section 3.2. Kostreva and Ogryczak [6] suggest using these linear functions of the cumulative ordered vectors as scalarization functions since maximizing this function using different weights will (possibly) result in different equitably efficient solutions. In this study, we use these functional forms in a sorting environment, in which we restrict the feasible weight space using the DM's preference information.

In this paper, we consider two subsets of the set of Schur-concave functions: additive concave functions and linear aggregation functions over the cumulative ordered vectors. Note that these functions are symmetric concave (symmetric quasi-concave), hence they respect the following convexity axiom for the preferences:

6. *Convexity* (*C*): $g^1 \preceq g^2$ and $g^3 = \alpha g^1 + (1-\alpha) g^2$, for a real α : $0 < \alpha < 1 \Longrightarrow g^1 \prec g^3$.

The convexity axiom is not necessary but sufficient for a preference relation that satisfies R, T, SM and IM to be equitable. This is because C together with IM imply that PT holds. To sum up, the two equitable utility function forms we consider are as follows:

- An *additive* function, defined as the sum of concave marginal utility functions. These marginal utility functions will be the same for each criterion since we have impartiality. In the first model we assume piecewise concave marginal utility functions and in the second one we relax this assumption and assume nondecreasing concave functions. We assume that these functions are concave and the underlying preference relation satisfies the convexity axiom and hence ensure an equitable rational preference model.
- A *linear* utility function over the cumulative ordered vectors. This is an OWA based approach as discussed in Section 3.2.

3.1. Additive utility function based approach

Additive utility function based approaches have been used in classical sorting problems [7,8,17]. In order to ensure equitability, we make two main modifications to the existing models that are designed for classical sorting settings. First, impartiality implies that $U(g(a_i)) = \sum_{j=1}^{n} u_j(g_j(a_i))$ where $u_j(g_j(a_i)) = u(g_j(a_i))$ for all j. That is, the marginal utility function for each criterion (marginal utility function of each entity) is the same. Second, we ensure that the utility function respects equitable preferences by assuming that $u(g_j(a_i))$ is concave.

We first use concave piecewise linear form for marginal utility functions, which is an extension of the method suggested for classical MCDM sorting problems in [7].

Compared to the model suggested by [7], we have the following differences: we assume that u is the same for all criteria, as we do not distinguish entities with respect to their utility function. We

assume that u is concave and approximate it using piecewise linear approximations. This implies a further restriction on the parameters of the piecewise linear function. The weight vector corresponding to the slopes of the intervals will be decreasing. We also assume that a limited amount of preference information is taken at the beginning, while they use an interactive approach. Also, as in [8,17], we use linear models rather than MILP models used in [7].

We approximate the concave marginal utility function u, using piecewise linear approximation. Let the number of partitions used for the piecewise linearization be P. Let b_p denote the *lengths* of the partitions used for piecewise linear approximation with slopes w_p for $p=1,2,\ldots,P$. We partition the interval so that $\sum_{p=1}^{P} b_p = Max_{i,j}\, g_j(a_i)$ (maximum outcome value in the set) and the first interval is between $Min_{i,j}\, g_j(a_i)$ (minimum outcome value in the set) and b_1. Let b_j^i show *the starting point* of the interval to which $g_j(a_i)$ belongs and r_{ij} the corresponding interval number. Fig. 1 shows the graph of the marginal utility function and the parameters we discuss for an example case where $P=3$. As an example, point $g_3(a_4)$ is shown. For this point $b_3^4 = b_1 + b_2$ and $r_{43} = 3$.

We now discuss our algorithm, which is a variation of the algorithm used in [7] for the settings where we have equity concerns. These concerns are incorporated by changing the modeling of the utility function. Let $R \subset A$ be the set of reference alternatives assigned to classes by the DM. Consider the following model for an alternative a_t in A/R and class C_h. Note that the s and γ values are two parameters and that the marginal utility values are scaled so that $U \in [0,1]$

Model 1(a_t, C_h)

Max ε

$$v_i = \sum_{j=1}^{n}\left(\sum_{p=1}^{r_{ij}-1} w_p b_p + (g_j(a_i) - b_j^i) w_{r_{ij}}\right) \forall a_i \in A \tag{1}$$

$$w_p - w_{p+1} \geq \gamma \text{ for } \quad p = 1,2,\ldots,P-1 \tag{2}$$

$$n\sum_{p=1}^{P} w_p b_p = 1 \tag{3}$$

$$u_k - u_{k+1} \geq s \text{ for } \quad k = 1,2,3,\ldots,q-2 \tag{4}$$

$$u_{q-1} \geq s \tag{5}$$

$$u_1 \leq v_i, \forall a_i \in C_1 \tag{6}$$

$$u_k \leq v_i \leq u_{k-1} - \varepsilon, \quad k = 2,\ldots,q-1, \forall a_i \in C_k \tag{7}$$

$$v_i \leq u_{q-1} - \varepsilon, \forall a_i \in C_q \tag{8}$$

$$v_t \leq u_h - \varepsilon \tag{9}$$

$$\varepsilon \geq 0 \tag{10}$$

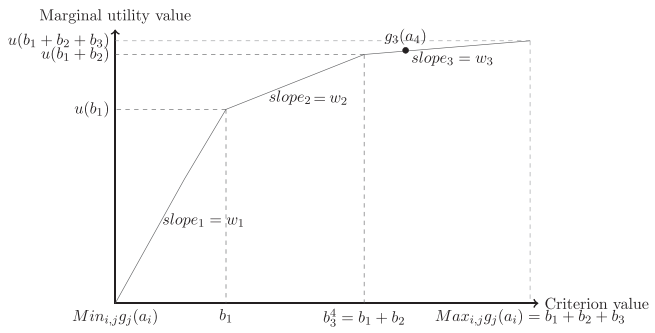

Fig. 1. Marginal utility function.

$$v_i \geq 0 \forall a_i \in A \tag{11}$$

$$w_p \geq 0 \forall p \tag{12}$$

The variables of the model are as follows:

v_i is the estimated utility value for alternative a_i, u_k is the estimated lower (upper) threshold value for class C_k (C_{k-1}) and w_p is the slope for partition p. This model has $m+p+q$ decision variables.

The model checks whether there is a sorting which is consistent with the provided reference assignments and which assigns alternative a_t to a class worse than C_h.

Constraint set 1 assigns a value to each alternative based on the assumption on the form of the marginal utility functions. Constraint set 2 ensures that the weights are decreasing (hence we have piecewise concave marginal utility functions). Constraint set 3 is for normalization and ensures that the utility values are all in [0,1]. Constraint sets 4 and 5 ensure that the utility thresholds of consecutive classes are sufficiently far way from each other. Constraint sets 6, 7 and 8 incorporate the provided information by the DM and ensure that the values of the alternatives that are already assigned to classes by the DM are within the limits of these classes. Constraint 9 forces the value of alternative t (v_t) to be less than the utility threshold of class h. If this is not possible given the preference information and the other constraints, we can conclude that the utility of alternative t must be above the threshold and the worst class that alternative t can be in is h. That is, if Model 1 is infeasible for any (a_t,C_h), then there is no additive utility function that satisfies the constraints and places alternative a_t to a class which is worse than C_h. Hence the worst class that a_t can be in is C_h.

Similarly one can formulate Model 2 (a_t,C_h) by changing the constraint $v_t \leq u_h - \varepsilon$ as $v_t \geq u_{h-1}$. Model 2 checks whether there is a feasible solution where alternative a_t is assigned to a class better than C_h. If Model 2 is infeasible we conclude that the best class a_t can be in is C_h.

Parameter γ determines the "degree" of concavity we impose on the marginal utility function. The larger the value of γ, the higher the level of concavity we assume. When γ is increased, a smaller set of utility functions is considered while making the sorting decisions. This results in the algorithm to make more assignments to a single class or restrict the number of classes that an alternative can belong to. In that sense, choosing a large value for γ might be attractive. However, if the underlying value function of the DM is not as concave as assumed, this might result in misclassification. Choosing a suitable value for γ is left to the decision analyst (DA). If the DA has the chance to interact with the DM, a relatively large γ value might be used and the results could be presented to the DM. If the DM is not satisfied with the assignments, smaller values for γ could be tried. Another approach would be presenting the results for different choices of γ and let the DM decide on the sorting that better reflects his preference model.

It is also possible to include further restrictions on weights if such information is available.

The sorting algorithm. The algorithm suggested in [7] picks a not-yet assigned alternative and starting from the worst class solves the corresponding version of model 1, which does not account for equity considerations, until it finds an infeasible case. In this way, the algorithm finds the worst class an alternative can be in. It also solves the corresponding version of model 2 to detect the best class an alternative can be in. If the best and worst classes the alternative can be assigned are not the same, the DM is asked to place the alternative into a class between the worst and the best classes the model finds.

Our sorting algorithm considers the case where the DM provides the reference assignments at the beginning of the algorithm. Then, using this information the algorithm returns the worst and best classes alternatives can be in. Note that it is possible to design an interactive algorithm but we prefer to see the extent to which we can narrow down possible assignments to classes given some limited information.

Below we give a description of the sorting algorithm. We keep the best and worst categories for the alternatives in arrays *ABEST* and *AWORST*, respectively. The algorithm assumes that a set of reference assignments has already been made by the DM.

We now provide an example of sorting countries into classes with respect their welfare levels using Algorithm 1, where the social welfare (utility) is assumed to be a function of the income distributions of the countries.

Algorithm 1.

Step 1. Initialization. Set $ABEST[t]=1$ and $AWORST[t]=q$ for all $a_t \in A$. For each $a_t \in R$ set $ABEST[t]=AWORST[t]=$ the index of the assigned class for these alternatives.
Step 2. Choose the next alternative $a_t \in A\backslash R$. If there is none, go to Step 3. Set $h=0$.
  Step 2.1. Set $h=h+1$
  Solve Model 1 (a_t,C_h).
If infeasible and $h=1$ set $AWORST[t]=ABEST[t]=1$ and go to Step 2.
If infeasible and $h>1$ set $AWORST[t]=h$, $h=h+1$ and go to STEP 2.2.
If feasible and $h \leq AWORST[t]-2$, repeat this Step.
If feasible and $h>AWORST[t]-2$, $h=AWORST[t]+1$ and go to Step 2.2.
  Step 2.2. Set $h=h-1$.
  Solve Model 2 (a_t,C_h).
If infeasible set $ABEST[t]=h$ and go to Step 2.
If feasible and $h \geq ABEST[t]+2$ repeat this Step. Otherwise, go to Step 2.
Step 3. Stop and report ABEST and AWORST.

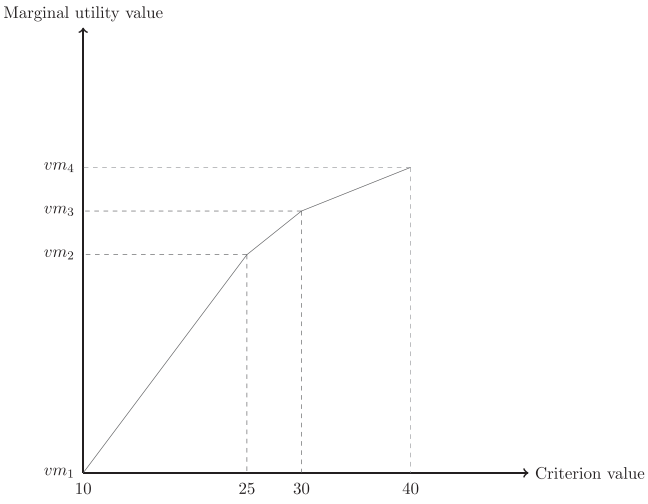

Fig. 2. Marginal utility function example.

Example 7. We use income distribution information of 66 countries from the World Bank [21] and UNU-WIDER (United Nations University- World Institute for Development Economics Research) [22] databases. We represent a country's income distribution using the quintile values. The quintile values are obtained as follows: for each country we take the percentage share of income that accrues to subgroups of population indicated by quintiles. We denote these percentage shares as S_i $i=1,\ldots,5$, where S_i% is the income share received by the ith 20% of the population. Given these percentage shares, for each country, we find mean income levels for each quintile, μ_i: $i=1,\ldots,5$ as follows:

$$\mu_i = \frac{Total\ Income * S_i}{Total\ Population * 20}, \quad i=1,\ldots,5$$

We use *Gross National Income* (*GNI*) [23] values to estimate *Total Income/Total Population*. Hence for each country we use a distribution vector of size 5 consisting of the mean income levels of each quintile. One can think of these μ_i values as the income levels of 5 representative people in the population. Table 9 in Appendix Appendix A shows the data. Suppose that we want to sort these distributions into 3 categories.

We assume that the social welfare function is of the form $U(g(a_i))=\sum_{j=1}^{n} u(g_j(a_i))$, where u is piecewise linear. We simulate $U(g(a_i))$ using randomly generated weights. We generate the weight values according to the following scheme:

Weight parameter generation scheme 1:

1. Generate random numbers from a uniform distribution $U(0,1)$.

2. Scale the generated weight values such that $n\sum_{p=1}^{P} w_p b_p = 1$ where $P=5$, where $b_p=(Max_{i,j}g_j(a_i)-Min_{i,j}g_j(a_i))/P$ for all p. That is, for piecewise linearization we use partitions of equal length.

3. Re-order the scaled weight values from maximum to minimum. Hence $w_1=Max_p\ w_p$ and $w_p=Min_p\ w_p$.

We calculate the utility values of the alternatives using the generated weights. We then divide the interval between the maximum and minimum utility values to q subintervals of equal length and use the end points of these intervals as the utility thresholds for classes. That is, $u_{q-k}=minutility+(k*((maxutility-minutility)/q))$ for $k=1,\ldots,q-1$, where *minutility* and *maxutility* are the minimum and the maximum utility values

Table 2
Best (B) and worst (W) classes of alternatives using approach 1.

Alt.	B	W	Alt.	B	W	Alt.	B	W	Alt.	B	W	Alt.	B	W	Alt.	B	W	Alt.	B	W
1	2	3	11	3	3	21	2	3	31	3	3	41	1	2	51	1	2	61	1	2
2	1	1	12	2	2	22	3	3	32	1	1	42	2	3	52	2	2	62	3	3
3	2	3	13	2	3	23	2	3	33	3	3	43	3	3	53	3	3	63	2	3
4	2	2	14	3	3	24	3	3	34	1	1	44	3	3	54	2	2	64	2	2
5	1	1	15	2	2	25	3	3	35	2	2	45	3	3	55	1	1	65	2	2
6	2	3	16	3	3	26	1	1	36	1	2	46	2	2	56	2	3	66	3	3
7	2	2	17	1	1	27	3	3	37	3	3	47	2	3	57	3	3			
8	2	2	18	2	3	28	2	3	38	1	2	48	2	3	58	2	3			
9	2	2	19	2	2	29	2	2	39	3	3	49	3	3	59	3	3			
10	3	3	20	2	3	30	3	3	40	3	3	50	1	1	60	3	3			

Table 3
Best and worst classes of alternatives using approach 2.

Alt.	B	W	Alt.	B	W	Alt.	B	W	Alt.	B	W	Alt.	B	W	Alt.	B	W	Alt.	B	W
1	1	3	11	2	3	21	2	3	31	2	3	41	1	3	51	1	2	61	1	2
2	1	1	12	1	3	22	2	3	32	1	2	42	2	3	52	2	2	62	2	3
3	1	3	13	1	3	23	2	3	33	3	3	43	2	3	53	2	3	63	1	3
4	1	3	14	3	3	24	2	3	34	1	2	44	3	3	54	2	2	64	2	2
5	1	1	15	1	3	25	2	3	35	1	3	45	2	3	55	1	1	65	1	3
6	2	3	16	3	3	26	1	2	36	1	3	46	1	3	56	1	3	66	2	3
7	1	3	17	1	2	27	3	3	37	2	3	47	2	3	57	2	3			
8	1	3	18	1	3	28	1	3	38	1	2	48	1	3	58	2	3			
9	1	2	19	2	2	29	1	3	39	2	3	49	2	3	59	2	3			
10	2	3	20	2	3	30	2	3	40	2	3	50	1	2	60	2	3			

in the feasible utility set. These reference alternatives are randomly selected and are $R=\{2,5,16,19,27,33,52,54,64\}$.

Based on the simulated utility function and the generated utility thresholds the DM provides the following reference assignments: $a_2, a_5 \rightarrow C_1, a_{19}, a_{52}, a_{54}, a_{64} \rightarrow C_2$ and $a_{16}, a_{27}, a_{33} \rightarrow C_3$. Given this information, the algorithm returns the assignments reported in Table 2 for $\gamma=0.005$ and $s=0.00001$.

Out of 57 alternatives, 37 are assigned to a single class and the number of possible classes for the remaining 20 alternatives is reduced to 2.

This approach uses pre-determined partitions (intervals) with equally distanced boundary points to represent the piecewise linear marginal utility function in the mathematical model as in [7]. Refs. [8,17] consider nondecreasing marginal utility functions and model these functions by using the attribute values of the alternatives as boundary points. Their model is more general as they do not restrict the analysis to piecewise linear functions. The models used in [8,17] are for the classical sorting problems and do not consider symmetric settings. We modify these models such that the utility functions respect equitable preferences by ensuring that the marginal utility functions are nondecreasing concave.

The general framework of the algorithm that is based on the additive utility function model with nondecreasing concave marginal utility functions is the same as Algorithm 1. Instead of models 1 and 2, we solve models 3 and 4 described below. We solve model 3 to check whether category C_h is the worst category that an alternative a_t can be in. In this model set L denotes the set of different output levels observed in the set of alternatives in the increasing order. For example when $n=3$ and $m=2$ with $a_1=(25,30,40)$ $a_2=(30,40,10)$ $L=\{10,25,30,40\}$. We denote the jth element of this set as L_j. For an alternative i, we denote the rank of the output level that entity j receives as La_j^i. The minimum value has a rank of 1. In our simple example, $La_3^2=1$ since $g_3(a_2)=10$ and 10 is the first level in set L. See Fig. 2 for an example graph

Model 3 (a_t, C_h)

$Max\varepsilon$

$$v_i=\sum_{j=1}^{n} vm_{La_j^i} \forall a_i \in A \tag{13}$$

$$\frac{(L_{l+2}-L_{l+1})(vm_{l+1}-vm_l)-(L_{l+1}-L_l)(vm_{l+2}-vm_{l+1})}{(L_{l+1}-L_l)(L_{l+2}-L_{l+1})} \geq \gamma \text{ for } l=1,\ldots,|L|-2 \tag{14}$$

$$vm_{l+1}-vm_l \geq 0 \text{ for } l=1,\ldots,|L|-1 \tag{15}$$

$$nvm_1=0 \tag{16}$$

$$nvm_{|L|}=1 \tag{17}$$

$$vm_l \geq 0 \text{ for all } l=1,\ldots,|L| \tag{18}$$

Constraint sets 4–11

The variables of the model are as follows: v_i and u_k are as in model 1. vm_l is the marginal utility value associated with the lth output level. This model has $m+|L|+q$ decision variables.

Constraint set (13) assigns values to the alternatives based on the assumption on the form of the utility function. Constraint sets (14) and (15) ensure that we have nondecreasing concave marginal utility functions. Constraint sets (16) and (17) are used for normalization purposes.

If model 3 (a_t,C_h) is infeasible, then the worst class that a_t can be in is h. Similarly, we solve a model 4 (a_t,C_h) by changing the constraint $(v_t \leq u_h-\varepsilon)$ as $v_t \geq u_{h-1}$. If this model is infeasible then the best class a_t can be in is C_h. We call the algorithm using these models Algorithm 2. Again, parameter γ shows the degree of concavity we assume for the marginal utility function.

This model considers a larger set of utility functions hence is more general than model 1. This comes with a possible increase in the computational burden since the number of intervals (decision variables) is expected to increase as the number of alternatives increases. We also expect Algorithm 2 to be more indecisive in terms of assigning alternatives to a single class since a larger set of functions is considered as compatible utility functions.

For the example setting, using the same underlying function and the same reference points, the results returned by this algorithm are as in Table 3. Out of 57 alternatives, 3 are assigned to a single class and the number of possible classes for 35 alternatives is reduced to 2. We set $s=0.00001$ as before and set γ to $(5*0.005)/(100*(|L|-1))$. 5 and 0.005 are the number of partitions and the γ value used in Example 7, respectively. Compared to Model 1, we use a smaller γ value here. This is because, the number of weights considered increases as $|L|$ increases so a smaller difference between consecutive weights should be ensured in Model 3.

3.2. Generalized Gini utility function based approach

This approach assumes that the utility function is of the form $U(g(a_j))=\sum_{j=1}^{n} w_j\theta_j(g(a_i))$. Since this function is an increasing function of the cumulative ordered vectors of the alternatives, it is an equitable aggregation and respects equitable dominance (see Theorem 5).

In this approach we do not assume additive utility over marginal utilities: we define the utility directly over the criteria (outcome) space (for all $g(a_i) \in G$). This social evaluation function is a *generalized Gini social evaluation function* [24] and is symmetric quasiconcave (hence Schur-concave).

Consider now a rank based utility function of the form $U(g(a_j))=\sum_{j=1}^{n} w'_j(\overrightarrow{g}_j(a_i))$ where $w'_j \geq w'_{j+1} \forall j$. Since $w' \in \overleftarrow{\mathbb{R}^n}$, in such a function the maximum weight is assigned to the entity that receives the minimum outcome, the second maximum weight is

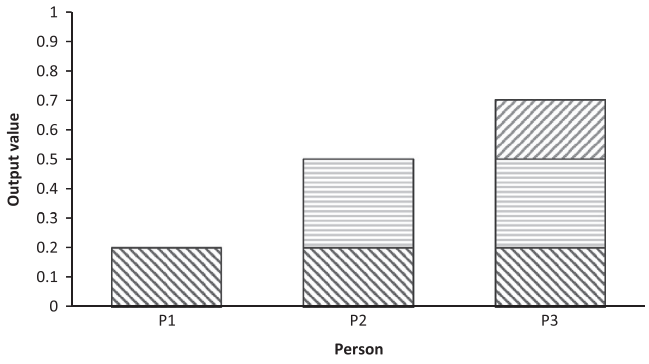

Fig. 3. Choquet integral example.

assigned to the entity that receives the second minimum outcome and so on. This makes the function inequity-averse. Note that while in the first two approaches the weights correspond to the slopes of the marginal utility function, in this function the weights $w' \in \overleftarrow{\mathbb{R}}^n$ are used to represent the relative importance of entities based on their rank within the allocation vector. This function is an OWA operator as defined below.

Definition 8. *Let $w'_1, w'_2, \ldots, w'_n$ be the set of weights such that $\sum_{j=1}^{n} w'_j = 1$. The OWA operator for a vector $g \in \mathbb{R}^n$ is defined as $OWA_{w'_1,\ldots,w'_n} = \sum w'_j \overrightarrow{g}_j$.*

Theorem 9 below shows that weighted sum of the elements of the cumulative ordered vector where weights are nonnegative is actually an ordered weighted averaging (OWA) function of the original vector with nondecreasing weights and vice versa. Hence there is a one-to-one correspondence between inequity averse OWA operators (these are OWA operators where $w' \in \overleftarrow{\mathbb{R}}^n$) and linear utility functions we defined over the cumulative ordered vectors. This relation is also discussed in [5].

Theorem 9 (*Kostreva and Ogryczak [5]*). *(i) For any utility function $U: U(g(a_i)) = \sum_{j=1}^{n} w_j \theta_j(g(a_i))$ where $w \in \mathbb{R}^n$, there exists $w' \in \overleftarrow{\mathbb{R}}^n$ such that $U(g(a_i)) = \sum_{j=1}^{n} w'_j \overrightarrow{g(a_i)_j}$, where $w' \in \overleftarrow{\mathbb{R}}^n$ and $\overrightarrow{g(a_i)_j}$ is the jth element of the ordered vector $\overrightarrow{g(a_i)}$* (such that $\overrightarrow{g(a_i)}_1 \le \overrightarrow{g(a_i)}_2 \le \cdots \le \overrightarrow{g(a_i)}_n$).

(ii) *For any utility function $U: U(g(a_i)) = \sum_{j=1}^{n} w'_j \overrightarrow{g(a_i)_j}$, where $w' \in \overleftarrow{\mathbb{R}}^n$, there exists $w \in \mathbb{R}^n$ such that $U(g(a_i)) = \sum_{j=1}^{n} w_j \theta_j(g(a_i))$.*

Proof. Part (i) Given $w \in \mathbb{R}^n$ define $w'_j = \sum_{h=j}^{n} w_h$ (note that $w' \in \overleftarrow{\mathbb{R}}^n$ holds). Then $U(g(a_i)) = \sum_{j=1}^{n} w_j \theta_j(g(a_i)) = \sum_{j=1}^{n} w_j \sum_{h=1}^{j} \overrightarrow{g(a_i)_h} = w_1 \overrightarrow{g(a_i)_1} + w_2(\overrightarrow{g(a_i)_1} + \overrightarrow{g(a_i)_2}) + \cdots + w_n(\overrightarrow{g(a_i)_1} + \overrightarrow{g(a_i)_2} + \cdots + \overrightarrow{g(a_i)_n}) = w'_1 \overrightarrow{g(a_i)_1} + w'_2 \overrightarrow{g(a_i)_2} + \cdots + w'_n \overrightarrow{g(a_i)_n} = \sum_{j=1}^{n} w'_j \overrightarrow{g(a_i)_j}$.

Part (ii) $w' \in \overleftarrow{\mathbb{R}}^n$ hence $w'_1 \ge w'_2 \ge \cdots \ge w'_n$. Define $w_j = w'_j - w'_{j+1}$ $\forall j$ and set $w'_{n+1} = 0$. $U(g(a_i)) = \sum_{j=1}^{n} w'_j \overrightarrow{g(a_i)_j} = (w'_1 - w'_2)\overrightarrow{g(a_i)_1} + (w'_2 - w'_3)(\overrightarrow{g(a_i)_1} + \overrightarrow{g(a_i)_2}) + \cdots + (w'_{n-1} - w'_n)(\overrightarrow{g(a_i)_1} + \overrightarrow{g(a_i)_2} + \cdots + \overrightarrow{g(a_i)_{n-1}}) + (w'_n)(\overrightarrow{g(a_i)_1} + \overrightarrow{g(a_i)_2} + \cdots + \overrightarrow{g(a_i)_n}) = \sum_{j=}^{} 1^n (w'_j - w'_{j+1}) \sum_{h=1}^{j} \overrightarrow{g(a_i)_h} = \sum_{j=1}^{n} w_j \theta_j(g(a_i))$.□

Theorem 9 shows that assuming a linear utility function over the cumulative ordered vectors is actually assuming an inequity-averse OWA utility function over the original vectors. Using inequity-averse OWA operators as social welfare functions has also been discussed in the economics literature (see e.g. [24]). An inequity-averse OWA operator is also a symmetric Choquet integral with a concave frequency distortion function [25] as discussed below.

Definition 10 (*Grabisch [26]*). Consider a finite set of criteria, that is the set of entities involved, $J = \{1,2,\ldots,n\}$ and its power set. A fuzzy measure μ defined on J is a set function $\mu: 2^{|J|} \longrightarrow [0,1]$ satisfying the following axioms: $\mu(\varnothing) = 0$, $\mu(J) = 1$, $A \subseteq B \Longrightarrow \mu(A) \le \mu(B)$.

In an MCDM setting, for any $A \subseteq M$ we can interpret $\mu(A)$ as the weight or degree of importance of the combination A of criteria. That is, in addition to the weights used for each criterion separately, we also use weights defined for any combination of the criteria [25]. In an impartial MCDM setting this would correspond to defining weights for any combination of the entities involved.

Definition 11. The Choquet integral of g with respect to μ is as follows : $C_\mu(g): \sum_{j=1}^{n} (\overrightarrow{g}_j - \overrightarrow{g}_{j-1}) \mu(A_{(j)})$ where $\overrightarrow{g}_0 = 0$ and $A_{(j)} = \{(j), (j+1), \ldots, (n)\}$ and $A_{(n+1)} = \varnothing$.

Theorem 12 (*Grabisch [26]*). *$OWA_{w'_1,\ldots,w'_n} = \sum w'_j \overrightarrow{g}_j = C_\mu(g)$ where μ is defined by $\mu(A) = \sum_{i=0}^{j-1} w'_{n-i}$, $\forall A: |A| = j$. That is, the weight of coalitions of size j is the sum of the weights corresponding to entities in the rank orders from $(n+1-j)$th to nth.*

Example 13. Consider the following case: we have three people (P1, P2 and P3) in the population with allocated output values 0.5, 0.2 and 0.7, respectively. Hence $g = (0.5, 0.2, 0.7)$ and $\overrightarrow{g} = (0.2, 0.5, 0.7)$. Suppose the weights for the OWA operator are $w'_1 = 0.7$, $w'_2 = 0.2$, $w'_3 = 0.1$. Then $OWA_{0.7,0.2,0.1} = 0.7*0.2 + 0.2*0.5 + 0.1* 0.7 = 0.14 + 0.1 + 0.07 = 0.31$.

Define the following: $\mu(1) = \mu(2) = \mu(3) = w'_3 = 0.1$ $\mu(\{1,2\}) = \mu(\{1,3\}) = \mu(\{2,3\}) = w'_2 + w'_3 = 0.3$ $\mu(\{1,2,3\}) = w'_1 + w'_2 + w'_3 = 1$.

Then we have $C_\mu = \overrightarrow{g}_1 * 1 + (\overrightarrow{g}_2 - \overrightarrow{g}_1) * 0.3 + (\overrightarrow{g}_3 - \overrightarrow{g}_2) * 0.1 = 0.2*1 + 0.3*0.3 + 0.2*0.1 = 0.2 + 0.09 + 0.02 = 0.31$. Fig. 3 illustrates the output values enjoyed by the coalitions.

Showing the relation between this type of utility function and the Choquet integral has some advantages in understanding the preference model structure that is assumed. Choquet integral has direct links with envy, hence it may be used as a way to bring envy into discussion by attempting to quantify it. In the example above, the contribution of the welfare of a coalition (group of entities) is the amount of outcome that is enjoyed by everyone in that coalition multiplied by the weight given to the coalition. In Example 13 all persons 1, 2 and 3 enjoy an outcome of 0.2, hence the contribution to the overall welfare is $0.2*\mu(\{1,2,3\})$, similarly, persons 2 and 3 both enjoy an extra of 0.3 and the contribution is $0.3*\mu(\{2,3\})$ (see Fig. 3). In the symmetric Choquet integral case that we assumed, the larger the cardinality of a coalition the larger the given weight and coalitions of the same cardinality have equal weights. As coalitions get smaller implying less people in the society enjoy the corresponding amount, the corresponding weight gets smaller. In that sense the model is envy-averse.

The general framework of the algorithm that is based on the generalized-Gini utility function will be the same as Algorithms 1 and 2. We solve models 5 and 6 described below. To check whether category C_h is the worst category that an alternative a_t can be in we solve model 5, which is as follows:

Model 5 (a_t, C_h)

$Max\varepsilon$

$$v_i = \left(\sum_{j=1}^{n} w_j \left(\sum_{k=1}^{j} g_k(a_i) \right) \right) \forall a_i \in A \tag{19}$$

$$\sum_{j=1}^{n} w_j * (j * Max_i \theta_n(g(a_i))/n) = 1 \tag{20}$$

Constraint sets 4 – 12

Table 4
Best and worst classes of alternatives using approach 3.

Alt.	B	W	Alt.	B	W	Alt.	B	W	Alt.	B	W	Alt.	B	W	Alt.	B	W	Alt.	B	W
1	2	3	11	3	3	21	2	3	31	3	3	41	2	2	51	2	2	61	2	2
2	2	2	12	2	2	22	3	3	32	1	2	42	2	3	52	1	1	62	3	3
3	2	3	13	2	3	23	2	3	33	3	3	43	3	3	53	3	3	63	2	3
4	2	3	14	3	3	24	3	3	34	1	2	44	3	3	54	2	2	64	2	2
5	2	2	15	2	2	25	3	3	35	2	2	45	3	3	55	1	1	65	2	2
6	2	3	16	3	3	26	1	1	36	2	2	46	2	2	56	2	3	66	3	3
7	2	3	17	1	2	27	3	3	37	3	3	47	2	3	57	3	3			
8	2	2	18	2	3	28	2	3	38	2	2	48	2	3	58	2	3			
9	2	2	19	2	2	29	2	2	39	3	3	49	3	3	59	3	3			
10	3	3	20	2	3	30	3	3	40	3	3	50	1	2	60	3	3			

Table 5
Results of algorithm 1, 3 classes.

Gamma	Scenario	m	Random						Middle						Boundary					
			Single		Two		CPU		Single		Two		CPU		Single		Two		CPU	
			Avg	Min	Avg	Min	Avg	Max	Avg	Min	Avg	Min	Avg	Max	Avg	Min	Avg	Min	Avg	Max
0	**1(10, 40, 50)**	50	13.8	11	30.6	14	1.81	1.89	13.2	11	29.8	17	1.78	2.00	34.2	26	15.6	10	1.53	1.67
		100	41.4	15	54.8	31	4.01	4.38	38.6	21	51	29	4.07	4.26	65.8	39	33.6	9	3.62	3.92
		150	78.8	62	68.2	54	6.59	6.96	60.6	43	80.8	76	6.87	7.16	105.8	48	42.8	27	6.31	6.94
		200	128.8	109	70.6	47	10.38	10.97	98.4	69	97.2	73	10.10	10.40	159	123	40.8	30	9.37	10.05
		250	132.4	75	114.2	83	14.43	15.15	161.4	120	88.2	51	13.59	14.23	218.6	204	31.4	10	12.80	13.20
		300	185.2	166	114	92	18.50	18.95	204.6	175	95.2	67	17.95	18.47	248.2	222	51.8	0	15.06	17.82
	2(20, 30, 50)	50	15.2	11	31.8	21	1.76	1.87	13.8	10	31.6	27	1.74	1.83	25.8	14	22.8	13	1.49	1.65
		100	38.6	26	55.2	47	3.84	4.07	39.6	25	50.6	39	3.86	4.15	64	34	33.2	17	3.43	3.92
		150	81.8	68	66.4	46	6.43	6.80	72.6	60	72.2	42	6.36	6.64	113.8	104	36.2	24	5.74	5.90
		200	126.4	109	73.4	55	10.05	10.47	114	94	83.6	57	9.41	9.88	163.8	143	36	18	8.48	8.66
		250	151.4	135	96.4	69	13.24	14.06	153.8	117	95.2	80	12.96	13.57	212.6	172	37	15	11.86	12.81
		300	205	186	94.4	74	17.24	17.83	191	155	106.6	81	16.90	17.83	247	230	52.8	22	15.96	16.16
	3 (33, 33, 33)	50	13.4	10	33	29	1.62	1.76	15.8	10	26.4	21	1.54	1.65	25.4	16	23.2	12	1.40	1.53
		100	44.4	21	49.8	43	3.47	3.88	49	38	44.2	26	3.26	3.42	69.4	59	30.2	17	3.04	3.28
		150	73	59	73.8	69	5.88	6.24	81.4	60	63.2	57	5.68	6.18	120.4	114	29.6	18	5.19	5.38
		200	111.2	94	88	67	9.04	9.83	126.2	109	70.6	47	8.07	8.28	147.8	138	52.2	33	7.97	8.16
		250	164.4	144	85.2	60	11.89	12.32	150.2	121	98.2	76	11.57	12.26	198.4	183	51.6	33	10.80	11.15
		300	182	157	116.4	92	15.94	16.66	201.2	196	98.6	94	15.02	15.71	235.8	226	64.2	47	14.66	15.02
0.005	**1(10, 40, 50)**	50	19	12	30.6	14	1.63	1.86	18	15	31	28	1.71	1.76	44.8	41	5	3	1.42	1.53
		100	51.8	30	48.2	24	3.66	3.95	52.8	40	47	33	3.89	4.27	84.6	74	15.4	3	3.44	3.60
		150	96.6	75	53.2	32	6.12	6.37	77.2	66	72	61	6.68	6.99	126.4	94	23.2	10	6.07	6.52
		200	147.2	128	52.6	32	9.50	9.84	118.8	101	80.6	53	9.92	10.28	174.2	150	25.6	16	9.08	9.64
		250	165.4	137	84.4	51	13.51	13.81	179.8	134	70.2	37	13.43	14.27	224.6	213	25.4	15	12.78	13.07
		300	205	184	94.8	65	17.71	18.14	230.4	208	69.6	45	17.71	18.21	262.6	235	37.4	22	17.17	17.68
	2(20, 30, 50)	50	19	13	30	16	1.56	1.65	19.4	14	29.8	26	1.63	1.73	35.2	32	14.8	8	1.39	1.47
		100	45.2	32	53	44	3.53	3.67	46.8	37	48.8	37	3.71	3.90	80.2	68	19.8	4	3.24	3.45
		150	94	85	55.8	40	6.13	6.27	84.8	68	63.2	33	6.23	6.55	124.4	114	25.6	14	5.64	5.74
		200	141	127	59	39	8.86	9.28	127	107	72.2	44	9.16	9.70	175	152	25	5	8.26	8.52
		250	166.6	147	82.8	52	12.50	13.15	172.2	134	77.8	64	12.71	13.32	223.4	183	26.6	14	11.84	12.92
		300	225	200	75	49	16.36	16.97	218	202	81.8	67	16.55	16.74	270.6	258	29.4	10	15.55	15.77
	3(33, 33, 33)	50	18.4	13	30.8	25	1.46	1.51	20.8	18	26.6	23	1.43	1.51	33.6	28	16.4	7	1.33	1.45
		100	52.6	29	46.2	35	3.12	3.49	59.4	48	37.4	22	3.16	3.37	79.6	74	20.4	13	2.93	3.07
		150	86.2	74	63.4	53	5.67	5.90	96.4	70	52.4	31	5.42	5.93	132.8	125	17.2	6	4.95	5.07
		200	130.8	108	69	39	8.18	8.66	138.8	113	60.2	21	7.93	8.25	171.8	154	28.2	14	7.66	7.96
		250	189.6	165	60.4	39	10.89	11.26	171	136	79	44	11.24	11.87	217	199	33	10	10.45	10.76
		300	208.2	178	91.4	56	14.82	15.62	222.8	213	77	68	14.76	15.37	261.4	246	38.6	25	14.26	15.05

Constraint set (19) assigns values to the alternatives based on the assumption on the form of the utility function. Constraint set (20) normalizes the utility values such that the maximum utility value is 1, which would be attained by a (possibly dummy) alternative that has the largest total value distributed in equal amounts to the recipients.

If model 5 (a_t,C_h) is infeasible, then the worst class that a_t can be in is h. Similarly, we solve a model 6 (a_t,C_h) by changing the constraint ($v_t \le u_h - \varepsilon$) as $v_t \ge u_{h-1}$. If this model is infeasible then the best class a_t can be in is C_h. We call the algorithm using these models Algorithm 3.

Example 14. Consider the problem described in Example 7. We now solve it assuming that the social welfare function is of the form $U(g(a_j)) = \sum_{j=1}^{n} w_j \theta_j(g(a_i))$. In order simulate DM's reference assignments, we use the following weight generation scheme.

Table 6
Results of algorithm 2, 3 Classes.

Scenario	m	Random						Middle						Boundary					
		Single		Two		CPU		Single		Two		CPU		Single		Two		CPU	
		Avg	Min	Avg	Min	Avg	Max	Avg	Min	Avg	Min	Avg	Max	Avg	Min	Avg	Min	Avg	Max
1 (10, 40, 50)	50	11.8	7	28.2	19	5.60	5.85	10.8	9	27.6	18	5.44	5.66	31.2	23	15.6	11	5.42	5.83
	100	37.8	17	45.6	32	29.54	30.30	26.2	19	48.4	34	27.17	28.70	58.8	33	35	17	28.58	30.97
	150	65.6	49	77.2	64	88.90	96.00	49.2	38	81.8	72	88.39	97.41	85	37	62	42	87.75	94.07
	200	102.8	76	93.8	74	194.14	205.03	79.4	50	105.8	74	172.10	189.63	123.2	64	71.6	46	193.55	206.31
	250	95	49	131.6	107	385.77	426.69	124.8	88	120.4	98	414.80	469.70	162	99	77.4	45	511.54	583.46
	300	121.2	90	156.8	137	616.26	693.72	119.6	63	165.4	135	601.45	733.68	177.2	148	119.2	80	795.68	896.22
2 (20, 30, 50)	50	11	5	25.6	22	5.40	5.90	12	6	29.4	25	5.35	5.71	24	15	22	13	5.02	5.34
	100	29	24	62	54	28.77	32.29	34	26	49.4	41	28.48	31.12	54.8	31	33.6	21	29.46	31.65
	150	68.8	52	73	65	86.99	90.29	55	45	82.6	71	90.15	100.48	91.6	79	56.2	46	90.69	94.47
	200	105.2	90	90.8	63	188.28	197.98	86.2	57	100.2	78	175.34	180.99	139.6	127	57	36	198.91	210.63
	250	134	115	106.8	85	412.20	447.53	133.8	97	110.8	90	401.78	455.88	196.6	173	51.4	0	455.83	537.03
	300	172.6	151	120	92	682.70	740.56	143.8	84	143.8	108	656.08	760.14	209.2	183	86.4	65	801.74	903.55
3 (33, 33, 33)	50	12.6	9	29.6	24	5.46	5.71	15.8	8	25	19	5.31	5.69	21.4	12	23.4	12	4.93	5.20
	100	33.6	14	50.2	46	29.80	31.01	41.4	33	45.2	34	30.83	31.93	54.4	28	39.8	28	30.91	31.70
	150	60.6	50	80	74	87.42	98.30	49.25	24	69.75	58	84.28	91.93	107.75	101	39.5	25	93.68	99.54
	200	91.8	68	100.2	84	191.12	205.66	103.4	80	89.6	63	192.27	208.15	114.4	88	80.8	59	198.97	207.93
	250	132.6	126	111.6	106	392.12	431.76	143.4	105	96.4	0	342.90	430.00	169.8	157	78.4	65	481.13	518.76
	300	143.8	134	143.4	124	630.48	756.23	174.8	146	120.8	90	766.02	806.13	188.6	175	109.4	95	786.80	866.72

Table 7
Results of algorithm 3, 3 classes.

Scenario	m	Random						Middle						Boundary					
		Single		Two		CPU		Single		Two		CPU		Single		Two		CPU	
		Avg	Min	Avg	Min	Avg	Max	Avg	Min	Avg	Min	Avg	Max	Avg	Min	Avg	Min	Avg	Max
1 (10, 40, 50)	50	11.4	7	36	27	1.62	1.76	11	9	31.4	26	1.70	1.83	34.2	29	15.8	9	1.33	1.37
	100	56.6	53	41	33	3.31	3.43	35	27	62.4	52	3.57	3.71	70.8	53	28.8	12	3.17	3.37
	150	81.4	56	67.8	35	5.78	6.04	65.8	49	77.8	56	5.79	6.04	123	103	27	10	5.31	5.62
	200	118	97	81.8	62	8.48	8.66	118.6	70	81	56	8.02	8.25	163.4	152	36.6	26	7.50	7.66
	250	156.6	135	93.2	60	11.5	11.75	155.8	130	94.2	78	11.31	11.65	207	190	43	23	10.23	10.53
	300	171	143	127.8	90	14.4	14.71	189.4	159	110.4	92	14.12	14.48	237.4	212	62.6	38	13.53	13.96
2(20, 30, 50)	50	14	10	32.4	23	1.59	1.67	12.2	7	35.2	32	1.60	1.72	31.8	22	17.6	3	1.23	1.37
	100	41.6	24	53.2	40	3.36	3.56	39.8	29	55	45	3.49	3.63	64.6	54	32.6	28	3.08	3.20
	150	77.2	67	70.6	63	5.63	5.79	77.2	68	69.8	56	5.28	5.44	113.8	97	35	25	4.89	5.29
	200	107.4	61	92	72	7.77	8.21	106.6	60	91	66	7.90	8.44	150.8	140	49.2	37	7.14	7.35
	250	150.4	125	99.2	79	10.75	11.04	145.8	102	102.8	79	10.53	10.86	190.4	165	59.6	29	9.53	9.81
	300	207	183	93	71	13.03	13.21	187.6	174	111.8	95	13.73	14.27	244	221	56	37	12.63	12.99
3 (33, 33, 33)	50	10.8	8	33.8	27	1.49	1.59	17	12	25.6	15	1.43	1.59	31	23	16.6	10	1.19	1.42
	100	47.8	27	45.6	36	2.97	3.35	48.6	40	42	34	2.89	3.09	69.4	58	29.8	12	2.67	2.79
	150	56.8	37	85.8	76	5.20	5.47	85.2	74	62.4	56	4.85	5.15	103	83	46.2	25	4.37	4.51
	200	99.6	85	97.6	84	7.22	7.58	126.6	107	71.6	56	6.81	7.04	138.6	122	61.4	41	6.48	6.83
	250	159	147	89.6	84	9.70	10.13	164.8	134	84.2	56	9.40	9.78	188.4	169	61.6	46	8.86	9.28
	300	184.8	150	113.8	103	12.09	12.64	191	169	108.4	91	12.10	12.54	245.2	228	54.8	35	11.19	11.42

Weight parameter generation scheme 2:

1. Generate random numbers from a uniform distribution $U(0,1)$.

2. Scale the generated weight values such that $\sum_{j=1}^{n} w_j * (j * Max_i \; \theta_j(g(a_i))/n) = 1$.

We find the utility values accordingly and set $u_{q-k} = minutility + (k * ((maxutility - minutility)/q))$ for $k = 1, \ldots, q-1$ as before. We use the same reference alternatives as in Example 7. Based on the simulated utility function and the generated utility thresholds the DM provides the following reference assignments: $a_{52} \to C_1, a_2, a_5, a_{19}, a_{54}, a_{64} \to C_2$ and $a_{16}, a_{27}, a_{33} \to C_3$. Given this information, the algorithm returns the assignments reported in Table 4 for $s = 0.00001$.

Out of 57 alternatives, 36 are assigned to a single class and the number of possible classes for the remaining 21 alternatives is reduced to 2.

4. Computational experiments

We performed experiments to see whether the proposed algorithms are computationally feasible and provide satisfactory

Table 8
Results for 4 classes.

Algorithm	Scenario	m	Single		Two		Three		CPU	
			Avg	Min	Avg	Min	Avg	Min	Avg	Max
1	**1 (5, 5, 40, 50)**	50	17.4	10	23.2	1	8.2	1	2.38	2.64
		100	52.6	44	44.4	36	3	1	5.17	5.3
		150	91.2	64	56.6	43	2.2	0	8.81	9.28
		200	118	99	81.2	49	0.8	0	13.47	13.81
		250	157.4	129	92	48	0.6	0	18.97	19.64
		300	221.4	152	77.6	52	1	0	24.77	25.99
	2 (20, 30, 25, 25)	50	15.8	13	15	2	16.8	4	2.32	2.61
		100	26.8	20	52.4	41	20.8	18	5.14	5.3
		150	65.4	46	73.2	46	11.4	2	8.33	8.52
		200	89.2	55	95.8	57	15	0	12.92	13.82
		250	140	114	88	45	22	0	17.71	18.11
		300	162.8	111	129.6	90	7.6	0	22.67	23.98
	3 (25, 25, 25, 25)	50	19	14	14.2	5	13.8	11	2.13	2.2
		100	41.4	36	42.4	35	14.4	9	4.40	4.57
		150	58.6	45	81.6	59	9.4	0	7.91	8.39
		200	93.4	70	85.6	77	20.8	11	11.73	12.62
		250	137	108	100.4	65	12.6	0	15.95	16.97
		300	167	131	129.4	99	3.6	0	20.98	22.26
2	**1 (5, 5, 40, 50)**	50	11.4	6	21.6	2	11.2	1	7.07	7.88
		100	36.2	22	49.8	45	12	5	36.39	37.71
		150	72.6	38	68	0	8.2	0	95.73	115.74
		200	97.2	65	93.6	0	9	0	202.46	244.34
		250	116.4	91	123.2	95	10.2	0	499.64	520.21
		300	179.4	125	111.4	99	9.2	2	965.24	1054.23
	2 (20, 30, 25, 25)	50	11.2	6	10.4	5	21	8	7.50	8.19
		100	18.6	15	35.8	30	37.6	32	38.62	39.98
		150	42	30	72.2	51	33.6	29	112.88	126.72
		200	48.4	33	85.2	56	60.2	38	245.41	269.16
		250	85	61	107.4	73	56	28	520.71	628.20
		300	106.4	79	141	113	51.8	7	1076.56	1172.65
	3 (25, 25, 25, 25)	50	14.2	10	12.6	9	14.4	4	7.12	7.43
		100	32	28	33.4	19	25.4	19	36.79	40.78
		150	36	24	73.2	52	37.2	20	110.71	116.59
		200	64.2	43	72.4	56	54.6	37	256.84	281.16
		250	100.2	79	98.2	55	49.6	10	525.10	571.40
		300	117.8	94	146.6	133	34.2	7	978.74	1194.70
3	**1 (5, 5, 40, 50)**	50	16	10	14.8	1	18.6	6	2.28	2.39
		100	40.2	25	44	31	15.6	0	5.02	5.59
		150	71.2	54	75.2	56	3.6	0	7.81	7.94
		200	103.4	76	93.6	71	3	0	11.39	12.29
		250	146.6	136	101.2	78	2.2	0	15.26	15.60
		300	203.8	172	95.8	74	0.4	0	18.70	19.17
	2 (20, 30, 25, 25)	50	10.6	6	9.8	2	22.6	17	2.31	2.40
		100	22.8	18	47.2	33	27	17	4.72	5.09
		150	44.8	29	78.4	56	26	7	7.59	8.19
		200	77	55	70.2	47	52.8	35	11.10	11.50
		250	117	106	96.2	80	36.8	1	13.75	14.98
		300	123.4	105	141	124	35.6	28	18.48	19.45
	3 (25, 25, 25, 25)	50	10.4	8	10.2	2	18.6	10	2.38	2.57
		100	35.4	28	40.8	32	22.2	14	4.48	4.71
		150	52	44	77.8	71	20.2	13	7.09	7.43
		200	76.4	68	94	69	29.2	9	9.65	10.23
		250	107.4	88	109.6	83	33	2	13.25	13.98
		300	150.4	122	125.2	95	24.4	0	16.96	17.80

results. We generate problem instances with 50, 100, 150, 200, 250 and 300 alternatives, where we set $n=5$ and $q=3$. Below we summarize our data generation scheme.

Step 0. Generate $g_j(a_i)$ values from a uniform distribution: $U(1,10)$.

Step 1. Generate weights using the weight parameter generation schemes described in Examples 7 and 14.

Step 2. Assign alternatives to their classes.

Step 3. Choose reference alternatives.

We assigned the alternatives to classes under three scenarios: in the first scenario 10% of the alternatives are assigned to class 1 (the best class), 40% are assigned to class 2 and 50% are to class 3. In the second scenario 20% , 30% and 50% of the alternatives are assigned to classes 1, 2 and 3, respectively. In the third scenario, we consider the case where all three classes are of equal size, i.e. each class contains 33% of the alternatives. For each parameter setting ($m=\{50,100,150\}$ and $scenario=\{1,2,3\}$), we generate 5 problem instances.

Table 9
Data for the example problem.[a]

Alt. a_i	$g_1(a_i)$	$g_2(a_i)$	$g_3(a_i)$	$g_4(a_i)$	$g_5(a_i)$	**Alt.** a_i	$g_1(a_i)$	$g_2(a_i)$	$g_3(a_i)$	$g_4(a_i)$	$g_5(a_i)$
1	3580	5350	7030	9240	19,010	34	5900	9920	14,040	19,760	39,690
2	2920	6480	10,490	16,630	39,200	35	2870	5220	7720	11,510	26,830
3	2370	3510	4610	6000	10,760	36	3230	6250	9840	15,510	36,980
4	3690	5580	7470	10,000	19,410	37	330	550	770	1100	2350
5	6310	9530	12,370	15,950	25,970	38	3190	6190	9640	14,900	41,090
6	640	1460	2530	4290	13,910	39	1130	1840	2590	3650	7510
7	3000	5070	7220	10,180	19,380	40	1310	2070	2890	4090	8140
8	1690	3790	6410	10,650	32,090	41	4770	7780	10,580	14,170	26,210
9	3300	6010	9180	13,870	33,690	42	1480	2390	3310	4700	10,920
10	180	230	300	410	840	43	240	440	630	920	2370
11	670	960	1340	1960	5270	44	290	420	550	740	1500
12	2850	5420	8440	13,230	39,520	45	1250	1720	2200	2880	5850
13	1130	2700	4810	8420	27,950	46	1970	4530	7860	13,230	37,140
14	90	140	210	320	780	47	980	2080	3340	5310	15,460
15	2340	4430	6850	10,830	29,920	48	1700	3610	5930	9430	24,050
16	460	830	1230	1800	3930	49	1100	1790	2690	4170	9900
17	7580	11,410	15,160	20,210	39,290	50	7230	11,410	15,500	20,920	40,130
18	1800	3480	5520	8760	23,950	51	5620	8960	11,910	15,810	27,960
19	1760	3590	5790	9180	26,050	52	5560	9310	13,530	19,960	47,540
20	1370	2880	4440	6680	16,580	53	250	450	690	1070	3430
21	1320	2570	3790	5500	11,750	54	5110	7580	9830	12630	21,000
22	420	780	1180	1750	3860	55	11,050	16,070	20,370	24,770	30,090
23	780	1660	2770	4500	13,320	56	1750	2640	3650	5200	12,120
24	310	510	740	1070	2260	57	480	790	1110	1540	3180
25	420	1180	2180	3770	11,110	58	1610	2880	4700	7910	24,140
26	8100	12,440	16,290	21,210	38,460	59	930	1290	1660	2180	4250
27	1610	2400	3230	4560	9720	60	210	410	600	870	1860
28	2080	3200	4390	6090	13,100	61	4050	7690	11,230	16,070	33,860
29	4620	6790	8860	11,670	21,170	62	360	590	850	1230	3120
30	960	1290	1690	2310	4670	63	3090	4470	5760	7400	12,210
31	870	1300	1760	2400	5150	64	3330	6270	9750	14,960	35,140
32	5520	9490	13,170	18,410	35,260	65	2930	5740	8780	13,030	29,280
33	110	190	260	360	740	66	1050	1580	2200	3120	6610

[a] All outcome values are divided by 5000 in the experiments for normalization purposes. This is to ensure that the outcome values are in the same range (less than 10) we used in our experiments with randomly generated data. One can also use the outcome values as they are, arranging parameters of the models accordingly.

We assume that the DM places 10% of the alternatives into their classes and the number of reference alternatives to be assigned to each class is determined before the algorithms begin. For example, when $m=50$, the DM is asked to place 1 alternative in class 1, 3 alternatives in class 2, and 1 alternative in class 3. In order to see the effect of the choice of the reference set, we investigate three different cases:

In the first case, the reference alternatives are chosen randomly. We refer to this case as “Random”.

In the second case, the reference set is chosen such that it includes the middlemost alternative(s) in each class. If the size of the reference set is greater than the total number of middlemost alternatives, the rest of the alternatives are chosen randomly, again ensuring that a predetermined ratio of reference alternatives per class is satisfied. We refer to this case as “Middle”.

In the third case the reference set is chosen such that it includes the worst alternative in class 1, the best and worst alternatives in class 2 and the best alternative in class 3. If the size of the reference set is greater than three, the rest of the alternatives are chosen randomly respecting the ratios. We refer to this case as “Boundary”.

We call the algorithms based on the additive utility with piecewise linear concave marginal utility functions, with general concave marginal utility functions and generalized-Gini utility functions Algorithm 1, Algorithm 2, and Algorithm 3 respectively. The algorithms are coded in Visual C++ and solved by a dual core (Intel Core i5 2.27 GHz) computer with 4 GB RAM. All models are solved by CPLEX 12.2. Tables 5–7 show the results for Algorithms 1, 2 and 3 respectively. We report the average and minimum values for the number of alternatives that could be assigned to a single class, and for the number of alternatives for which there are two possible classes. We also report the average and maximum solution times of the algorithms. The solution times are expressed in central processing unit (CPU) seconds. We set $s=0.001$ in all experiments. In order to see the effect of parameter γ we use two different levels (0 and 0.005, respectively) for Algorithm 1. Also in Algorithm 2, we set parameter γ to $(5*0.005)/(100*(-1))$. We also tried $\gamma=0$ (hence considered concave functions rather than strictly concave functions) but the results did not change significantly. We use the same problem instances (the same reference assignments) for Algorithms 1 and 2 in our experiments.

It is observed from Tables 5–7 that the algorithms perform well in terms of reducing the number of possible classes for the alternatives. For almost all of the alternatives, the number of possible classes is reduced to either one or two. As seen in Table 5 , as the value of γ increases, more alternatives are assigned to a single class. The choice of the reference set also has a noticeable effect on the amount of the reduction in the number of possible classes. If the reference set includes alternatives which are the worst and best alternatives in each class (boundary case), the performances of both algorithms increase in terms of the number of solutions assigned to a single class compared to a random selection of the reference set. Including middlemost alternatives of each class decreases the performance of the algorithms compared to the random selection, especially when the classes are not of equal size. Note that asking the DM to detect the best and worst alternatives in each class might be cognitively demanding. A possible approach would be estimating a utility function and

updating the estimate as the DM makes assignments. This function can be used to guess the best and worst alternatives of each class and such alternatives can be added to the reference set.

The solution times are not excessive for the problem sizes selected but they are expected to increase as the problems get larger. One can observe the increase in solution times when Algorithm 2 is used. This is mostly due to the increase in the number of decision variables. Moreover, as expected, the number of alternatives assigned to a single class is smaller in Algorithm 2.

In order to see the effect of increasing the number of classes on the performances of our algorithms, we performed experiments for the problem instances where the alternatives are sorted into 4 classes. We assigned the alternatives to classes under three scenarios: in the first scenario 5% of the alternatives are assigned to class 1 (the best class), 5% of the alternatives are assigned to class 2, 40% are assigned to class 3 and 50% are assigned to class 4. That is, we divided the best class in the first scenario of our previous analysis into two classes. In the second scenario 10%, 40%, 25% and 25% of the alternatives are assigned to classes 1, 2, 3 and 4, respectively. We obtained this scenario by dividing the worst class into two classes in the second scenario of our previous analysis. In the third scenario, we consider the case where all four classes are of equal size, i.e. each class contains 25% of the alternatives. For each parameter setting ($m = \{50, 100, 150, 200, 250, 300\}$ and $scenario = \{1, 2, 3\}$), we generate 5 problem instances.

We again assume that the DM places 10% of the alternatives into their classes before the algorithms begin. In this part we report the results for the case where the reference alternatives are chosen randomly. Table 8 shows the results of our experiments for $s = 0.001$ and the same γ values that are used for the 3 classes case for Algorithms 1 and 2.

It is observed in Table 8 that the solution times of all algorithms increase as the number of classes increases. Also, the performances of the algorithms seem to be worse than those of the previous case in terms of the number of alternatives assigned to a single class and the number of alternatives for which the number of potential classes is reduced to two. Some exceptions to this are observed in the larger problem instances.

The three utility function forms that we considered are easy to understand and have intuitive interpretations. Moreover, they respect the equitable dominance relation. They may prove valuable in providing support to the DMs faced with sorting decisions with equity concerns.

In this work we do not attempt a direct comparison between approaches (1–2) and 3 as the conceptual framework behind these approaches are different and each might be considered appropriate in different settings. In the first two approaches a (single) marginal utility function is assumed. One can think of this as the utility function of an entity. Inequity-aversion is ensured by enforcing a concave structure for these marginal utility functions. The overall utility of the allocation (the social welfare) is taken as the sum of these individual utilities. In that sense, they are utilitarian approaches. If the decision maker finds such a utilitarian framework appropriate for structuring his/her preference model then these two approaches could be used. In settings where the number of observed levels of the outcome is low, Algorithm 2 may be preferred as it is based on less restrictive assumptions on the form of the marginal utility function. If the number of different outcome levels is high, using a piecewise linear function with a predetermined number of equally sized intervals as in approach 1 may be preferred. In the third approach, however, the social welfare function is modeled as a linear function of cumulative ordered vectors, which implies an inequity-averse ordered weighted averaging function. Weights are assigned to the ordered outcome values rather than outcome values allocated to specific entities, which makes them rank dependent and the resulting function symmetric and non-additive. We ensure inequity-aversion by assuming that the weights are nondecreasing. Being symmetric Choquet integrals, such functions also explicitly assign weights to coalitions and quantify envy resulting from an allocation.

The proposed algorithms may be used as heuristic approaches for the case where the DM's utility function is Schur-concave. In that case if inconsistencies arise due to approximation of a nonlinear function with a linear one, one can interact with the DM and request him/her to assign the alternative leading to inconsistency. The corresponding assignment constraints can be ignored for the following iterations.

5. Conclusion

In this study we discussed multicriteria sorting problems with equity concerns and proposed three solution approaches based on different assumptions on the form of the utility function. The first two methods are based on the additive utility function assumption and the third approach uses a rank dependent utility function. The additive utility (social welfare) functional forms have been discussed in the economics literature. However, to the best of our knowledge, it is the first time they are used in a decision support framework for sorting purposes which includes preferences of the DM (for the problem types that we consider). Also, our models extend the current models in the multicriteria sorting literature such that equitable preferences are considered rather than rational preferences.

The appropriate form of the utility function can be determined through interactions with the DM. The first two approaches attempt to quantify individual utilities and then aggregate them to find the total utility in an allocation whereas the third avoids such a reduction to individual utilities. If the DM is more comfortable with conceptualizing the social utility as the aggregation of individual utilities and may provide information on the rate of increase in the entity's (recipients) utility as a result of an increase in the amount of good it receives, then the first or the second approach can be used. The second approach is more general since it assumes that the marginal utility functions are concave rather than piecewise linear concave. However considering a larger set of marginal utility functions may result in an increase in the computational burden as well as a decrease in the number of alternatives assigned to a single class or to a small number of classes. This is also observed in our computational experiments. If the DM finds it easier to provide information on the relative importance of the entities with respect to their rank in the allocation then the third approach would be appropriate.

We have shown that all approaches lead to tractable computations. Moreover, the algorithms avoid direct elicitation of the parameters, hence the DM's cognitive effort is kept as small as possible. The information from the DM is gathered by requesting reference assignments which is a more natural way than asking parameters directly. We investigated different strategies for the selection of the reference set. We observed that the performance increase in the algorithms is noticeable when the reference set includes the worst and best alternatives of classes.

Future research might consider the cases where the equitable utility functions are nonlinear. Also computational experiments and real life tests can be performed to compare the performances of an interactive approach in which the assignments are requested in a progressive way with the performance of the a priori approach we use in which the assignments are gathered at the beginning of the process.

Acknowledgments

The author is grateful to Professor Alec Morton for many helpful discussions and feedback and to the anonymous referee and the area editor for their insightful and constructive comments.

Appendix A. Data for the example problem

See Table 9.

References

[1] Zopounidis C, Doumpos M. Multicriteria classification and sorting methods: a literature review. Eur J Oper Res 2002;138(2):229–46.
[2] Chen Y, D.Kilgour M, Hipel KW. Multiple criteria classification with an application in water resources planning. Comput Oper Res 2006;33 (11) :3301–3323. Part special issue: Operations Research and Data Mining.
[3] Karsu Ö, Morton A. Inequity averse optimisation in operational research, http://dx.doi.org/10.1016/j.ejor.2015.02.035; 2015.
[4] Shorrocks A. Ranking income distributions. Economica 1983;50:3–17.
[5] Kostreva MM, Ogryczak W. Linear optimization with multiple equitable criteria. RAIRO Oper Res 1999;33:275–97.
[6] Kostreva MM, Ogryczak W, Wierzbicki A. Equitable aggregations and multiple criteria analysis. Eur J Oper Res 2004;158:362–77.
[7] Köksalan M, Ozpeynirci SB. An interactive sorting method for additive utility functions. Comput Oper Res 2009;36:2565–72.
[8] Greco S, Mousseau V, Slowinski R. Multiple criteria sorting with a set of additive value functions. Eur J Oper Res 2010;207(3):1455–70.
[9] Greco S, Matarazzo B, Slowinski R. Rough sets methodology for sorting problems in presence of multiple attributes and criteria. Eur J Oper Res 2002;138(2):247–59.
[10] Greco S, Matarazzo B, Slowinski R. Rough sets theory for multicriteria decision analysis. Eur J Oper Res 2001;129(1):1–47.
[11] Blaszczynski J, Greco S, Slowinski R. Multi-criteria classification—a new scheme for application of dominance-based decision rules. Eur J Oper Res 2007;181(3):1030–44.
[12] Greco S, Kadzinski M, Slowinski R. Selection of a representative value function in robust multiple criteria sorting. Comput Oper Res 2011;38(11):1620–37.
[13] Koksalan M, Ulu C. An interactive approach for placing alternatives in preference classes. Eur J Oper Res 2003;144(2):429–39.
[14] Ulu C, Köksalan M. An interactive approach to multicriteria sorting for quasiconcave value function. Naval Res Logist 2014;61(6):447–57.
[15] Soylu B. A multi-criteria sorting procedure with Tchebycheff utility function. Comput Oper Res 2011;38(8):1091–102.
[16] Jacquet-Lagreze E, Siskos Y. Preference disaggregation: 20 years of MCDA experience. Eur J Oper Res 2001;130:233–45.
[17] Kadzinski M, Tervonen T. Stochastic ordinal regression for multiple criteria sorting problems. Dec Support Syst 2013;55(1):55–66.
[18] Rothschild M, Stiglitz JH. Some further results on the measurement of inequality. J Econ Theory 1973;6(2):188–204.
[19] Dasgupta P, Sen A, Starrett D. Notes on the measurement of inequality. J Econ Theory 1973;6:180–7.
[20] Sen A. On economic inequality. Oxford: Clarendon Press; 1973.
[21] Available at ⟨http://data.worldbank.org/indicator/SI.DST.05TH.20⟩, [accessed 15.08.11]; 2011.
[22] Available at ⟨http://www.wider.unu.edu/research/Database/en-GB/wiid⟩ [accessed 15.08.11]; 2011.
[23] Available at ⟨http://data.worldbank.org/data-catalog/GNI-per-capita-Atlas-and-PPP-table⟩[accessed 15.08.11]; 2011.
[24] Weymark JA. Generalized Gini inequality indices. Math Soc Sci 1981;1:409–30.
[25] Marichal J-l. Aggregation of interacting criteria by means of the discrete Choquet integral. In: Calvo T, Mayor G, Mesiar R, editors. Aggregation operators: new trends and applications. Series: studies in fuzziness and soft computing, vol. 97. Heidelberg: Physica-Verlag; 2002. p. 224–44.
[26] Grabisch M. On equivalence classes of fuzzy connectives—the case of fuzzy integrals. IEEE Trans Fuzzy Syst 1995;3(1):96–109.