AN INTERACTIVE APPROACH FOR MULTI-CRITERIA SORTING PROBLEMS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

BURAK KESER

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
INDUSTRIAL ENGINEERING

APRIL 2005

Approval of the Graduate School of Natural and Applied Sciences.

Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Çağlar Güven
Head of the Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Prof. Dr. Murat Köksalan
Supervisor

Examining Committee Members

Prof. Dr. Nesim Erkip	(METU, IE)	
Assoc. Prof. Yasemin Serin	(METU, IE)	
Prof. Dr. Erdal Erel	(Bilkent U., MAN)	
Prof. Dr. Murat Köksalan	(METU, IE)	
Assist. Prof. Esra Karasakal	(METU, IE)	

ABSTRACT

AN INTERACTIVE APPROACH FOR MULTI-CRITERIA SORTING PROBLEMS

Keser, Burak
M. Sc., Department of Industrial Engineering
Supervisor: Prof. Dr. Murat Köksalan

April 2005

This study is concerned with a sorting problem; the placement of alternatives into preference classes in the existence of multiple criteria. An interactive model is developed to address the problem, assuming that the decision maker has an underlying utility function which is linear. A recent methodology, Even-Swaps, which is based on value tradeoff is utilized in the model for both making an estimation of the underlying utility function and generating possible dominance among the alternatives on which it is performed. Convex combinations, dominance relations, weight space reduction, Even-Swaps and direct decision maker placements are utilized to place alternatives in preference classes. The proposed algorithm is experimented with randomly generated alternative sets having different characteristics.

Keywords: multiple criteria decision making, sorting, even swaps

ÖZ

ÇOK KRİTER ALTINDA SIRALAMA PROBLEMLERİ İÇİN ETKİLEŞİMLİ BİR YAKLAŞIM

Keser, Burak
Yüksek Lisans, Endüstri Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Murat Köksalan

Nisan 2005

Bu çalışma, alternatiflerin çok kriter altında tercih sınıflarına yerleştirilmesi ile ilgilidir. Karar vericinin gizli fayda fonksiyonunun doğrusal olduğu varsayılarak, etkileşimli bir yöntem geliştirilmiştir. Değer ödünleşmeleri üzerine kurulu yeni bir metodoloji olan Eş-Takas yönteminden, hem karar vericinin gizli fayda fonksiyonunu tahminlemek hem de üzerinde uygulandığı alternatifler arasında olası bir baskınlık ilişkisi oluşturmak üzere faydalanılmıştır. Alternatifleri tercih sınıflarına yerleştirmek için konveks kombinasyonlar, baskınlık ilişkisi, ağırlık uzayı daraltılması, Eş-Takas ve karar vericinin doğrudan yerleştirmelerinden faydalanılmıştır. Önerilen algoritma, rastsal oluşturulmuş farklı karakterdeki alternatif kümeleriyle denenmiştir.

Anahtar Kelimeler: çok kriterli karar verme, sıralama, eş-takas

To my lovely family

ACKNOWLEDGMENTS

I would like to thank my thesis supervisor Prof. Dr. Murat Köksalan for his continuous support, guidance and patience throughout my work.

I am also thankful to Onur Aktuğ, Cenk Güray, Bulut Aslan, Ümit Sönmez, Engin Akyürek, and Gizem Karslı.

Finally my dear family; Elif, Remziye and Üzeyir Keser; this work would not be possible without their support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Problem Definition

The problem considered in this study is a multi-criteria decision making (MCDM), problem where the decision maker (DM) intends to group a set of alternatives into preference classes. The approach developed in this study will be more suitable for problems where the alternative set is large, and the number of criteria to consider is small.

The placement of alternatives into predefined classes or groups is referred to as classification or sorting problems depending on whether the groups are nominal or ordinal. The problem of sorting / classification has numerous practical applications, some of which are listed below (Zopounidis, C., Doumpos, M., 2002)

- Medicine: performing medical diagnosis through the classification of patients into diseases groups on the basis of some symptoms (Stefanowski and Slowinski, 1998; Tsumoto, 1998; Belacel, 2000; Michalowski et al., 2001).
- Pattern recognition: examination of the physical characteristics of objects or individuals and their classi-fication into appropriate classes (Ripley, 1996; Young and Fu, 1997; Nieddu and Patrizi, 2000). Letter recognition is one of the best examples in this field.
- Human resources management: assignment of personnel into appropriate occupation groups according to their qualifications (Rulon et al., 1967; Gochet et al., 1997).

- Production systems management and technical diagnosis: monitoring the operation of complex production systems for fault diagnosis purposes (Nowicki et al., 1992; Catelani and Fort, 2000; Shen et al., 2000).
- Marketing: customer satisfaction measurement, analysis of the characteristics of different groups of customers, development of market penetration strategies, etc. (Dutka, 1995; Siskos et al., 1998).
- Environmental and energy management, ecology: analysis and measurement of the environmental impacts of different energy policies, investigation of the efficiency of energy policies at the country level (Diakoulaki et al., 1999; Rossi et al., 1999; Flinkman et al., 2000).
- Financial management and economics: business failure prediction, credit risk assessment for firms and consumers, stock evaluation and classification, country risk assessment, bond rating, etc. (Altman et al., 1981; Slowinski and Zopounidis, 1995; Zopounidis, 1998; Doumpos and Zopounidis, 1998; Greco et al., 1998; Zopounidis et al., 1999a,b).

Below table (Zopounidis, C., Doumpos, M., 2002) shows some real-world applications of classification / sorting problems (Table 1). Some of these studies use real-world data for illustrative purposes in order to present the practical applicability of classification and sorting theory in real-world data sets. Other studies use real-world data for performance evaluation of selected methods originating from the developed methods with existing methods, most commonly originating from the field of statistics.

Table 1. Real world applications of classification and sorting problems

Application Area	**Studies**
Business failure prediction	Mahmood and Lawrence (1987), Gupta et al. (1990), Slowinski and Zopounidis (1995), Gehrlein and Wagner (1997b), Greco et al. (1998), Zopounidis and Dimitras (1998), Zopounidis and Doumpos (1999), Zopounidis et al. (1999b), Konno and Kobayashi (2000)

Application Area	**Studies**
Credit cards assessment	Lam et al. (1996), Zopounidis et al. (1998)
Country risk evaluation	Doumpos and Zopounidis (2001a)
Ecology	Rossi et al. (1999), Flinkman et al. (2000)
Educational administration	Choo and Wedley (1985), Lam et al. (1993)
Energy planning	Diakoulaki et al. (1999)
Medicine	Stefanowski and Slowinski (1998), Belacel (2000), Michalowski et al. (2001)
Personnel management	Gochet et al. (1997)
Portfolio selection and management	Zopounidis et al. (1999a), Nakayama and Kagaku (1998), Doumpos et al. (2000)
R&D project evaluation	Jacquet-Lagreze (1995)
Technical diagnosis	Nowicki et al. (1992)
Venture capital investments	Stam (1990)

This broad range of application domain lead researchers to develop different approaches for constructing sorting/classification models. The approach developed in this study is an interactive method, which utilizes Even-Swap method, weight space reduction techniques and dominance relations. The use of Even-Swap method in the study is slightly different from it is original use, where the original method aims to find the best alternative in a given alternative set by interacting with the DM.

1.2 Literature Survey

Considering a set of alternatives described by a number of criteria; different problems are considered in the literature. One, which is more frequently addressed, is to identify the best alternative or select a limited set of the best alternatives (this problem is also referred as “the choice problem”). Another problem is to construct a rank ordering of the alternatives from best to the worst ones (this problem is also referred as “the ranking problem”). One other problem is to classify or sort the alternatives into groups, where the groups may either have a preference relation or not (this problem is also referred as “the classification/sorting problem”).

The problem of identifying the most preferred alternative among a number of alternatives where each alternative is defined by several criteria is well studied in the literature. The studies of Keeney and Raiffa (1976) and Green and Srinivasan (1978) attempt to solve this problem by fitting a utility function that explains the preferences of the decision maker (DM), and then finding the alternative that performs best according to the fitted utility function.

Another approach for finding most preferred alternative had been interactive approach. Interactive approaches typically assume that the DM has an underlying utility function. However, the exact form of the utility function is assumed to be unknown to both the DM and the analyst. The DM is expected to be consistent with his/her underlying utility function while expressing his/her preferences. For the case where the underlying utility function is assumed to be linear, Zionts (1981) and Köksalan (1984) developed interactive approaches. Several interactive approaches have been developed for the quasi-concave utility function case (Korhonen et al. 1984, Köksalan et al. 1984, Köksalan and Taner 1992, Malakooti 1989); Köksalan and Sagala (1995) also developed an approach for the general monotone utility function case.

Korhonen (1998) developed a visual interactive approach that makes no assumption on the underlying utility function of the DM. Köksalan and Öden (1989) and Köksalan and Rizi (2001) have also developed visual interactive approaches and utilized graphical aids in their interactive approaches. A more recent review of the multi-criteria literature is provided by Jacquet-Lagreze and Siskos (2001).

The problem of constructing a rank ordering of the alternatives from has also been

a problem of interest. Assuming that the attributes have been measured at least on an ordinal scale, Korhonen and Soimaa (1981) attempt to find a complete rank ordering of alternatives. Malakooti (1989) uses quasi-concave non-linear multi-attribute utility functions to rank multiple criteria alternatives, and shown that pair comparison questions can be used to generate partial information on the weights.
Another type of problem is the assignment of alternatives into predefined groups, which is referred to as classification or sorting problems. While both classification and sorting refer to the assignment of a set of alternatives into predefined groups, they differ with respect to the way that the groups are defined. Classification refers to the case where the groups are defined in a nominal way. On the contrary, sorting refers to the case where the groups are defined in an ordinal way starting from those including the most preferred alternatives to those including the least preferred alternatives.

Earlier work on classification can be traced back to Fisher (1936), whose work was on the linear discriminant analysis. Some other statistical approaches was developed following Fisher (Bliss 1934, Berkson 1944, McFadden 1974), which was later on criticized for their statistical assumptions (Altman et al. 1981).

Recent research on developing classification and sorting models is mainly based on operations research and artificial intelligence. Compared to other approaches, multi-criteria decision aiding research (MCDA) does not focus solely on developing automatic procedures for analyzing an existing data set in order to construct a classification/sorting model. MCDA researchers also emphasize on the development of efficient preference modeling methodologies that will enable the decision analyst to incorporate the decision maker's preferences in the developed classification/sorting model (Zopounidis, C., Doumpos, M., 2002).

Outranking relation and utility function are the most widely used criteria aggregation models in MCDA literature, which are also employed for classification and sorting purposes. The most widely used sorting method based on outranking relations is the ELECTRE TRI method (Yu 1992, Roy and

Bouyssou 1993). An alternative approach for outranking relation, the utility theory framework, is used in UTADIS for sorting purposes (Jacquet-Lagreze 1995, Zopounidis and Doumpos 1999).

Köksalan and Ulu (2001, 2003), developed an interactive procedure for partitioning the alternatives into preference classes regarding different forms of utility functions of the DM. Dominance, weight space reduction and direct DM placement techniques are used to place alternatives.

More recently significant research has been conducted on the use of the rough set approach as a methodology of preference modeling in multi-criteria decision problems (Greco et al. 1999, 2000). The rough approximations of decision classes involve dominance relation, instead of indiscernibility relation considered in the basic rough sets approach. They are built of reference alternatives given in the sorting example. Decision rules derived from these approximations constitute a preference model. Also, the dominance-based rough set approach is able to deal with sorting problems involving both criteria and regular attributes (whose domains are not preference ordered), (Greco et al., 2002), and missing values in the evaluation of reference alternatives (Greco et al., 1999, 2000b).

The use of neural networks is another interesting approach that can be used for preferential modeling purposes in multi-criteria classification and sorting problems. Neural networks enable the modeling of highly complex non-linear behaviors of decision-makers. Main disadvantage of the neural networks is that, the results of a neural networks are difficult to interpret in terms of the given inputs to the network. The major advantage on the other hand is that, neural networks can be used to assess utility functions, without posing any assumptions or restrictions on their particular structure or properties. Arhcer and Wang (1993) showed that neural networks can provide an efficient mechanism for preference modeling in sorting problems.

It is important to note that, the development of decision support systems that will enable decision-makers to take advantage of the capabilities that the classification and sorting approaches provide. Several multi-criteria decision support systems have been developed over the past decade implementing MCDA classification and sorting methods. The most characteristic are the RANGU system developed by Stam and Ungar (1995), the PREFDIS system of Zopounidis and Doumpos (2000a), the ELECTRE TRI-Assistant system of Mousseau et al. (2000), the ROSE system of Predki et al. (1998) and the 4eMka system of Greco et al. (1999a) (Zopounidis, C., Doumpos, M., 2002).

1.3 Even Swaps

Even Swaps (Hammond et al. 1998, 1999) is a multi-criteria decision making method based on value trade-offs which are called *even swaps*. Performing sensible trade-offs is one of the most important and difficult challenges in decision making (Keeney and Raiffa 1976; Keeney 2002). The even swaps method is developed in order to fill the gap of clear, easy-to-use and rational trade-off methodology. It provides a practical way of making trade-offs among any set of objectives across a range of alternatives. It is a form of bartering that forces the decision maker to think about the value of one objective in terms of another. The even swap method does not argue that it provides a mechanism which makes complex decisions easy, but what it does provide is a reliable mechanism for making trades and consistent framework in which to make them.

In an even swap, the value of an alternative in one attribute is changed and this change is compensated with a preferentially equal value change in some other attribute. The new alternative with these revised values is equally preferred to the initial one and thus it can be used instead. The aim of the method is to carry out even swaps that make either attributes irrelevant, in the sense that all the alternatives have equal values on this attribute, or alternatives dominated, in the sense that some other alternative is at least as good as this alternative on every attribute. Such attributes and alternatives can be eliminated, and the process continues until one alternative, i.e. the most preferred one, remains.

The main requirement for using the Even Swaps method is to understand the idea of an even swap. The decision maker (DM) does not need to have a mathematical background to use the method. Hammond et al. (1998, 1999) emphasize on the practical aspects of the process, and let the DM to focus on the most important work of decision making: deciding the real value to him/her.

The general flow of the even swap framework is provided in Figure 1:

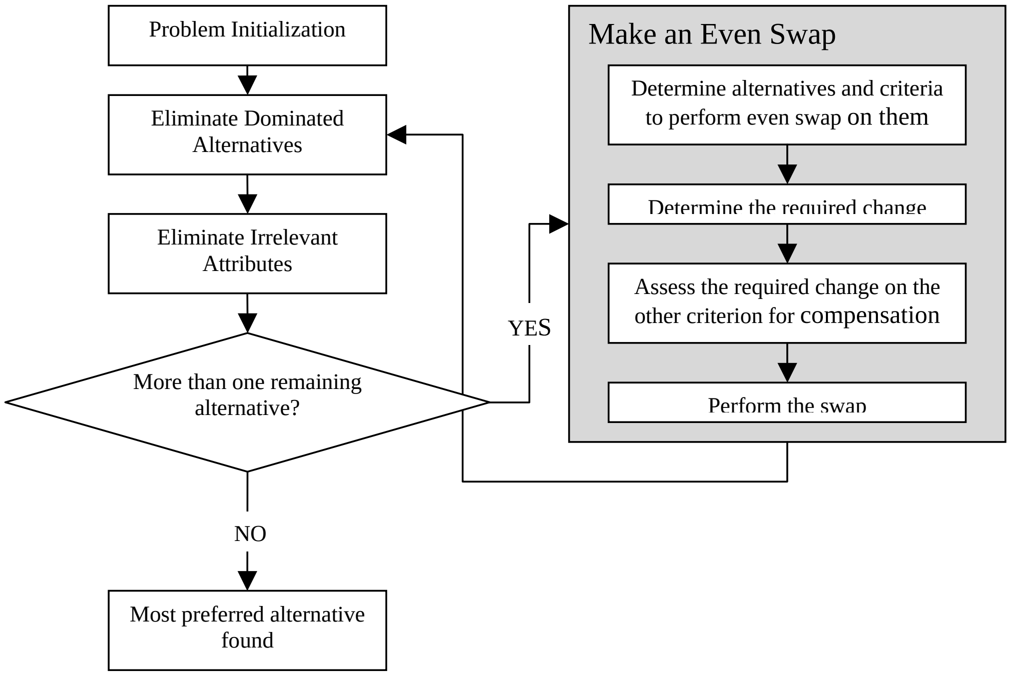

Problem Initialization
Eliminate Dominated Alternatives
Eliminate Irrelevant Attributes
More than one remaining alternative?
YES
NO
Most preferred alternative found
Make an Even Swap
Determine alternatives and criteria to perform even swap on them
Determine the required change
Assess the required change on the other criterion for compensation
Perform the swap

Figure 1. An overall flow of the Even-Swap framework

At the problem initialization step, a consequence table is constructed in order to have a clear picture of the alternatives and their consequences for each criterion. The important thing when constructing consequence tables is to use consistent terms for each criterion. Once the consequences table is constructed and criterion values for each alternative is mapped, look for opportunities to eliminate one or more alternatives. If an alternative A is better than alternative B in some criteria and not worse than B in all other criteria, alternative B can be eliminated from consideration (alternative B is dominated by alternative A). Another issue is to eliminate irrelevant criteria with an obvious tenet; if every alternative is rated equally on a given criterion, that criterion can be ignored while making decisions. Now, the challenge is to choose the most proper even-swap to perform among numerous choices. This selection shall be made considering the information that will be provided after the swap. Even-swaps that can lead to a possible "dominance" or "irrelevant criteria" shall be preferred.

Determining the relative value of different criterion values is hard. The originators of the even-swap approach, Hammond, Keeney and Raiffa, quoted some suggestions to make sound trade-offs (Hammond et al. 1998, 1999).

There are few reported applications of even-swaps in the literature; where one of them is on strategy selection in a rural enterprise (Kajanus et al. 2001) and another one on environmental planning (Gregory and Wellman 2001). Despite of the simplicity of the method, the lack of use may be due to insufficient computational help provided. Recently Mustajoki and Hämäläinen (2004) developed a decision support system for even swap approach, which is supported by Preference Programming. Preference programming is a framework for modeling incomplete information within multi-attribute value theory.

1.4 Even –Swap Example

To illustrate the above discussed methodology, a small example is given below. The problem is about selecting a second-hand car among a number of alternatives. Alternatives are evaluated on three criteria (of course, there should be more, but three of them are selected for simplicity), these are:

- Age of the car (given by model year)
- Mileage of the car (given in kilometers)

- Price of the car (in YTL.)

Model year of the car is a higher the better type criterion. But the last two, mileage and price are lower the better.

The alternatives are selected from a popular car sales web-site from Turkey. The alternatives and their values on three criteria are given in Table 2 (this is the consequence table for the DM):

Table 2. Even Swap Example - Consequence Table

	Toyota Corolla GL	Peugeot 206 XR	Opel Corsa Swing	Honda Civic HB	Ford Fiesta
Model	1999	2000	1999	2000	2000
Mileage	77000 km	55000 km	91000 km	130000 km	64000 km
Price	14500 YTL	16000 YTL	15750 YTL	14750 YTL	15000 YTL

Now, we will look for opportunities to eliminate one or more alternatives by dominance. It can be observed that, “Opel” is dominated by both “Ford” and “Toyota”, so it can be eliminated for further consideration, (see Table 3).

Table 3. Even-Swap Eample - Eliminated Alternative by Dominance

	Toyota Corolla GL	Peugeot 206 XR	~~Opel Corsa Swing~~	Honda Civic HB	Ford Fiesta
Model	1999	2000	~~1999~~	2000	2000
Mileage	77000 km	55000 km	~~91000 km~~	130000 km	64000 km
Price	14500 YTL	16000 YTL	~~15750 YTL~~	14750 YTL	15000 YTL

Now, we will perform an even-swap on “Toyota” and ask the DM “How much will you increase the price, for an increase in the model from 1999 to 2000?”. The DM says “I will increase the price from 14500 YTL to 15250 YTL”. The new consequence table is given below (Table 4), the swapped values are highlighted.

All alternatives score the same on “Model” criterion, so this alternative is now irrelevant and can be eliminated; furthermore, “Toyota” is now dominated by “Ford”, so it can be eliminated from the alternative set, these are also shown in Table 4.

Table 4. Even-Swap Example - A Criterion and Alternative Eliminated

	~~Toyota Corolla GL~~	Peugeot 206 XR	Honda Civic HB	Ford Fiesta
~~Model~~	~~2000~~	~~2000~~	~~2000~~	~~2000~~
Mileage	~~77000 km~~	55000 km	130000 km	64000 km
Price	~~15250 YTL~~	16000 YTL	14750 YTL	15000 YTL

The consequence table is reduced to a much smaller form than the original table. But, three alternatives are left, so we need more swaps and propose the DM another one on “Ford”: “How much will you increase the price, for a decrease in the mileage from 64000 to 55000?”. The DM says “I will increase the price from 15000 YTL to 15500 YTL”. Now, “Peugeot” is dominated by “Ford” and eliminated for further consideration (Table 5).

Table 5. Even-Swap Example - Eliminated Alternative

	~~Peugeot 206 XR~~	Honda Civic HB	Ford Fiesta
Mileage	~~55000 km~~	130000 km	55000 km
Price	~~16000 YTL~~	14750 YTL	15500 YTL

Two alternatives are left, one more swap is required, the following question is asked to DM, for performing an even-swap on “Honda”: “How much will you increase the price, for a decrease in the mileage from 130000 to 55000?”. The DM says “I will increase the price from 14750 YTL to 16500 YTL”. Finally “Honda” is dominated by “Ford”, and eliminated. This reveals “Ford” to be the preferred alternative for the DM (Table 6).

Table 6. Even-Swap Example - Final Table

	~~Honda Civic HB~~	Ford Fiesta
Mileage	~~55000 km~~	55000 km
Price	~~16500 YTL~~	15500 YTL

1.5 The Evolution of the algorithm

This study is based on different approaches developed in the field of multi-criteria decision making and sorting problems literature; and proposes a new interactive approach for multi-criteria sorting problems. The weight space reduction ideas generated in Köksalan and Ulu (2001, 2003) are utilized, but LPs used for weight space reduction are different in order to be more efficient. Even-Swaps methodology (Hammond et al. 1998, 1999) is included in the algorithm both for eliciting information from the DM and for placing the alternatives. Even-Swaps approach is originally proposed for selecting the best alternative among a set of alternatives, however, in this study it is used for sorting a set of alternatives to preference classes.

The following chapter discusses the approaches developed in each step of the proposed algorithm. The third chapter presents the algorithm step by step, and illustrates a manual example. The fourth chapter introduces the developed automation for the algorithm, and presents some results, which are obtained by using the algorithm. The final chapter gives a summary of the study, presents some conclusions and proposes some possible future work, which can be performed as extensions to this study.

CHAPTER 2

DEVELOPMENT OF THE MODEL

This chapter discusses the approach developed in each step of the algorithm. A detailed flow of the algorithm is given in Appendix A.

The proposed algorithm is two phased, the initialization phase and then the selecting alternatives phase. At the initialization phase an initial estimation of the DM's underlying utility function is made, using the information gained from an Even-Swap which is proposed to DM by selecting two alternatives from the alternative set. At the placing alternatives phase, initially an alternative is selected to be placed, among the set of unplaced alternatives. The algorithm tries to place the selected alternative into a preference class either using dominance relations, convex combinations or weight space reduction techniques. If the alternative cannot be placed, an Even-Swap is performed on the alternative for swapping to a dummy alternative, which can be placed by convex combinations. If the alternative has not been placed yet, the DM is asked to place the alternative among the range of possible preference classes. Following sections discusses all the approaches developed in the algorithm.

2.1 Some Notation and Assumptions

Following list defines some notation that will be used in this study,

- i^{th} alternative that is to be placed in preference classes is represented as X_i
- $X_i = (x_{i,1}, x_{i,2}, ..., x_{i,j}, ..., x_{i,p})$ where $x_{i,j}$ is the score of the i^{th} alternative

in the j^{th} criterion

- C_i is the i^{th} class that the alternatives can be placed in; C_1 being the "best" class and C_t being the "worst", where there exist k classes.

Other notation will be introduced when they are defined in the flow of the algorithm.

The model assumes that the DM has a linear utility function. That is, the utility of alternative X_i is given by:

$$U(X_i) = \sum_j \lambda_j U_j(x_{i,j})$$

where $U_j(x_{i,j})$ is the criterion score of X_i on criterion j, and λ_j is the weight of criterion j.

It is assumed that the DM is consistent with his/her responses and can place alternatives consistently whenever s/he I asked. One other assumption is that, at least one of the criteria is ordinal and continuous. This assumption is required to enable performing the swap on that criterion. The model assumes that λ_js – criteria weights- are not known, and tries to generate an estimated region for λ_js using DM's responses.

A consistency index, α, is used when evaluating the Even-Swap information. This is for evaluating the DM's swap response within a precision bound, the discussions on this α value will be given in the following sections.

2.2 Even Swaps

At the initialization phase of the approach, an Even-Swap is performed. The intent of the Even-Swap is to have an idea about the DM's underlying utility function. This early information on the utility function provides the infrastructure for the latter steps. Following sections discuss: the selection of the alternatives that will be taken into consideration when performing the Even-Swap, performing the Even-Swap, and the case when there exists more than two criteria for the alternatives.

2.2.1 Selecting the alternatives for the even swap

Two alternatives will be selected to perform the Even-Swap. Selecting the alternatives is crucial since we want to make the best use of the DM's response while performing the swap. The original Even-Swap methodology is utilized for generating possible dominance relationships among alternatives which do not dominate each other. So, the selected alternatives shall not be dominating each other. Another concern in alternative selection is the ease of the swap. If the criterion values of the alternatives are too far, it will be hard for the DM to make such a big swap, and as the size of the swap increases, the size of the error may increase.

The two alternatives are selected among the alternatives in such a way that

i. they do not dominate each other

ii. they have the smallest Euclidean distance

The selected alternatives are presented to the DM.

2.2.2 Performing the Even-Swap

The Even-Swap will be performed with the motivation discussed in section 1.3. Following steps will be followed while performing the swap:

- Take one of the selected alternatives as the base (call it the base alternative); the swap will be performed on the other alternative (swapped alternative).
- Select one of the criteria (call it fixed criterion) and equate the value of the swapped alternative on that criterion to that of the base alternative.
- Ask the DM, how much he/she wants to swap on the other criterion, which is not fixed, to compensate the change on the fixed criterion.

Even swap is illustrated on a two criteria example:

Assume that we have two alternatives A and B having the following scores on two criteria (Table 7):

Table 7. Even-Swap example - alternatives

	Alternative A	Alternative B
Criterion 1	45	60
Criterion 2	78	58

Let the DM make a swap on alternative A, and alternative B is selected as the base alternative. Then, we equate the first criterion value of alternative A to that of alternative B, and ask the DM how much he/she is willing to decrease on criterion 2 value, to compensate the increase in criterion 1 (Table 8):

Table 8. Even-Swap example - the swap

	Alternative A	Alternative $A^{swapped}$
Criterion 1	45 $\longrightarrow$	60
Criterion 2	78 $\longrightarrow$	???

Say the DM is willing to decrease the value of the second criterion from 78 to 65, to compensate the increase in the first criterion from 45 to 60. So the alternative $A^{swapped}$ becomes (60, 65). Initially no dominance relations are apparent between alternatives A and B. However, now alternative $A^{swapped}$ dominates alternative B. So, $A \succ B$. This example is show graphically in Figure 2.

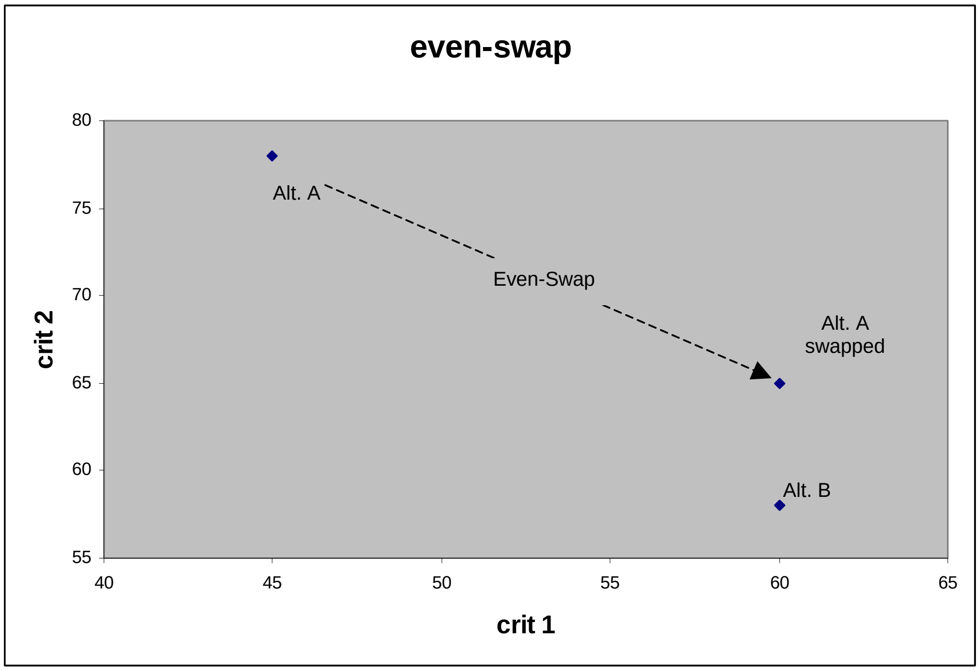


Figure 2. Even-Swap example - graphical representation of the swap

2.2.3 Even-Swap on more than two criteria

As discussed in section 1.3. the Even-Swap method is originally proposed to be performed on two criteria. However, we use the approach for comparing two alternatives, and alternatives can have more than two criteria. For that reason, the method is expanded to enable the comparison possible for alternatives having more than two criteria. This is done by performing consecutive swaps. The idea is illustrated on a four criteria example below:

Illustration:

Let X_i and X_j be the selected alternatives, having the criterion values respectively: $X_i = (x_{i,1}, x_{i,2}, x_{i,3}, x_{i,4})$ and $X_j = (x_{j,1}, x_{j,2}, x_{j,3}, x_{j,4})$. Let X_i be the fixed alternative and the Even-Swap be performed on X_j. The following steps will be used:

- Initially choose the first criterion to be the fixed criterion

- Equate the first criterion value of X_j to that of X_i
- Swap the second criterion value of X_j to compensate the change in the first criterion, say the swapped value is $x^s_{j,2}$
- Equate the second criterion value of X_j, that is $x^s_{j,2}$, to that of X_i
- Swap the third criterion value of X_j to compensate the change in the second criterion, say the swapped value is $x^s_{j,3}$
- Continue in the same manner

The approach discussed above is pictured in Figure 3, where dashed squares represent the Even-Swaps, arrows representing the swaps. X_j^{1c}, X_j^{2c}, $X_j^{'}$ are all equivalent alternatives to X_j, which are generated after the Even-Swaps.

In X_j^{1c}, the DM chooses the value $x^s{}_{j,2}$ such that, in the first two criteria, the DM is indifferent between ($x_{j,1}, x_{j,2}$) and ($x_{i,1}$, $x^s{}_{j,2}$). Then the DM chooses $x^s{}_{j,3}$ such that s/he is indifferent between ($x_{j,1}, x_{j,2}$, $x_{j,3}$) and ($x_{i,1}, x_{i,2}, x^s{}_{j,3}$). Finally, the DM choose $x^s{}_{j,4}$ such that $X_j = (x_{j,1}, x_{j,2}, x_{j,3}, x_{j,4})$ and $X_j^{'} = (x_{i,1}, x_{i,2}, x_{i,3}, x^s{}_{j,4})$. It would be beneficial to leave the easier swap to the last step, since all the changes on the other criteria will be compensated by this swap.

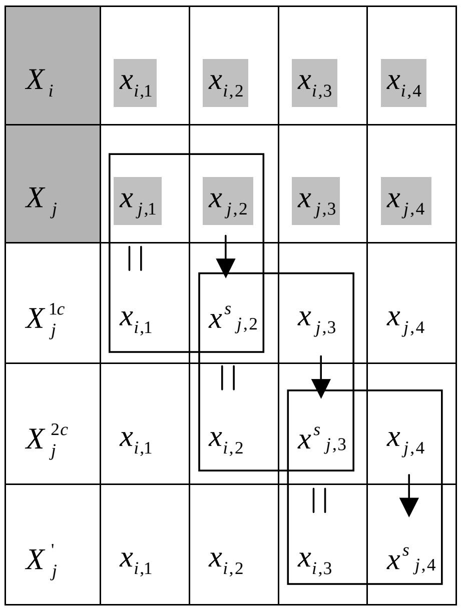

X_i $x_{i,1}$ $x_{i,2}$ $x_{i,3}$ $x_{i,4}$
X_j $x_{j,1}$ $x_{j,2}$ $x_{j,3}$ $x_{j,4}$
X_j^{1c} $x_{i,1}$ $x^s{}_{j,2}$ $x_{j,3}$ $x_{j,4}$
X_j^{2c} $x_{i,1}$ $x_{i,2}$ $x^s{}_{j,3}$ $x_{j,4}$
$X_j^{'}$ $x_{i,1}$ $x_{i,2}$ $x_{i,3}$ $x^s{}_{j,4}$

Figure 3. Even-Swap on more than two criteria

2.2.4 Estimating the utility function using the Even-Swap

An LP is constructed, in order to make an estimation of the DM's underlying utility function, using the information obtained from the swaps made in the previous step. In fact the Even-Swap implies a direct ratio relationship between the weights of the criteria on which it is performed. However, the developed LP evaluates this direct relationship within a consistency interval, depending on the precision of the DM; the size of the interval can be changed.

There are two constraints coming from each performed Even-Swap and one from the implied preference relationship after the swaps. These constraints are elaborated below:

Constraints obtained from the swap:

Assume that the DM performs the swap on alternative i and the Even-Swap will be performed on criterion 1 and criterion 2, where both criteria are higher the better type. The values of the alternative on these pair of criteria which the swap is performed are $x_{i,1}$ and $x_{i,2}$. Again assume that the DM makes a swap from $x_{i,2}$ to

${x^s}_{i,2}$, to compensate the change for going from $x_{i,1}$ to ${x^s}_{i,1}$. Where, if $x_{i,1}$ is greater than ${x^s}_{i,1}$, $x_{i,2}$ shall be smaller than ${x^s}_{i,2}$; and if $x_{i,1}$ is smaller than ${x^s}_{i,1}$, $x_{i,2}$ shall be greater than ${x^s}_{i,2}$, that is the swap shall be performed in the reverse direction. By this swap, the DM implies the following ratio between the weights of criterion 1 and 2 (for simplicity $x_{i,j}$ is used instead of $u_j(x_{i,j})$):

$$\frac{\lambda_2}{\lambda_1} = \frac{{x^s}_{i,1} - x_{i,1}}{x_{i,2} - {x^s}_{i,2}}$$

However, as mentioned above, the response of the DM is evaluated within a consistency interval. Let α represent the consistency index for the DM's implied ratio on the criterion weights. Then, the relationship becomes:

$$(1-\alpha)\frac{{x^s}_{i,1} - x_{i,1}}{x_{i,2} - {x^s}_{i,2}} \le \frac{\lambda_2}{\lambda_1} \le (1+\alpha)\frac{{x^s}_{i,1} - x_{i,1}}{x_{i,2} - {x^s}_{i,2}}$$

This relationship gives two constraints for the weight space:

$$\lambda_1(1+\alpha)\frac{{x^s}_{i,1} - x_{i,1}}{x_{i,2} - {x^s}_{i,2}} - \lambda_2 \ge 0$$

and

$$\lambda_2 - \lambda_1(1-\alpha)\frac{{x^s}_{i,1} - x_{i,1}}{x_{i,2} - {x^s}_{i,2}} \ge 0$$

Constraints obtained from the preference relation on the initial alternatives:

After the swap a dominance relation appears between the swapped alternative and the base alternative, since all but one of the criteria values are equalized. That unequal criterion value determines the direction of the dominance.

Let's assume that, the criterion value of the swapped alternative is higher than that of the base alternative. Then the swapped alternative dominates the base alternative. Since the swapped alternative is assumed to have the same utility value with its original state and all the intermediate alternatives generated during the Even-Swap (when there exist more than two criteria), all these alternatives are preferred to the base alternative. Then, corresponding constraints are added to constrain the weight space. If the swapped alternative is dominated by the base alternative, all preference relations are reversed.

The preference relation is represented with "$\succ$" or "$\prec$". For all implied preference relation the following constraint is added:

$$\lambda\left(X_q - X_r\right) \geq \varepsilon \quad where \quad X_q \succ X_r$$

With the constraints generated from the performed Even-Swap, the following initial model is developed to define the weight space for a two criteria example:

$$\begin{aligned}
&\max \varepsilon \\
&s.t. \\
&\lambda_1(1+\alpha)\frac{x^s{}_{i,1} - x_{\ i,1}}{x_{\ i,2} - x^s{}_{i,2}} - \lambda_2 \geq \varepsilon \\
&\lambda_2 - \lambda_1(1-\alpha)\frac{x^s{}_{i,1} - x_{\ i,1}}{x_{\ i,2} - x^s{}_{i,2}} \geq \varepsilon \\
&\lambda\left(X_q - X_r\right) \geq \varepsilon \qquad\qquad \forall X_q \succ X_r \\
&\sum \lambda_j = 1 \\
&\lambda \geq 0 \\
&\varepsilon \geq 0
\end{aligned}$$

All the constraint are written to be greater than a value of ε, which is to be maximized, to force the LP to find a weight set which is most distant to the nearest bound.

Solving the LP, an estimated weight set of the criteria is obtained, which will be used in the following stages of the approach. Each time when a new preference relation is implied by DM placement, a new constraint is added to the model, and the estimated criterion weights are recalculated.

2.3 Alternative Selection

The model utilizes some set of rules to select the alternative that will be treated in the algorithm for placing it in a preference class. Alternative selection is very crucial for the performance of the algorithm, since a good selection of alternatives will reduce the DM's effort. To provide such a good selection, the model uses the estimated utility function, obtained from the initially performed Even-Swap, and some set of rules. The rules for alternative selection are given below:

1. If the bounds for all preference classes are not defined

   The bounds for the preference classes are determined using the estimated utility function. For a preference class, among the previously placed alternatives to that class, the one having the highest estimated utility is the upper bound for that preference class, and the one having the lowest estimated utility is the lower bound. For the best class, if only one alternative is placed previously, than that defines the lower bound, similarly if only one alternative is placed in the worst class than that defines the upper bound for that class. If one of the following conditions is satisfied, there is no way of determining all the bounds for the problem.

      i. No alternatives placed in the best class:
      ii. No alternatives placed in the worst class
      iii. Less than two alternatives placed in an intermediate class
         1. No alternatives placed in that intermediate class
         2. One alternative is placed in that intermediate class

   Initially number of alternatives in each class is calculated, considering the number of all alternatives and the number of preference classes, and assuming there is approximately equal number of alternatives in each class. Then the alternatives are sorted according to the estimated utility function. This sorting will give an estimated grouping of the alternatives. Finally, following decisions are made for the alternative selection, for the above stated cases:

      i. Select the alternative which has the minimum estimated utility value in the “estimated best class” (note that “estimated best class” term is used, since no alternatives are actually placed in this class yet, the selection is based on the sorting made using estimated utility function)

ii. Select the alternative which has the maximum estimated utility value in the “estimated worst class”

iii. Again the selection will be based on the estimated sorting of the alternatives

a) Select the alternative having the maximum estimated utility value, among the estimated set of alternatives for that intermediate class, this alternative is expected to form the upper bound for that class.

b) Select the alternative having the minimum estimated utility value, among the estimated set of alternatives for that intermediate class, this alternative is expected to form the lower bound for that class. (note that if there is only one alternative actually placed in that class, that alternative will be forming the upper bound for that class since the algorithm tries to place the alternatives having higher estimated utility values first)

2. If the bounds for all preference classes are defined

The unplaced alternatives, which are out of bounds defined by the estimated utility values, have precedence when selecting the alternative to place. So, once the bounds for all alternatives are defined, the algorithm searches for the existence of alternatives which are out of the estimated bounds.

i. If there exists some alternatives that fall out of the defined estimated boundaries, among those alternatives, select the one which is most distant from the closest boundary in terms of estimated utility. The reason for selecting the most distant alternative is to maximize the benefit of the information that will be obtained from the placement of that alternative. The bound defined by estimated utilities is enlarged to the maximum extent possible by either placing that alternative to the better class or the worse class.

ii. If there are no alternatives falling outside the estimated boundaries, select the alternative which is closest to the closest boundary.

2.4 Determining Best and Worst Classes

Along the execution of the algorithm, the best and worst possible classes that the selected alternative can belong to, are utilized. The intent is to narrow down the range of possible classes that the alternative may be placed, and whenever best possible class is the same as the worst possible class, the algorithm places the alternative to that class.

Initially, best class index for all alternatives is set to 1 and worst class index for all alternatives is set to *t* (number of classes). Then, as alternatives are placed to preference classes, going over the previously placed alternatives, alternatives that are dominated by the selected alternative are searched. Among the dominated alternatives set, ones that have the smallest class index (lower the index, better the class), determines the worst class that selected alternative may belong to, and it will be denoted by X_k^W. Similar approach is followed for determining the best class that the selected alternative may belong to. Going over the previously placed alternatives, the alternatives that dominate the selected alternative are searched. Among the dominating alternatives set, ones that have the largest class index (higher the index, worse the class), determines the best class that the selected alternative may belong to, and it is denoted by X_k^B.

The alternative set is traced to identify dominance relationships. For each alternative, set of dominating alternatives and set of dominated alternatives are constructed. These sets are useful when the alternative is placed in the 1st class (best class) or *t*th class (worst class). If the alternative is placed in the 1st class, then all the alternatives dominating that alternative can be safely placed to the 1st class. Similarly if the alternative is placed in the *t*th class, then all the alternatives dominated by that alternative can be safely placed to the *t*th class.

2.5 Convex Combination Check

If the selected alternative can be expressed as a convex combination of alternatives belonging to the same class, then the selected alternative also belongs to that class. The following LPs are used to decide whether the selected alternative is a convex combination, where X_k is the selected alternative and C_t is the class under consideration. $\mathbf{e}_1$, $\mathbf{e}_2$ are vectors such that $\mathbf{e}_1 = [\varepsilon_1,...,\varepsilon_1,...\varepsilon_1,]$ and $\mathbf{e}_2 = [\varepsilon_2,...,\varepsilon_2,...\varepsilon_2,]$, where ε_1 and ε_2 are scalars.

LP1

$$\max\ \varepsilon_1$$
s.t.
$$\sum_{\mathrm{X_i} \in C_t} \mu_i X_i - X_k - \mathbf{e}_1 \geq 0$$
$$\sum \mu_\mathrm{i} = 1$$
$$\mu_i \geq 0$$

LP2

$$\max\ \varepsilon_2$$
s.t.
$$X_k - \sum_{\mathrm{X_i} \in C_t} \mu_i X_i - \mathbf{e}_2 \geq 0$$
$$\sum \mu_\mathrm{i} = 1$$
$$\mu_i \geq 0$$

Starting from the best class that the selected alternative may belong to, the above LPs are solved. By looking at the values of ε_1 and ε_2 a decision is made about the class that the selected alternative shall belong to. If either ε_1 or ε_2 is zero, that means the selected alternative can be expressed as a convex combination, so it belongs to C_t. If both ε_1 and ε_2 are greater than zero, that means there exist two convex combinations where the first one dominates the selected alternative, and the other is dominated by the selected alternative; so the alternative belongs to C_t. If ε_1 is greater than zero but ε_2 is smaller than zero, that means there exist some convex combinations that dominates the selected alternative, but no convex combination is dominated by the selected alternative; so this is the best class that the alternative may belong to. If ε_2 is greater than zero but ε_1 is smaller than zero, that means there exist some convex combinations that are dominated by the selected alternative, but no convex combination dominates the selected alternative; so this is the worst class that the alternative may belong to. If both ε_1 and ε_2 are smaller than zero, that means there are no convex combinations either dominating or dominated by the selected alternative. This result gives no information on the possible class of the selected alternative. All this discussion is summarized in the below table.

Table 9. Decisions for different values of decision variables

		ε_1		
		< 0	$=0$	> 0
ε_2	< 0	No info	$X_k \in C_t$	$X_k^B = C_t$
	$=0$	$X_k \in C_t$	$X_k \in C_t$	$X_k \in C_t$
	> 0	$X_k^W = C_t$	$X_k \in C_t$	$X_k \in C_t$

Going over all classes that the selected alternative may belong to, the alternative is either placed or its best or worst possible class index is updated.

Figure 4 provides a two dimensional graphical example to the above discussion. Suppose that, alternatives 1, 2 and 3 belong to the same class (say, class *f*), and other alternatives are evaluated with above LPs:

- As it can be observed graphically alternative 4 can be represented as a convex combination of alternatives 1, 2 and 3. So, for alternative 4, both $\varepsilon_1 > 0$ and $\varepsilon_2 > 0$ and alternative 4 also belongs to class *f*.
- For alternative 5, $\varepsilon_1 > 0$ and $\varepsilon_2 < 0$, that means class f is the best possible class alternative 5 can belong to.
- For alternative 6, $\varepsilon_1 < 0$ and $\varepsilon_2 > 0$, that means class f is the worst possible class alternative 6 can belong to.
- For alternatives 7 and 8, both $\varepsilon_1 < 0$ and $\varepsilon_2 < 0$, and no information provided.
- For alternatives on the lines bounding the shaded region, either $\varepsilon_1 = 0$ and $\varepsilon_2 = 0$, that can be expressed as a convex combination. Alternative 9 is such an alternative, and it also belongs to class *f*.

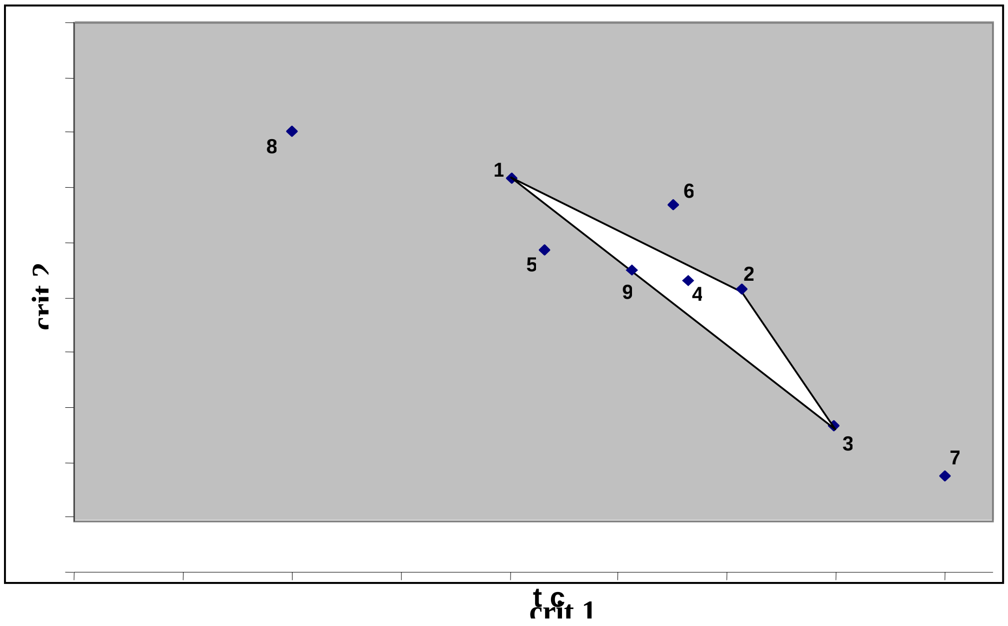


Figure 4. Graphical representation of the example

2.6 Utilizing LPs for best and worst classes

As mentioned above, the algorithm tries the narrow the range of possible classes that the selected alternative can be placed in. Two similar LPs are used to check the possibility of the placement of the selected alternative in preference classes with respect to the defined weight space. The developed LPs are similar to those developed by Köksalan and Ulu (2003), where each alternative in a certain class is considered separately. Here all alternatives in a class is considered in one LP. The reasoning behind this is that, to decide whether the class is the worst/best class that the selected alternative can belong to there should not exist any feasible weight set that makes the selected alternative worse/better than “all” alternatives in that certain class.

Starting with the initial utility estimation, weight space is defined. New constraints are added (weight space will be reduced) in either of the following cases:

- whenever a preference relation is implied by an Even-Swap
- whenever an alternative is placed to a class, and there are alternatives in worse classes which that alternative is not dominating. This implies a preference relation. If there exist some alternatives in better classes which are not dominating that alternative, this will also imply a preference relation.

The following LP is used for determining the best class that the selected alternative may belong to with respect to the defined weight space where *“S”* represents the defined weight space.

$$\begin{aligned}
&\max\ \varepsilon \\
&\text{s.t.} \\
&\lambda\,(X_k - X_h) \geq \varepsilon \qquad \forall X_h \in C_t \\
&\lambda \in S
\end{aligned}$$

The above LP is solved for each class C_t starting from the worst possible class until the worst possible class the selected alternative may belong to. If $\varepsilon < 0$ or the LP is infeasible, it is concluded that there is no weight set in the defined weight space which makes the selected alternative, X_k, better than all alternatives in that class under consideration. So, the class is marked to be the best class that X_k may belong to, X_k^B. If the LP is feasible, consider the next class.

A Similar approach is followed to determine the worst class. The following LP is constructed for that purpose:

$$\begin{aligned}
&\max\ \varepsilon \\
&\text{s.t.} \\
&\lambda\ (X_h - X_k) \geq 0 \quad \forall X_h \in C_i \\
&\lambda \in S
\end{aligned}$$

The above LP is solved for each class C_t starting from the best possible class until the worst possible class the selected alternative may belong to. If $\varepsilon < 0$ or the LP is infeasible, it is concluded that there is no weight set in the defined weight space which makes the selected alternative, X_k, worse than all alternatives in that class under consideration. So, the class is marked to be the worst class that X_k may belong to, X_k^W. If the LP is feasible, consider the next best class.

Whenever, $X_k^B = X_k^W$ the selected alternative can safely be placed in that preference class.

2.7 Finding equivalent dummy points

As mentioned in Section 2.5 if the selected alternative can be expressed as a convex combination of alternatives belonging to the same class, that selected alternative also belongs to that class. The following LP is constructed in order to check for a dummy point that dominates a convex combination of alternatives of a class, and is dominated by another convex combination of the same class and equivalent to the selected alternative in terms of estimated utility value. Since the dummy point is both dominating and dominated by two different convex combinations, it will be a member of that class.

$$
\begin{aligned}
&\max \varepsilon \\
&s.t. \\
&\lambda_{est}(X_{dum} - X_k) = 0 \\
&\sum_{X_i \in C_t} \mu_i X_i - X_{dum} \geq \varepsilon \\
&X_{dum} - \sum_{X_i \in C_t} \beta_i X_i \geq \varepsilon \\
&\sum \mu_i = 1 \\
&\sum \beta_i = 1 \\
&\mu_i \geq 0; \beta_i \geq 0; X_{dum} \geq 0; \varepsilon \geq 0
\end{aligned}
$$

Figure 5 shows an example of the case where this step can be used. Here, the “dummy” alternative can be expressed as a convex combination of alternatives 1, 2 and 3, which belong to same class. And using the estimated utility function, it can be said that alternative 5 and the “dummy” alternative has the same utility value. The DM is asked to perform an even-swap on alternative 5, to come around the “dummy alternative”. If the swapped point can also be expressed as a convex combination – it is expected because of the estimated utility function – we can place alternative 5 in the same class.

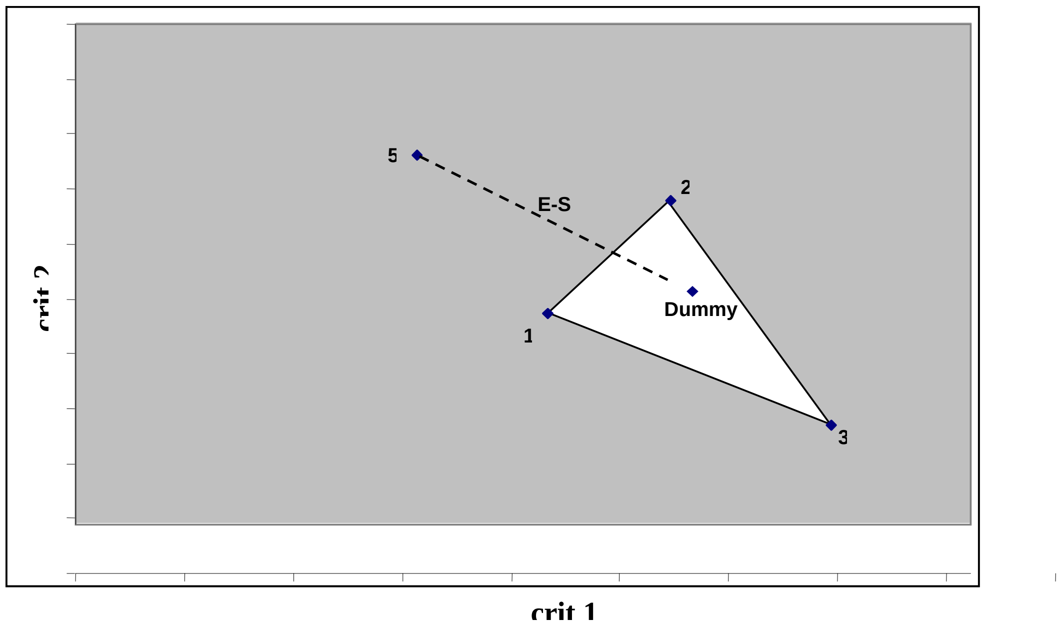


Figure 5. Step 6 shown graphically

Starting from the best class that the selected alternative may belong to, the above LP is solved. If a feasible dummy point can be found, an Even-Swap will be performed on the selected alternative, to generate a swapped alternative around the dummy point. Since the dummy alternative will be a member of that class, the swapped alternative is a close candidate to be a member. But before placing dominance relations with the convex combinations will be checked, since the dummy point is found using the estimated weights, so the swapped point can be slightly different.

2.8 Decision maker placement

If the selected alternative cannot be placed to a preference class with one of the above procedures, then the DM will be asked to place the alternative to a preference class between X_k^W and X_k^B.

Let the DM place the alternative to a preference class. If there exists some alternative X_d belonging to a worse class but not dominated by X_k, a constraint will be added to the weight space indicating $X_k \succ X_d$, that is $\lambda(X_k - X_d) \geq 0$. If there exists some alternative X_b belonging to a better class but not dominating X_k, a constraint will be added to the weight space indicating $X_b \succ X_k$, that is $\lambda(X_b - X_k) \geq 0$. (Koksalan, M.M., Ulu, C., 2003).

Figure 6, is a graphical example of how dominance and weight space reduction is utilized for alternative placements (Koksalan, M.M., Ulu, C., 2003). Suppose, DM places X_k to the worst class, the alternatives in the dashed rectangle are dominated by X_k, and they also belong to the worst class. Assume that, it is known by weight space reduction constraints that slope of the DM's underlying utility function lies between l_1 and l_2. Then, the alternatives which are marked with a "*", will also belong to same class by weight space reduction.

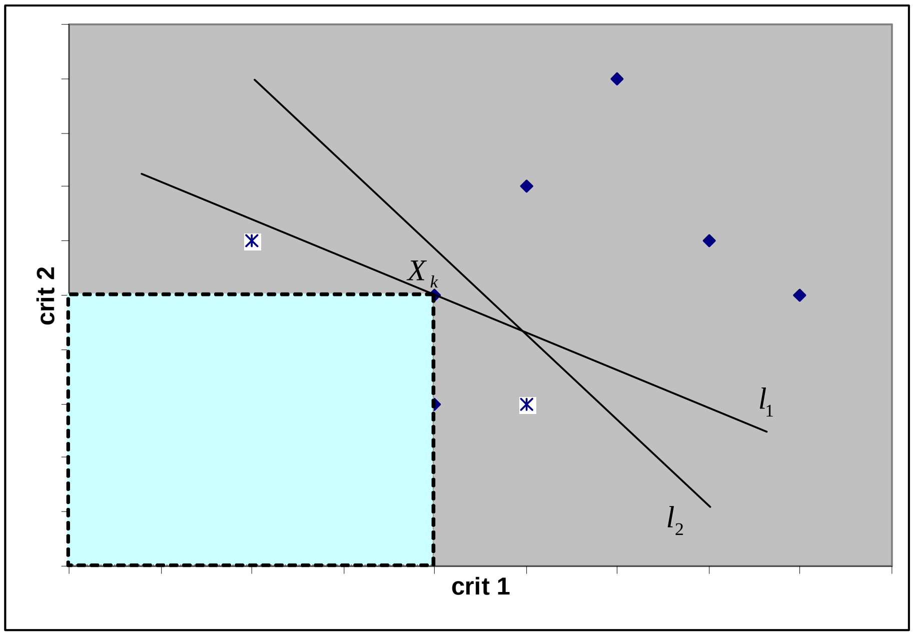


Figure 6. Graphical representation of dominance and weight space reduction

CHAPTER 3

THE ALGORITHM

3.1 Summary of the Algorithm

The algorithm used for sorting can be divided into two phases:

- initialization
- placing alternatives

At the “initialization phase” two alternatives are selected from the alternative set and an Even-Swap is performed on them. Utilizing the information coming from the Even-Swap, constraints are generated and an estimation of the DM’s utility function is made. An overall flow of the initialization phase is given in Figure 7 (for the detailed flow, look at Appendix A):

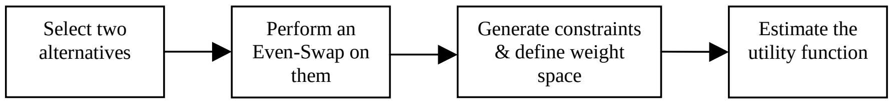

Select two alternatives
Perform an Even-Swap on them
Generate constraints & define weight space
Estimate the utility function

Figure 7. Flow of the initialization phase

At the “placing alternatives phase”, first the alternative to be placed is selected among the unplaced alternatives. Then according to dominance relations, best and worst classes for the alternatives are determined. The next step is to check the selected alternative for convex combinations of the preference classes. Then utilizing generated LPs and the reduced weight space, possible range of classes that the selected alternative may belong to is narrowed down. Then an equivalent dummy alternative, which belongs to a preference class, in terms of estimated utility is searched and an even swap on the selected alternative is performed to come around the dummy alternative. Whenever it is found that the possible best and worst classes are the same for the selected alternative, during the execution of the steps of the algorithm, the alternative is placed to that class. If this does not happen, DM is asked to place the alternative to a class among the range of classes that the alternative may belong to. An overall flow of the "placing alternatives phase" is given in Figure 8 (for the detailed flow, look at Appendix A):

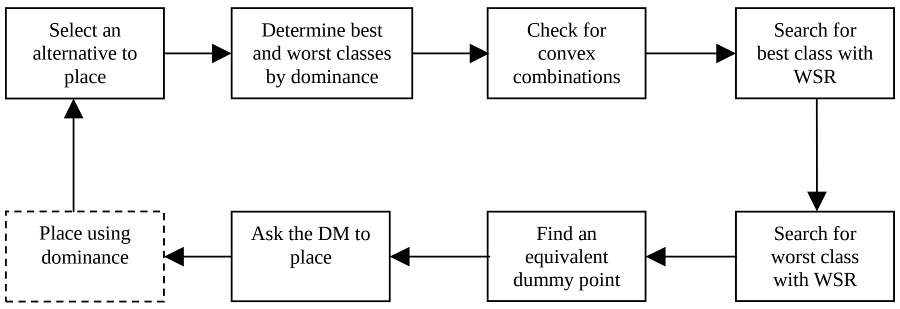

Select an alternative to place
Determine best and worst classes by dominance
Check for convex combinations
Search for best class with WSR
Search for worst class with WSR
Find an equivalent dummy point
Ask the DM to place
Place using dominance

Figure 8. Flow of the "placing alternatives phase"

3.2 The Algorithm

INITIALIZATION

Step 0.

Define X to be the set of all alternatives, and let there be n alternatives, define each as X_i. Then $X = \{X_1, X_2, ..., X_n\}$.

Define the j^{th} criterion value of the alternative X_i as $x_{i,j}$, and let there be p criteria, then $X_i = (x_{i,1}, x_{i,2}, ..., x_{i,j}, ..., x_{i,p})$.

Let X_k^B be the index of the best class that X_k can belong to; X_k^W be the index of the worst class that X_k can belong to; and X_k^C be the index of the class that X_k belongs to, when it is placed.

Let C_i be the set of alternatives that belong to i^{th} class; and let there be t preference classes where C_1 represents the "best" class and C_t represents the "worst".

Go to next step.

Step1.

Select two alternatives from the alternative set X. Among the alternative pairs which do not dominate each other, the closest one, in terms of Euclidean distance is selected. Let the selected alternatives be, X_q and X_r.

Go to next step.

Step 2.

Ask the DM to perform Even-Swap on the selected alternatives. Let the alternative to be swapped be X_q. If criteria *k* and *l* are considered, equate the k^{th} criterion value of X_q to that of X_r, and ask the DM how much s/he wants to swap on the l^{th} criterion. Let the DM swap from $x_{r,l}$ to $x^s{}_{r,l}$.

$x_{q,k} --> x_{r,k}(x^s{}_{q,k})$

$x_{r,l} --> ?(x^s{}_{r,l})$

After this swap, X_q becomes $X_q^c = (x_{q,1},...,x^s{}_{q,k}, x^s{}_{q,l},....,x_{q,p})$. Perform all the swaps on all criteria. Finally X_q becomes $X^s{}_q = (x^s{}_{q,1},...,x^s{}_{q,k}, x^s{}_{q,l},....,x^s{}_{q,p})$ where all criterion values except the last are equal to that of X_r's.

Go to next step.

Step 3.

One of $X^s{}_q$ or X_r dominates the other depending on the value of their last criterion. Let $X^s{}_q$ dominate X_r, write a preference constraint between X_q and X_r to constrain the weight space:

$$X_q \succ X_r$$

Write couple of constraints for each swap performed, using previously defined consistency index α:

$$\lambda_l(1+\alpha)\frac{x^s{}_{q,l}-x_{q,l}}{x_{q,k}-x^s{}_{q,k}}-\lambda_k \geq 0$$

and

$$\lambda_k-\lambda_l(1-\alpha)\frac{x^s{}_{q,l}-x_{q,l}}{x_{q,k}-x^s{}_{q,k}} \geq 0$$

Step 4.

Solve the following LP, and find an estimate of the DM's utility function.

$$\max \varepsilon$$

$$s.t.$$

$$\left.\begin{array}{l}\lambda_l(1+\alpha)\dfrac{x^s{}_{i,l}-x_{i,l}}{x_{i,k}-x^s{}_{i,k}}-\lambda_k \geq \varepsilon \\ \lambda_k-\lambda_l(1-\alpha)\dfrac{x^s{}_{i,l}-x_{i,l}}{x_{i,k}-x^s{}_{i,k}} \geq \varepsilon\end{array}\right\} \text{For each swap performed}$$

$$\lambda(X_q-X_r)\geq\varepsilon \qquad \forall X_q \succ X_r$$

$$\sum\lambda_j=1$$

$$\lambda\geq 0$$

$$\varepsilon\geq 0$$

Go to next phase.

PLACING ALTERNATIVES

Step 5. (initializing all alternatives)

Initially equate all best possible class indexes for all alternatives to best class, that is $X_k^B=1$; and equate all worst possible class indexes for all alternatives to worst class, that is $X_k^W=t$. Equate all class indexes for all alternatives to zero, $X_k^C=0$, that means they do not belong to any class yet.

Go to next step.

Step 6. (select one alternative)

Select an alternative among those currently unplaced. Let the selected alternative be X_k.

Go to next step.

Step 7. (decide best and worst classes)

Look for the set of alternatives that dominate X_k, let $\mathrm{D}\{X_k\}$ denote the set of alternatives dominating X_k. Looking at the class indexes (X^C) of all alternatives in the dominating set, assign the highest class index as the best possible class index of X_k; $X_k^B = \text{maximum class index in } \mathrm{D}\{X_k\}$.

Look for the set of alternatives that are dominated by X_k, let $\mathrm{D}'\{X_k\}$ denote the set of alternatives that are dominated by X_k. Looking at the class indexes (X^C) of all alternatives in the dominating set, assign the lowest class index as the worst possible class index of X_k; $X_k^W = \text{minimum class index in } \mathrm{D}'\{X_k\}$.

If $X_k^B = X_k^W = y$, then place X_k to class *y*. Go to Step13

Go to next step.

Step 8. (convex combination check)

Starting from class X_k^B to X_k^W solve the following LP couple to check whether X_k can be expressed as a convex combination of alternatives that belong to same class.

LP1 (where $\mathbf{e}_1 = [\varepsilon_1,...,\varepsilon_1,...\varepsilon_1,])$

$$
\begin{aligned}
&\max \ \varepsilon_1 \\
&\text{s.t.} \\
&\sum_{X_i \in C_t} \mu_i X_i - X_k - \mathbf{e}_1 \geq 0 \\
&\sum \mu_i = 1 \\
&\mu_i \geq 0
\end{aligned}
$$

LP2 (where $\mathbf{e}_2 = [\varepsilon_2,...,\varepsilon_2,...\varepsilon_2,])$

$$
\begin{aligned}
&\max \ \varepsilon_2 \\
&\text{s.t.} \\
&X_k - \sum_{X_i \in C_t} \mu_i X_i - \mathbf{e}_2 \geq 0 \\
&\sum \mu_i = 1 \\
&\mu_i \geq 0
\end{aligned}
$$

If both ε_1 and ε_2 are positive for class *y*, then place X_k to class *y*, go to Step 13. If either ε_1 or ε_2 is equal to zero, again place X_k to class *y*, and go to Step 13.

If $\varepsilon_1 > 0$ and $\varepsilon_2 < 0$ for class y, then equate best possible class index of X_k to *y*, $X_k^B = y$; if $\varepsilon_1 < 0$ and $\varepsilon_2 > 0$ for class y, then equate worst possible class index of X_k to *y*, $X_k^W = y$; if both $\varepsilon_1 < 0$ and $\varepsilon_2 < 0$ for class y, this gives no information; go on with the next class.

If there are less than two previously placed alternatives in a class under consideration, skip that class and consider the next.

When all classes are considered go to next step.

Step 9. (check for best class with WSR)

Starting from the worst class that X_k can belong to, X_k^W, solve the following LP for all possible classes, where *S* representing the weight space.

LP3

$$\begin{aligned}&\max\ \varepsilon\\ &\text{s.t.}\\ &\lambda(X_k - X_h) \geq \varepsilon \qquad \forall X_h \in C_i\\ &\lambda \in S\end{aligned}$$

If the problem is infeasible or $\varepsilon \leq 0$ in the optimal solution, then class *i* is the best class that X_k can belong to, that is $X_k^B = i$, terminate this step. If $X_k^B = X_k^W = y$, then place X_k to class *y*, go to Step 13. If $X_k^B \neq X_k^W$, go to Step 10.

If the problem is feasible and $\varepsilon > 0$ in the optimal solution, go on with the next worst class, loop this step till the class just before the best possible class, $X_k^B + 1$, is considered.

If there are no previously placed alternative in a class under consideration, skip that class and consider the next.

When all classes are considered go to Step 10.

Step 10. (check for worst class with WSR)

Starting from the best class that X_k can belong to, X_k^B, solve the following LP for all possible classes, where *S* representing the weight space.

LP4

$$\begin{aligned}&\max\ \varepsilon\\ &\text{s.t.}\\ &\lambda(X_h - X_k) \geq \varepsilon \quad \forall X_h \in C_i\\ &\lambda \in S\end{aligned}$$

If the problem is infeasible or $\varepsilon \leq 0$ in the optimal solution, then class *i* is the worst class that X_k can belong to, that is $X_k^W = i$, terminate this step. If

$X_k^B = X_k^W = y$, then place X_k to class *y*, go to Step 13. If $X_k^B \neq X_k^W$, go to Step 10.

If the problem is feasible and $\varepsilon > 0$ in the optimal solution, go on with the next best class, loop this step till the class just before the worst possible class, $X_k^W - 1$, is considered.

If there are no previously placed alternative in a class under consideration, skip that class and consider the next.

When all classes are considered go to Step 11.

Step 11. <find a dummy equivalent alternative>

Starting from the best possible class that X_k can belong to, solve the following LP.

LP5

$$\begin{aligned}
&\max \varepsilon \\
&s.t. \\
&\lambda_{est}(X_{dum} - X_k) = 0 \\
&\sum_{X_i \in C_t} \mu_i X_i - X_{dum} \geq \varepsilon \\
&X_{dum} - \sum_{X_i \in C_t} \beta_i X_i \geq \varepsilon \\
&\sum \mu_i = 1 \\
&\sum \beta_i = 1 \\
&\mu_i \geq 0; \beta_i \geq 0; X_{dum} \geq 0; \varepsilon \geq 0
\end{aligned}$$

If a feasible dummy point, X_{dum}, can be found for class *y*, perform an Even-Swap on X_k to compare with X_{dum}. That is step-by-step equating X_k's criterion values to that of X_{dum}'s, and performing the swap. After the Even-Swap check the swapped point, X_k^s, if it can be expressed as convex combination of the alternatives of the class under consideration, go to Step 8.

If no feasible solution can be found, consider the next best class. Loop this step till all possible classes are considered.

If there are less than two previously placed alternatives in a class under consideration, skip that class and consider the next.

When all classes are considered and no feasible solution obtained, go to next step.

Step 12. (ask DM to place)

Ask the DM to place X_k to a preference class among the range of classes X_k^B to X_k^W. If the DM chooses class *y*, go to Step 13.

Go to next step.

Step 13. (assignments, placement by dominance, update best & worst classes)

Assign X_k to class y, that is $X_k^C = y$.

If the alternative is placed directly by the decision maker: add a constraint to the weight space showing $X_k \succ X_d$, that is $\lambda(X_k - X_d) \geq 0$, for all X_d belonging to a worse class and not dominated by X_k. Update the estimated utility function.

If the alternative is placed directly by the decision maker: add a constraint to the weight space showing $X_b \succ X_k$, that is $\lambda(X_b - X_k) \geq 0$, for all X_b belonging to a better class and not dominating X_k. Update the estimated utility function.

If $y = 1$, that is the best class, then all alternatives dominating X_k, D$\{X_k\}$, shall also belong to class 1; place those alternatives to class 1. Recursively, the set of alternatives that are dominating these alternatives shall also be in class 1, check dominating sets for all placed alternatives.

If $y = t$, that is the worst class, then all alternatives that are dominated by X_k,

$D'\{X_k\}$, shall also belong to class *t*; place those alternatives to class *t*. Recursively, the set of alternatives that are dominated by these alternatives shall also be in class *t*, check sets of dominated alternatives for all placed alternatives.

If *y* is an intermediate class, the best possible class index for the set of alternatives that are dominated by X_k shall be at least (lower the better) equal to *y*; if there exist some alternatives having best possible class index lower than *y*, in the dominated alternatives set, equate their index to *y*, that is $X^B = y$.

Again, if *y* is an intermediate class, the worst possible class index for the set of alternatives that are dominating X_k shall be at most (higher the worse) equal to *y*; if there exist some alternatives having best possible class index lower than *y*, in the dominating alternatives set, equate their index to *y*, that is $X^W = y$.

If all the alternatives are placed, exit the algorithm and present the preference classes to the DM. If there are some unplaced alternatives left, go to Step 6.

An Example

The algorithm is illustrated on an example, with 20 alternatives having values on two criteria. Assume that more is better in both criteria. The DM tries to sort these 20 alternatives in three preference classes. The consistency index is selected to be, $\alpha = 0.05$. The alternatives are presented in Figure 9, and the alternative IDs and criterion values are tabulated in Table 10.

While executing the algorithm for this example, two settings are made to simulate DM's responses:

- a linear underlying utility function of $U(X_i) = 0.7 * X_{i,1} + 0.3 * X_{i,2}$ is set, and DM's responses are declared accordingly.
- DM's boundaries between preference classes are set like the following:
    - If $U(X_i) > 0.650$ then place X_i to class 1, $X_i \in C_1$
    - If $0.250 < U(X_i) \leq 0.650$ then place X_i to class 2, $X_i \in C_2$

- If $U(X_i) \leq 0.250$ then place X_i to class 3, $X_i \in C_3$

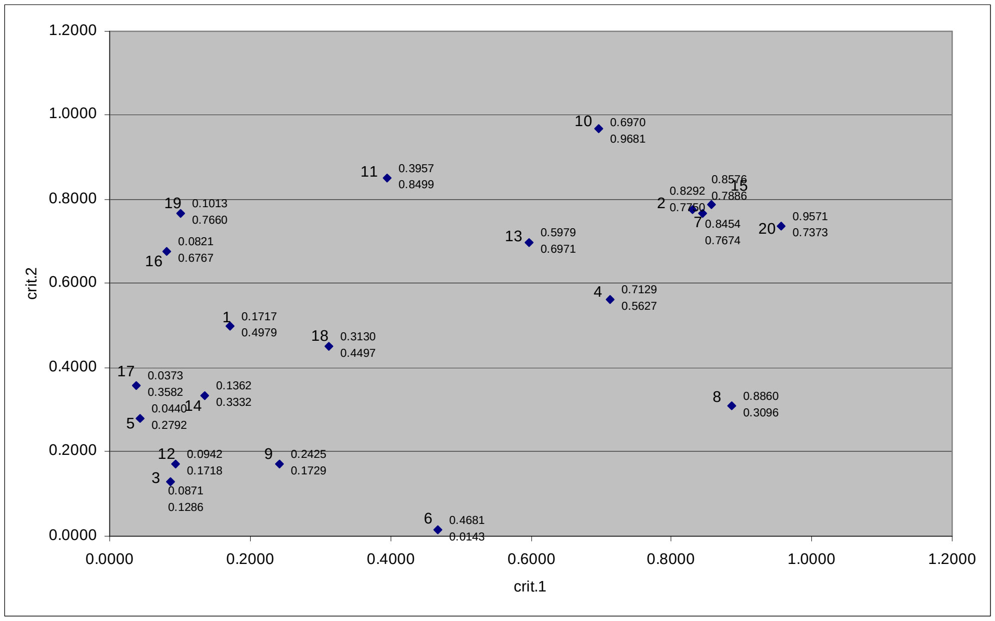


Figure 9. Example - Alternatives graphically represented

Table 10. Example - List of alternatives and their criteria values

Alt ID	crit.1	crit.2
1	0.1717	0.4979
2	0.8292	0.7750
3	0.0871	0.1286
4	0.7129	0.5627
5	0.0440	0.2792
6	0.4681	0.0143
7	0.8454	0.7674
8	0.8860	0.3096
9	0.2425	0.1729
10	0.6970	0.9681

Alt ID	crit.1	crit.2
11	0.3957	0.8499
12	0.0942	0.1718
13	0.5979	0.6971
14	0.1362	0.3332
15	0.8576	0.7886
16	0.0821	0.6767
17	0.0373	0.3582
18	0.3130	0.4497
19	0.1013	0.7660
20	0.9571	0.7373

Step 1.

Euclidean distance between all alternatives are calculated, the closest alternatives which are not dominating each other are alternatives 2 and 7. These two alternatives are selected for initial Even-Swap.

Step 2.

An Even-Swap will be performed on alternative 2 (0.8292, 0.7750) and alternative 7 (0.8454, 0.7674). Let the swap be performed on alternative 2. For alternative 2:

Crit 1:	0.8292 → 0.8454
Crit 2:	0.7750 → ?

Assume that asking the DM to make the swap to compensate the change, the following response is taken: “the increase in the first criterion from 0.8292 to 0.8454 is equal to a decrease in the second criterion from 0.7750 to 0.7372”

Step 3.

Then, alternative $2^{'}$ becomes (0.8454, 0.7372), and this swapped alternative is dominated by alternative 7. Then it can be said that X_7 is preferred to X_2, and following constraint can be written:

$X_7 \succ X_2$

Following two constraints can be generated from the swap performed and using the consistency index α:

$\lambda_1(1+0.05)\dfrac{x^s{}_{2,1} - x_{\ 2,1}}{x_{\ 2,2} - x^s{}_{2,2}} - \lambda_2 \geq 0$ which gives $0.450\lambda_1 - \lambda_2 \geq 0$

and

$\lambda_2 - \lambda_1(1-0.05)\dfrac{x^s{}_{2,1} - x_{\ 2,1}}{x_{\ 2,2} - x^s{}_{2,2}} \geq 0$ which gives $\lambda_2 - 0.407\lambda_1 \geq 0$

Step 4.

The following LP will be solved to make an estimation of the DM’s criteria weights:

$$\begin{aligned}
&\max \varepsilon \\
&s.t. \\
&0.450\lambda_1 - \lambda_2 \geq \varepsilon \\
&\lambda_2 - 0.407\lambda_1 \geq \varepsilon \\
&\lambda(X_7 - X_2) \geq \varepsilon \qquad\qquad \textit{for } X_7 \succ X_2 \\
&\sum \lambda = 1 \\
&\lambda \geq 0 \\
&\varepsilon \geq 0
\end{aligned}$$

Initial estimated weights are found to be: $\lambda_1 = 0.7075$ and $\lambda_2 = 0.2925$.

Step 5.

All best possible class indexes are equalized to 1 ($X_i^B = 1$ for all *i*), all worst possible class indexes are equalized to 3 ($X_i^W = 3$ for all *i*) and all class indexes are equalized to zero, since no alternatives have been placed yet ($X_i^C = 0$ for all *i*).

Step 6.

Alternative 4 is selected to be placed. Since all the preference classes are initially empty, none of the steps give results till Step 12; the algorithm steps forward to Step 12 (DM placement).

Step 12.

The DM is asked to place X_4, and he places X_4 to class 1. Step forward to Step 13.

Step 13.

$X_4^C = 1$, and all alternatives dominating X_4 will also be placed in class 1, these are alternatives 2, 7, 15, 20. Then, $X_2^C = X_7^C = X_{15}^C = X_{20}^C = 1$. Select a new alternative and continue. Step forward to Step 6.

Step 6.

Alternative 13 is selected to be placed. Since some preference classes are empty, none of the steps give results till Step 12; the algorithm steps forward to Step 12 (DM placement).

Step 12.

The DM is asked to place X_{13}, and he places X_{13} to class 2. Step forward to Step 13.

Step 13.

$X_{13}^{C} = 2$, update possible worst class indexes for all unplaced alternatives dominating X_{13} to class 2, there is only one X_{10}, $X_{10}^{W} = 2$. Update possible best class indexes for all unplaced alternatives dominated by X_{13} to class 2, there are 10 alternatives dominated by X_{13}, $X_1^B = X_3^B = X_5^B = X_6^B = X_9^B = X_{12}^B = X_{14}^B = X_{16}^B = X_{17}^B = X_{18}^B = 2$.

Since X_{13} is not dominated by X_4 but in a worse class, a preference relation is implied, $X_4 \succ X_{13}$, and the following constraint is added to weight space:

$\lambda(X_4 - X_{13}) \geq 0$

Solving the LP for estimated utility function again with the new constraint, it is seen that the estimated weights do not change. Select a new alternative and continue. Step forward to Step 6.

Step 6.

Alternative 16 is selected to be placed. Since some preference classes are empty, none of the steps give results till Step 12; the algorithm steps forward to Step 12 (DM placement).

Step 12.

The DM is asked to place X_{16}, and he places X_{16} to class 2. Step forward to Step 13.

Step 13.

$X_{16}^{C} = 2$, update possible worst class indexes for all unplaced alternatives dominating X_{16} to class 2, there are two unplaced dominating alternatives, $X_{11}^{W} = X_{19}^{W} = 2$. Update possible best class indexes for all unplaced alternatives dominated by X_{16} to class 2, there are 2 alternatives dominated by X_{16}, alternatives 5 and 17, but their worst class index is already 2, no update required.

Since X_{16} is not dominated by X_4 but in a worse class, a preference relation is implied, $X_4 \succ X_{16}$, and the following constraint is added to weight space:

$\lambda(X_4 - X_{16}) \geq 0$

Solving the LP for estimated utility function again with the new constraint, it is seen that the estimated weights do not change. Select a new alternative and continue. Step forward to Step 6.

Step 6.

Alternative 9 is selected to be placed. Since some preference classes are empty none of the steps give results till Step 12; the algorithm steps forward to Step 12 (DM placement).

Step 12.

The DM is asked to place X_9, and he places X_9 to class 3. Step forward to Step 13.

Step 13.

$X_9^C = 3$, and all alternatives dominated by X_9 shall also be placed in class 3, these are alternatives 3 and 12. Then, $X_3^C = X_{12}^C = 3$.

Since X_9 is not dominated by X_{16} but in a worse class, a preference relation is implied, $X_{16} \succ X_9$, and the following constraint is added to weight space:

$\lambda(X_{16} - X_9) \geq 0$

Solving the LP for estimated utility function again with the new constraint, it is seen that the estimated weights do not change.

Currently 10 alternatives are placed, and the algorithm status is shown in Table 11, highlighted alternatives are placed:

Table 11. Example - Current status of the alternatives (1)

Alternative ID	class	best	worst
1	0	2	3
2	1	1	1
3	3	3	3
4	1	1	1
5	0	2	3
6	0	2	3
7	1	1	1
8	0	1	3
9	3	2	3
10	0	1	2
11	0	1	2
12	3	3	3
13	2	2	2
14	0	2	3
15	1	1	1
16	2	2	2
17	0	2	3
18	0	2	3
19	0	1	2
20	1	1	1

classes		
1	2	3
2	13	3
4	16	9
7		12
15		
20		

Select a new alternative and continue. Step forward to Step 6.

Step 6.

Alternative 8 is selected to be placed.

Step 7.

$X_8^B = 1$ and $X_8^W = 3$.

Step 8.

Starting from best class that X_8 can belong to, that is class 1; convex combination check will be performed, by solving LP1 and LP2. For the first class LP1 gives $\varepsilon_1 > 0$ and LP2 gives $\varepsilon_2 < 0$; this implies $X_8^B = 1$. For the second class, both $\varepsilon_1 < 0$ and $\varepsilon_2 < 0$; no information gained. For the third class $\varepsilon_1 < 0$ and $\varepsilon_2 > 0$; this implies $X_8^W = 3$.

Step forward to next step.

Step 9.

Starting from worst class that X_8 can belong to, that is class 3; LP3 will be solved. For both class 3 and 2 LP3 gives feasible and positive solutions; so, no updates to X_8^B. Step forward to next step.

Step 10.

Starting from best class that X_8 can belong to, that is class 1; LP4 will be solved. For class 1, LP4 gives a feasible solution, but $\varepsilon < 0$; so the worst class that X_8 can belong to is updated, $X_8^W = 1$.

Now, $X_8^B = X_8^W = 1$, then place X_8 to class 1. Go to Step 13.

Step 13.

$X_8^C = 1$, no updates needed for best and worst possible class indexes of other alternatives due to dominance relations.

Select a new alternative and continue. Step forward to Step 6.

Step 6.

Alternative 11 is selected to be placed.

Step 7.

$X_{11}^B = 1$ and $X_{11}^W = 2$.

Step 8.

Starting from best class that X_{11} can belong to, that is class 1; convex combination check will be performed, by solving LP1 and LP2. For the first class LP1 gives $\varepsilon_1 < 0$ and LP2 gives $\varepsilon_2 < 0$; no information gained. For the second class, both $\varepsilon_1 < 0$ and $\varepsilon_2 > 0$; this implies $X_{11}^W = 2$.

Step 9.

Starting from worst class that X_{11} can belong to, that is class 2; LP3 will be solved. For both class 2 LP3 gives feasible solution, but $\varepsilon < 0$; so the best class that X_{11} can belong to is updated, $X_{11}^B = 2$.

Now, $X_{11}^{B} = X_{11}^{W} = 2$, then place X_{11} to class 2. Go to Step 13.

Step 13.

$X_{11}^{C} = 2$, update possible best class indexes for all unplaced alternatives dominated by X_{11} to class 2, there is only one unplaced alternative dominated by X_{11}, alternative 19, $X_{19}^{B} = 2$.

Now, $X_{19}^{B} = X_{19}^{W} = 2$, then place X_{11} to class 2. X_{19} will be placed to class 2.

Go to Step 13 to place alternative 19.

Step 13.

$X_{19}^{C} = 2$, no updates required for best and worst possible indexes of classes.

Select a new alternative and continue. Step forward to Step 6.

Step 6.

Alternative 10 is selected to be placed.

Step 7.

$X_{10}^{B} = 1$ and $X_{10}^{W} = 2$.

Step 8.

Starting from best class that X_{10} can belong to, that is class 1; convex combination check will be performed, by solving LP1 and LP2. For the first class LP1 gives $\varepsilon_1 < 0$ and LP2 gives $\varepsilon_2 < 0$; no information gained. For the second class, $\varepsilon_1 < 0$ and $\varepsilon_2 > 0$; this implies $X_{10}^{W} = 2$.

Step forward to next step.

Step 9.

Starting from worst class that X_{10} can belong to, that is class 2; LP3 will be solved. Solving for both class 2, LP3 gives feasible positive solution, so no update to X_{10}^{B} Go to Step 10.

Step 10.

Starting from best class that X_{10} can belong to, that is class 1; LP4 will be solved. For class 1, LP4 gives a feasible solution, but $\varepsilon < 0$; so the worst class that X_{10} can belong to is updated, $X_{10}^{W} = 1$.

Now, $X_{10}^{B} = X_{10}^{W} = 1$, then place X_{10} to class 1. Go to Step 13.

Step 13.

$X_{10}^{C} = 1$, no updates required for best and worst possible indexes of classes, and no preference relations implied. Select a new alternative and continue. Step forward to Step 6.

Step 6.

Alternative 14 is selected to be placed.

Step 7.

$X_{14}^{B} = 2$ and $X_{14}^{W} = 3$.

Step 8.

Starting from best class that X_{14} can belong to, that is class 2; convex combination check will be performed, by solving LP1 and LP2. For class 2 LP1 gives $\varepsilon_1 > 0$ and LP2 gives $\varepsilon_2 < 0$; this implies $X_{14}^{B} = 2$. For the second class, $\varepsilon_1 < 0$ and $\varepsilon_2 > 0$; this implies $X_{14}^{W} = 3$.

Step forward to next step.

Step 9.

Starting from worst class that X_{14} can belong to, that is class 3; LP3 will be solved. For class 3 LP3 gives feasible solution, but $\varepsilon < 0$; so the best class that X_{14} can belong to is updated, $X_{14}^{B} = 3$.

Now, $X_{14}^{B} = X_{14}^{W} = 3$, then place X_{14} to class 3. Go to Step 13.

Step 13.

$X_{14}^{C} = 3$, no updates required for best and worst possible indexes of classes.

Select a new alternative and continue. Step forward to Step 6.

Step 6.

Alternative 1 is selected to be placed.

Step 7.

$X_{1}^{B} = 2$ and $X_{1}^{W} = 3$.

Step 8.

Starting from best class that X_1 can belong to, that is class 2; convex combination check will be performed, by solving LP1 and LP2. For class 2 LP1 gives $\varepsilon_1 > 0$ and LP2 gives $\varepsilon_2 < 0$; this implies $X_{1}^{B} = 2$. For class 3, $\varepsilon_1 < 0$ and $\varepsilon_2 > 0$; this implies $X_{1}^{W} = 3$.

Step forward to next step.

Step 9.

Starting from worst class that X_1 can belong to, that is class 3; LP3 will be solved. For class 3 LP3 gives feasible positive solution, so no update on X_{1}^{B}.

Step 10.

Starting from best class that X_1 can belong to, that is class 2; LP4 will be solved. For class 1, LP4 gives a feasible solution, but $\varepsilon < 0$; so the worst class that X_1 can belong to is updated, $X_{1}^{W} = 2$. Now, $X_{1}^{B} = X_{1}^{W} = 2$, then place X_1 to class 2.

Go to Step 13.

Step 13.

$X_1^C = 2$, no updates required for best and worst possible indexes of classes.

Currently 16 alternatives are placed, and the algorithm status is shown in the below table, highlighted alternatives are placed:

Table 12. Example - Current status of the alternatives (2)

Alternative ID	class	best	worst
1	2	2	2
2	1	1	1
3	3	3	3
4	1	1	1
5	0	2	3
6	0	2	3
7	1	1	1
8	1	1	1
9	3	2	3
10	1	1	1

Alternative ID	class	best	worst
11	2	2	2
12	3	3	3
13	2	2	2
14	3	3	3
15	1	1	1
16	2	2	2
17	0	2	3
18	0	2	3
19	2	2	2
20	1	1	1

classes		
1	2	3
2	13	3
4	16	9
7	11	12
15	19	14
20	1	
8		
10		

Remaining four alternatives are also placed by weight space reduction. X_5 and X_{17} are placed to class 3, X_6 and X_{18} are placed to class 2. The final placements are given in the below table (Table 13), and shown graphically in Figure 10.

Table 13. Example - Final status

classes		
1	2	3
2	13	3
4	16	9
7	11	12
15	19	14
20	1	5
8	6	17
10	18	

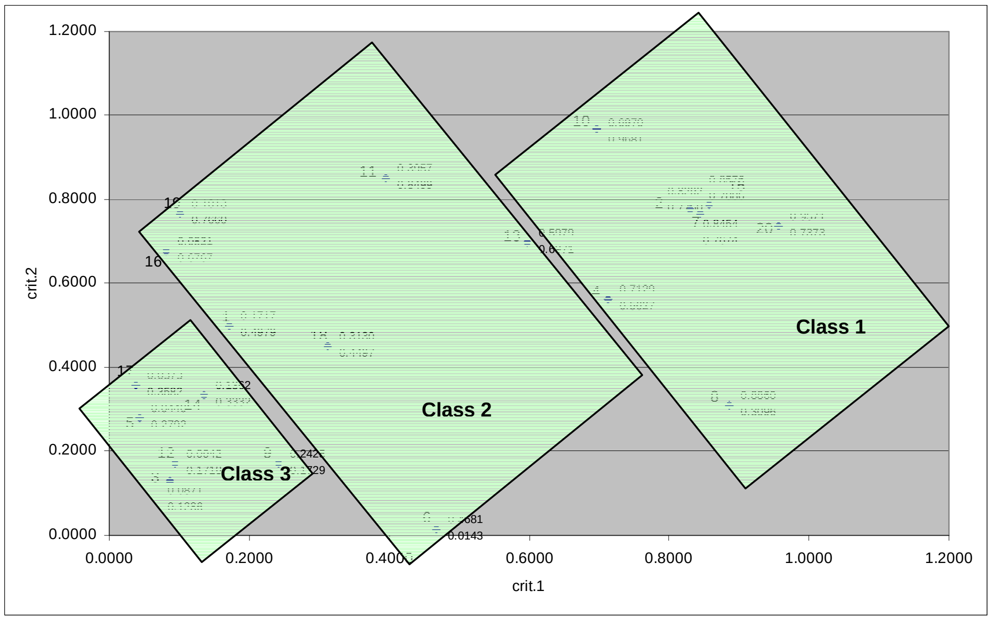


Figure 10. Example - Final status represented graphically

Below table shows how the alternatives are placed, weight space reduction (WSR), dominance (DOM) or DM placement (DM). 9 placements are made by WSR, 7 placements are by DOM and 4 by DM placements. Looking at the alternatives that are placed by DM, it is observed that these are the alternatives which constitute the boundary for the classes, this proves the effectiveness of selecting alternatives for placement technique; all other alternatives are placed either by WSR or DOM. Only one even-swap is required, which was at the initialization phase of the algorithm.

Table 14. Example - Means of placements

WSR		DOM		DM	
Alternative ID	Class ID	Alternative ID	Class ID	Alternative ID	Class ID
8	1	2	1	4	1
11	2	7	1	13	2
10	1	15	1	16	2
14	3	20	1	9	3
1	2	3	3		
6	2	12	3		
18	2	19	2		
17	3	5	3		

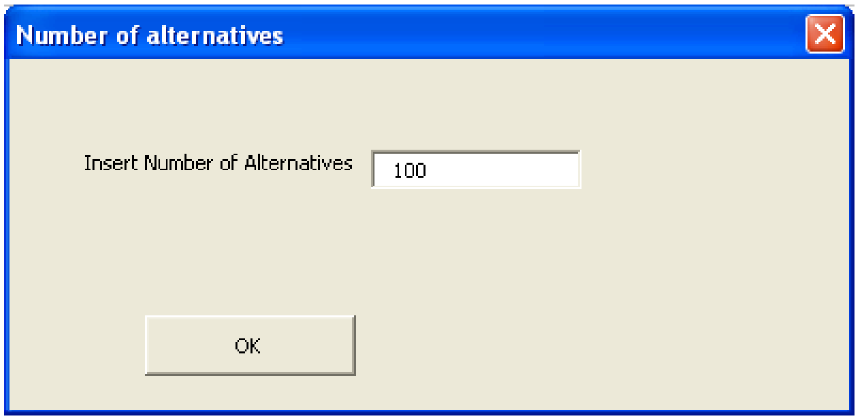
CHAPTER 4

AUTOMATED APPROACH AND EXPERIMENTATION

An automation of the proposed algorithm is developed for bi-criteria problems in order to provide an infrastructure for DMs to implement the algorithm. The developed automation is also used to test and interpret the behavior of the algorithm to problems with certain characteristics. The extension of the automation to more than two criteria problems may result in longer run-times to execute and some complications especially dealing with Even-Swaps.

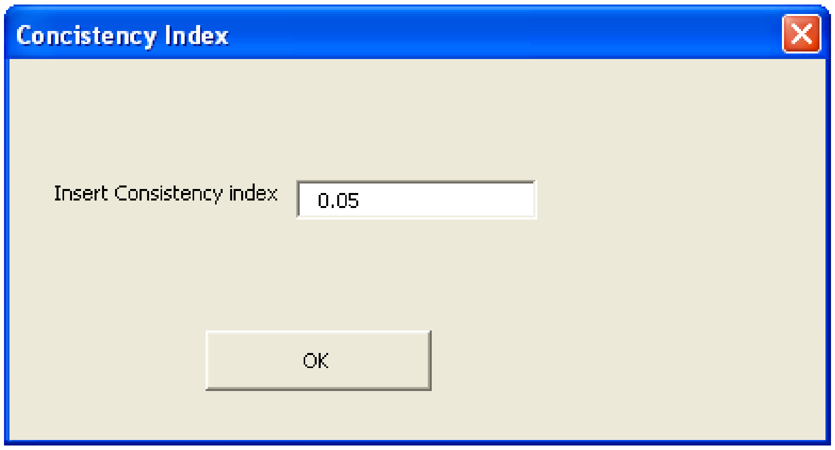
4.1 Development of the Automation and User Screens

The automation is developed using Visual Basic with MS Excel, and utilizes Excel Solver for the LPs in the algorithm. The alternatives are read from a worksheet and model parameters - number of alternatives to be considered and the consistency index - are taken interactively. The following snapshots show interaction screens; Figure 11 for inserting number of alternatives and Figure 12 for inserting consistency index.



Insert Number of Alternatives 100

Figure 11. Inserting number of alternatives to be considered



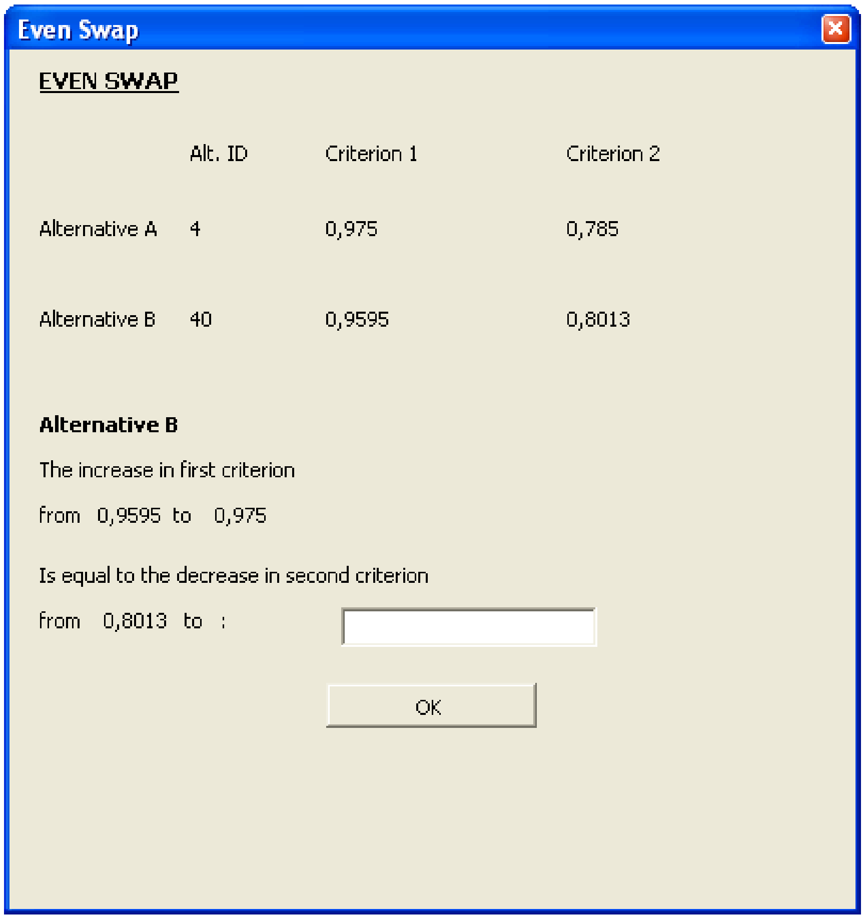
Insert Consistency index 0.05

Figure 12. Inserting the consistency index

Two other interaction points with the DM are:

- Performing the Even-Swap : Step 2 at the initialization phase and Step 11 at the placing alternatives phase.
- Placing alternatives directly : Step 12 at the placing alternatives phase.

The first one requires DM to perform even swaps. Whenever an Even-Swap is required, the screen shown in Figure 13 appears and asks the DM the decrease or increase in the value of one criterion to compensate the change in the other criterion. The criteria values for the both alternatives, and the change in the other criterion is presented to the DM, and s/he is expected to make the swap.



Even Swap

EVEN SWAP

	Alt. ID	Criterion 1	Criterion 2
Alternative A	4	0,975	0,785
Alternative B	40	0,9595	0,8013

Alternative B

The increase in first criterion

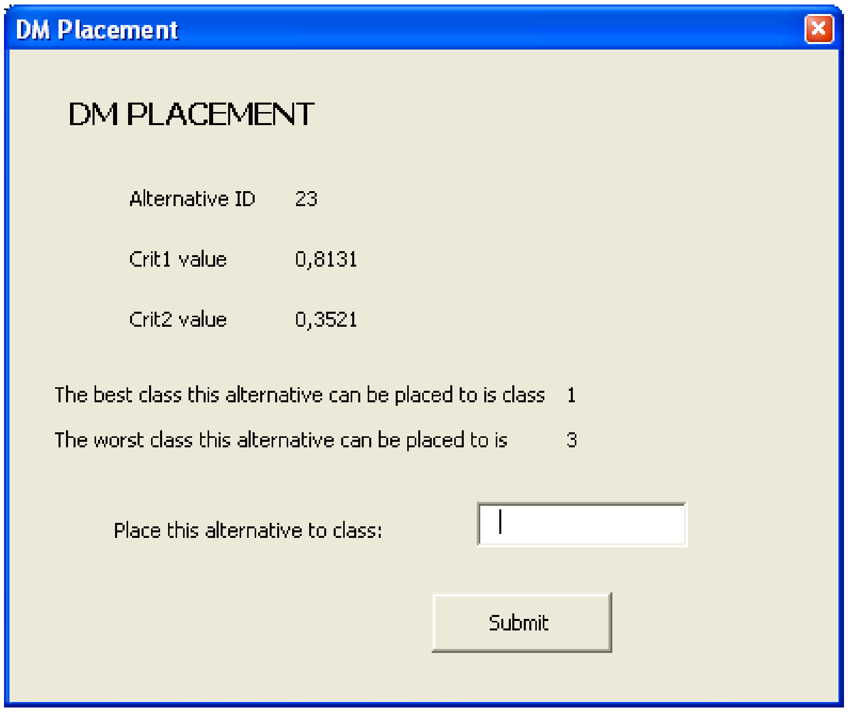
from 0,9595 to 0,975

Is equal to the decrease in second criterion

from 0,8013 to :

Figure 13. Even Swap Screen

If none of the former steps can place the selected alternative, the algorithm asks the DM to place the alternative to a class in between its possible best and worst classes. These best and worst possible class indexes, and the criteria values of the alternative is presented to the DM in the screen shown in Figure 14, and s/he is expected to place the alternative in one class.


DM Placement

DM PLACEMENT

Alternative ID	23
Crit1 value	0,8131
Crit2 value	0,3521

The best class this alternative can be placed to is class 1

The worst class this alternative can be placed to is 3

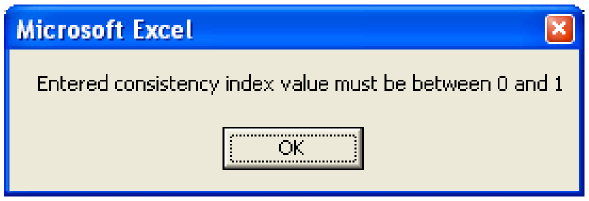
Place this alternative to class:

Figure 14. DM Placement

Entering correct info is crucial, so some checks are done in order to reject erroneous data entry, error messages appear when one of the following cases occur:

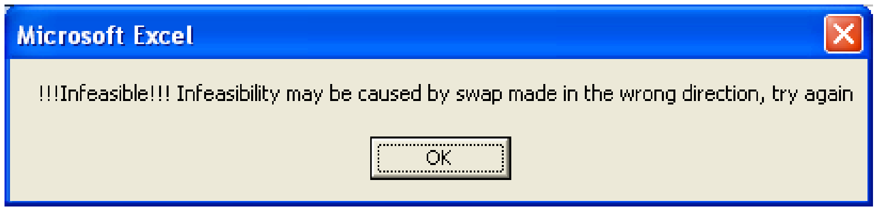
- Consistency index not between 0 and 1. (Figure 15)
- When swap is done in the wrong direction. (Figure 16)
- When DM tries to place the alternative out of the presented best and worst possible class indexes bound. (Figure 17)

The following error messages appear, and the algorithm continues when the error is corrected.


Microsoft Excel

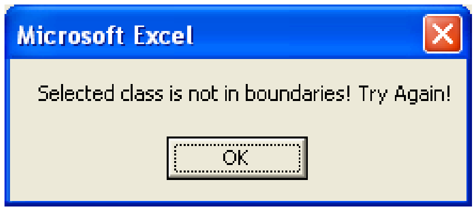
Entered consistency index value must be between 0 and 1

Figure 15. Consistency index not valid


Microsoft Excel

!!!Infeasible!!! Infeasibility may be caused by swap made in the wrong direction, try again

Figure 16. Swap done in the wrong direction


Microsoft Excel

Selected class is not in boundaries! Try Again!

Figure 17. DM placed the alternative to a wrong class

4.2 Experimentation

The proposed algorithm is experimented with the developed automation, in order to analyze the behavior of the algorithm to alternative sets with different characteristics. Four different parameters can be considered when testing the algorithm:

- number of classes: this parameter is fixed at 3 for all runs, the implementation may be extended to handle more classes, but currently it places alternatives to three preference classes
- number of alternatives: three different sizes of alternative sets are used, sets with 20, 50 and 100 alternatives are considered
- weights of the utility function: two different weight sets are used, "weight1 / weight2" ratio of the first one is 0.7/0.3 and the other one is 0.1/0.9.
- consistency index value, alpha: three different consistency indexes are used, these are; 0.05, 0.15 and 0.30.

The following table summarizes the runs made, with their run codes and run parameters (Table 15):

Table 15. Run codes and Run parameters

Run Codes	# of Classes	# of Alternatives	Utility (weight 1 / weight 2)	Alpha
Run001	3	20	0,7 / 0,3	0.05
Run002	3	20	0,7 / 0,3	0.15
Run003	3	20	0,7 / 0,3	0.30
Run004	3	20	0,1 / 0,9	0.05
Run005	3	20	0,1 / 0,9	0.15
Run006	3	20	0,1 / 0,9	0.30
Run007	3	50	0,7 / 0,3	0.05
Run008	3	50	0,7 / 0,3	0.15
Run009	3	50	0,7 / 0,3	0.30
Run010	3	50	0,1 / 0,9	0.05
Run011	3	50	0,1 / 0,9	0.15
Run012	3	50	0,1 / 0,9	0.30
Run013	3	100	0,7 / 0,3	0.05
Run014	3	100	0,7 / 0,3	0.15
Run015	3	100	0,7 / 0,3	0.30
Run016	3	100	0,1 / 0,9	0.05
Run017	3	100	0,1 / 0,9	0.15
Run018	3	100	0,1 / 0,9	0.30

During the runs, the DM's responses for direct placements and Even-Swaps are given using the underlying utility function of the DM. It is assumed that all preference classes are approximately of the same size. Three different alternative sets of sizes 20 alternatives, 50 alternatives and 100 alternatives are used, whose criteria values are randomly generated. The alternative sets and their criteria values are given in Appendix B. How each alternative is placed - either by convex combinations, weight space reduction, Even-Swaps, direct DM placement or dominance - is tracked and documented. Results of all runs are given in Appendix B. The Following table (Table 16) summarizes the results obtained from all runs.

Table 16. Run Results Summary

	# of Classes	# of Alternatives	Utility (weight1/ weight2)	Alpha	Number of alternatives placed with different tools				
					Convex Combination	WSR	Even Swaps	DM Placement	Dominance
Run001	3	20	0,7 / 0,3	0.05		8		4	8
Run002	3	20	0,7 / 0,3	0.15		7	1	4	8
Run003	3	20	0,7 / 0,3	0.30	1	5	2	4	8
Run004	3	20	0,1 / 0,9	0.05		5		4	11
Run005	3	20	0,1 / 0,9	0.15		5		4	11
Run006	3	20	0,1 / 0,9	0.30		5		4	11
Run007	3	50	0,7 / 0,3	0.05	5	18		5	22
Run008	3	50	0,7 / 0,3	0.15	4	15	4	5	22
Run009	3	50	0,7 / 0,3	0.30	7	9	6	7	21
Run010	3	50	0,1 / 0,9	0.05	6	14		4	26
Run011	3	50	0,1 / 0,9	0.15	6	14		4	26
Run012	3	50	0,1 / 0,9	0.30	6	10	3	5	26
Run013	3	100	0,7 / 0,3	0.05	32	18	8	4	38
Run014	3	100	0,7 / 0,3	0.15	24	22	3	6	45
Run015	3	100	0,7 / 0,3	0.30	35	9	16	6	34
Run016	3	100	0,1 / 0,9	0.05	21	30	0	6	43
Run017	3	100	0,1 / 0,9	0.15	21	29	1	6	43
Run018	3	100	0,1 / 0,9	0.30	28	25	2	6	39

Some graphs are plotted using the above tabulated results, to enable the interpretation of the results graphically. The averages are taken by grouping the runs according to “number of alternatives”, and Figure 18 is plotted. This figure shows the average percent of the cases an alternative is placed by a certain placement method.

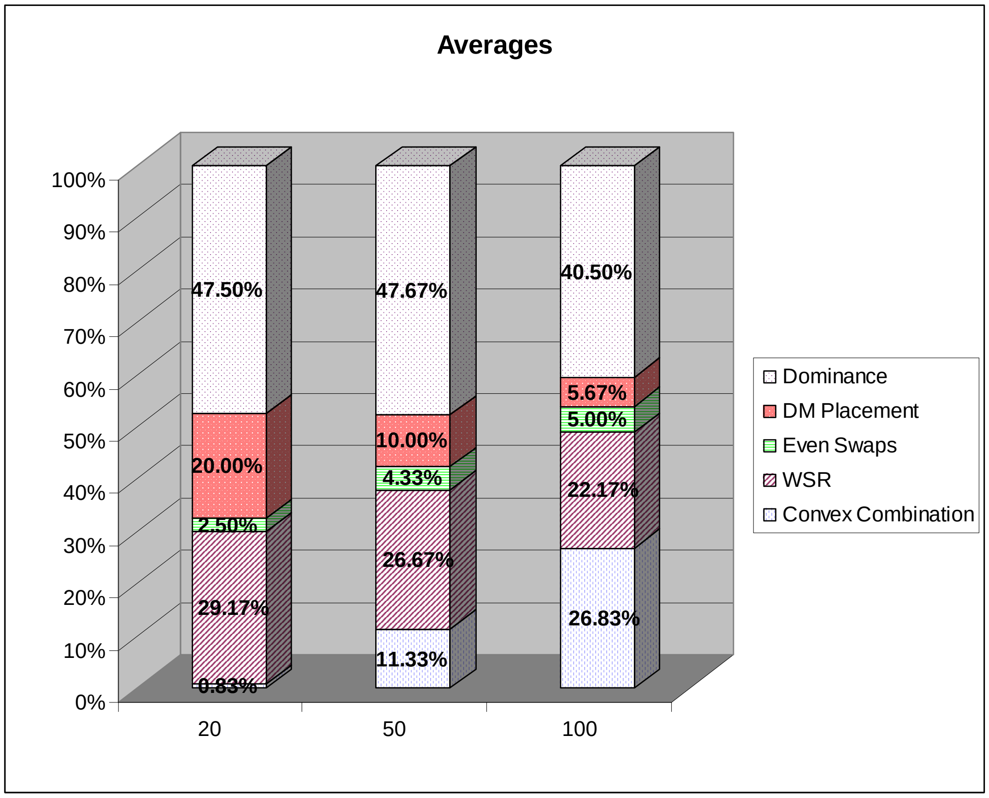


Figure 18. Averages with differing alternative set sizes

By looking at the alternatives placed by direct DM placement, we can observe a decreasing trend. The average percent for Even-Swaps increases as the number of alternatives considered increases, however, if we sum up the average percentages for DM placement and Even-Swaps the following decreasing trend can be observed; 22.50%, 14.33% and 10.67% for alternative sets of sizes 20, 50 and 100 respectively. Thus, we can conclude that with larger alternative sets the algorithm requires relatively less DM effort.

Another observation from this graph is that, the percent of alternatives that are placed by WSR decreases whereas the percent of alternatives that are placed by convex combinations increases as the size increases. This can be due to increasing number of convex combination relationships among all alternatives with the increase in the number of alternatives.

The consistency index is set at the beginning of the algorithm; the information coming from the Even-Swaps is evaluated within the range defined by this consistency index. The average percentages for way of placing the alternatives, under different consistency indexes are plotted in Figure 19. As expected, when the consistency index increases, the required DM input increases, if we sum up the average percentages for DM placement and Even-Swaps the following increasing trend can be observed; 12.67%, 14.50% and 20.33% for consistency index values of 0.05, 0.15 and 0.30 respectively.

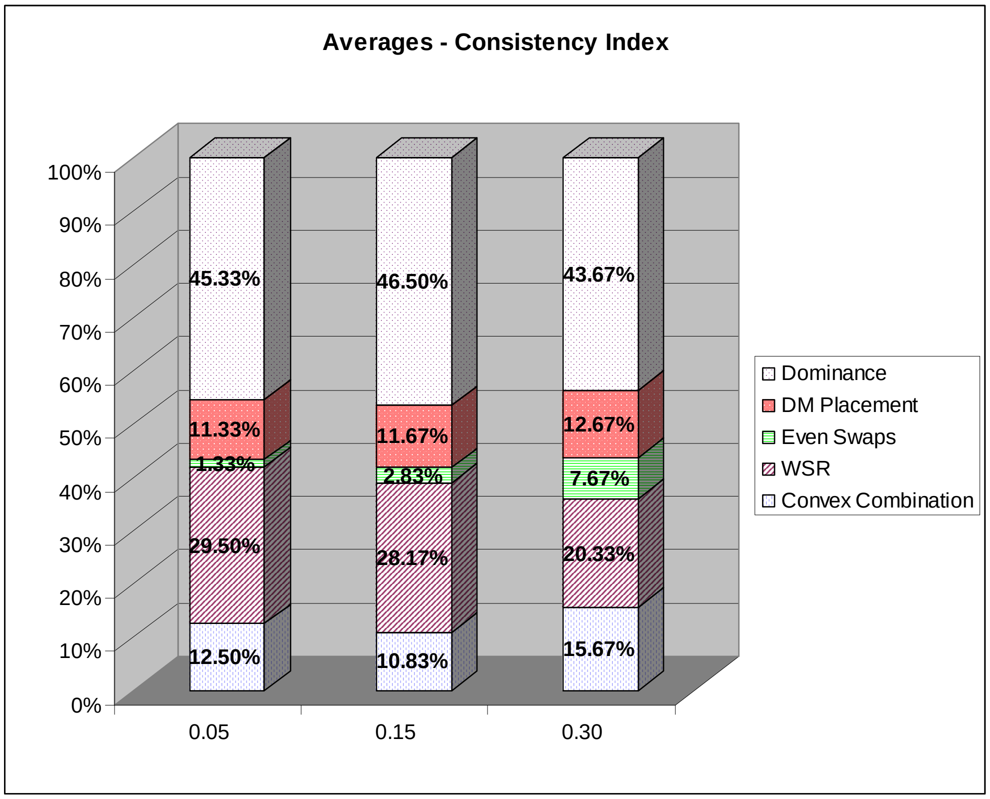


Figure 19. Averages with differing consistency indexes

The results are grouped according to the number of alternatives considered, and three graphs are plotted for alternative sets of sizes 20, 50 and 100. Figure 20 illustrates the runs for the alternative set having 20 alternatives. The most interesting observation from this graph is that, number of alternatives that are placed by direct DM placement stays constant at 4, with different utility functions and consistency indexes. These four alternatives are the ones that constitute the boundaries for preference classes; one for the best class, one for the worst and two for the middle class. The step for placing the alternative by an Even-Swap is invoked only three times; the reasoning behind this is that, since the number of alternatives dealt is small, the case when it is feasible to perform an Even-Swap is less probable. Another observation from Figure 20 is that, for runs using the utility function weight ratio "0.1/0.9", (these are the runs 4, 5 and 6) the results are stable for different consistency indexes.

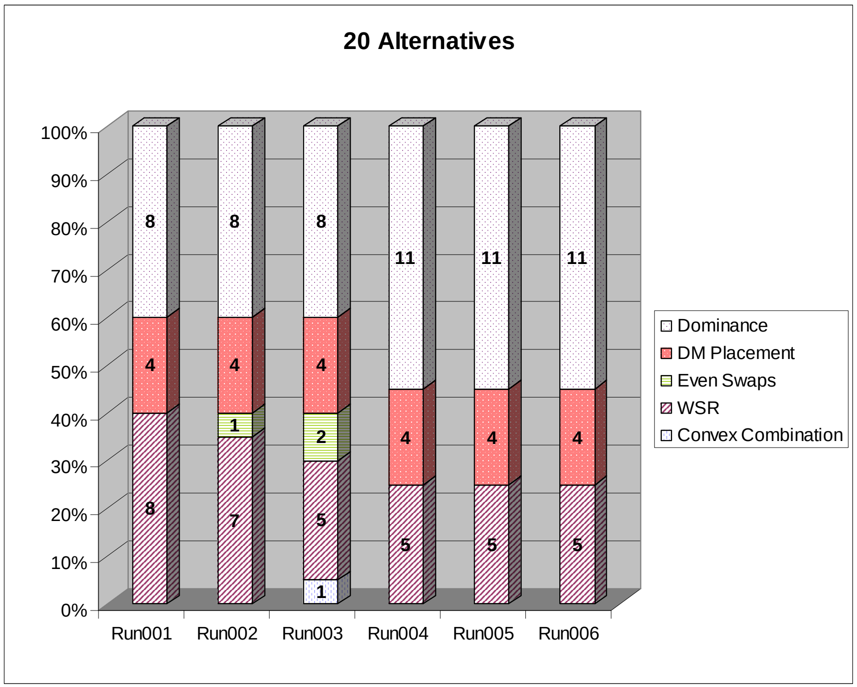


Figure 20. Results for 20 alternatives

Figure 21 and Figure 22, summarize the runs for 50 and 100 alternatives. It can be observed that; as the consistency index increases, the number of DM placements and number of placements by Even-Swaps increases, whereas the number of alternatives that are placed by either WSR or convex combination relations decreases. The results are more stable with the runs that use utility function with weights 0.1 and 0.9 for criterion 1 and criterion 2 respectively than the runs that use weights 0.7 and 0.3.

Although we double the size of the alternative set from 50 to 100, we observe just a slight increase in the number of placements made directly by the DM, so relatively it is decreasing.

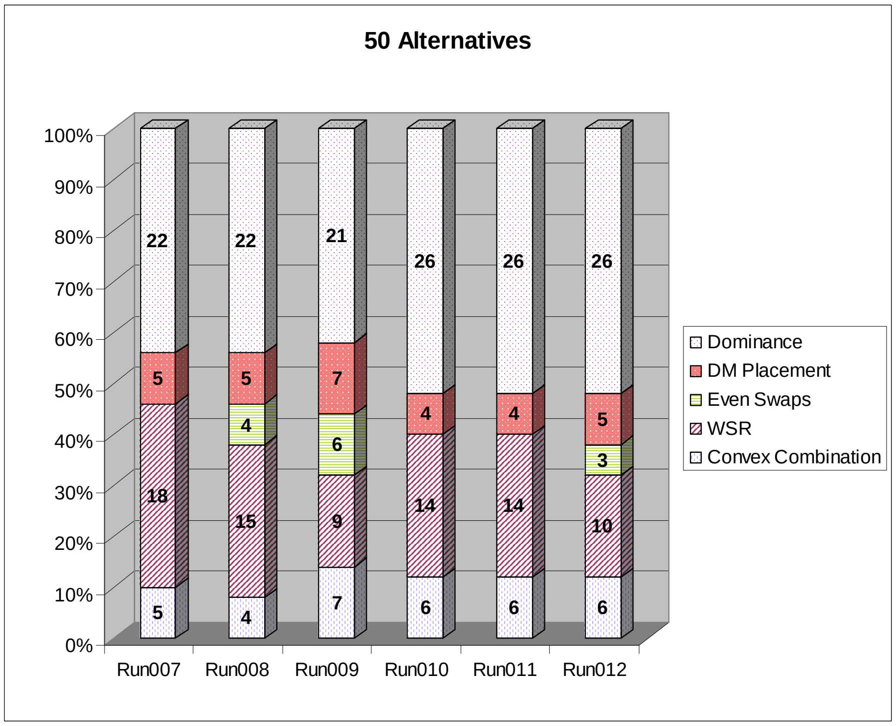


Figure 21. Results for 50 alternatives

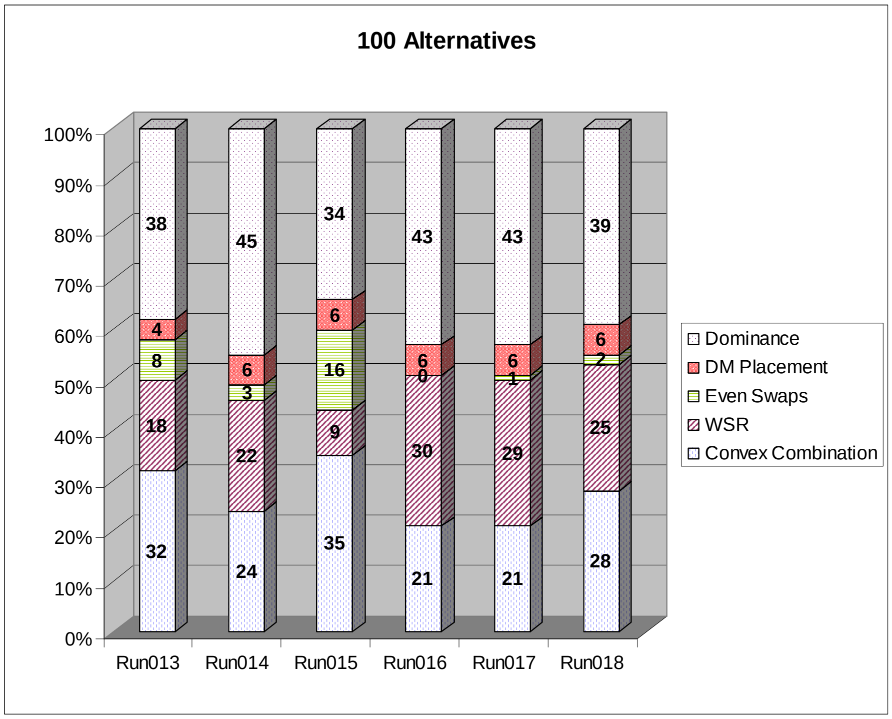


Figure 22. Results for 100 alternatives

The algorithm requires some computational effort; runs with alternative sets having 100 alternatives, required execution of 277 LPs on the average. Run 15, with 0.7/0.3 utility weight ratio and 0.30 consistency index, required execution of 348 LPs. It takes couple of minutes to complete a run.

CHAPTER 5

SUMMARY AND CONCLUSION

An interactive model for the problem of sorting alternatives to preference classes in the existence of multiple criteria is proposed in this study.

It is assumed that the DM's underlying utility function is linear; the model tries to generate an estimated region for the criteria weights using DM's responses and place the alternatives to preference classes either utilizing weight space reduction or dominance wherever possible.

A two-phased approach is proposed for the problem. The first phase initializes the placement algorithm, starts with the selection of two alternatives from the alternative set and an even swap is performed on these two alternatives. Then, utilizing the information gained from this swap, an estimation of the DM's underlying utility function is made. This is done by the help of an LP which constrains the weight space of the DM.

With the estimated utility function, the algorithm steps forward to second phase. This phase loops until all alternatives in the preference set are placed to a preference class. The first step in this phase, is selecting an alternative placement. Then, by looking at dominance relations with formerly placed alternatives, its possible best and worst classes are determined. Next, it is checked whether the selected alternative can be expressed as a convex combination of alternatives that are already assigned to a given class. This check is conducted for all possible classes the selected alternative may belong to. When it is proved that the alternative can be expressed as a convex combination of alternatives of a given class, it is placed to that class. Otherwise, the information gained is utilized and the algorithm tries to place the alternative using weight space reduction with two LPs. If this cannot be possible, the next step tries to find an equivalent dummy point –in terms of estimated utility value- which can be expressed as a convex combination of the alternatives of a class. Then an even swap is performed on the selected alternative to swap around the dummy alternative, and it is expected that the swapped alternative can also be expressed as a convex combination of alternatives in that class. If no feasible dummy alternative can be found, then the selected alternative is presented to the DM, and he/she is asked to place the alternative to one of the possible preference classes.

In each step of the algorithm; whenever information is gathered from the DM, the weight space is updated accordingly. The weight space is reduced during the execution of the algorithm. So, although not guaranteed, it is expected and observed that the DM's estimated utility function improves as the algorithm iterates. As more alternatives are placed to preference classes, the algorithm becomes more effective. That is, as the number of alternatives placed by dominance and weight space reduction increases, the required DM involvement for placements decreases.

At the last step of the algorithm the DM is asked to directly place the selected alternative to a preference class. But before that step, if appropriate, the DM is asked to place the alternative by performing an Even-Swap using the dummy alternative approach. In some cases, performing the Even-swap can be harder than directly placing the alternative. That depends on many factors: the characteristic of the problem, scaling of the criteria, number of criteria to be considered, the DM's value judgments, the amount of the swap required, etc. For example if the number of criteria to be considered is five, the DM has to make four iterations to perform the Even-Swap; s/he would rather prefer directly placing the alternative. However, evaluating the criteria and making the placement may also be difficult, especially when the number of criteria is large.

The algorithm appears to be more effective especially when the number of alternatives in the alternative set is large and the number of criteria to be considered is not too many. Without a proper mechanism, it would be hard for the DM to consider and place too many alternatives; the model provides automatic placements by making inferences. The problem with too many criteria is due to the increase in the number of swaps required.

An automated approach for the proposed algorithm is developed. Utilizing the automation, the model is solved with various alternative sets having different characteristics, and the results are discussed in Chapter 4.

Possible Future Work

It was assumed that, the DM is consistent with his/her responses; it will be interesting to analyze cases when there exists inconsistency, beyond the amount considered by the consistency index. In the same way the consistency index, α, can be defined dynamically as a function of DM's responses.

A real life application of the algorithm can be performed; the possible application areas are listed in the literature survey section.

The Even-Swaps method is originally proposed for finding the best alternative among a small number of alternatives, however, utilizing the ideas developed in this study it can be possible to use the method to find the best alternative among a large number of alternatives, which is a special case of the sorting problem studied in this thesis.

REFERENCES

1. Altman, E.I., Avery, R., Eisenbeis, R., Stinkey, J., 1981. Application of Classification Techniques in Business, Banking and Finance.Contemporary Studies in Economic and Financial Analysis, Vol. 3. JAI Press, Greenwich, CT
2. Arhcer, N.P., Wang, S., 1993. Application of the back propagation neural networks algorithm with monotonicity constraints for twogroup classification problems. Decision Sciences 24, 60–75.
3. Berkson, J., 1944. Application of the logistic function to bio-assay. Journal of the American Statistical Association 39, 357–365.
4. Bliss, C.I., 1934. The method of probits. Science 79, 38–39.
5. Fisher, R.A., 1936. The use of multiple measurements in taxonomic problems. Annals of Eugenics 7, 179–188.
6. Greco, S., Matarazzo, B., Slowinski, R., 1999a. The use of rough sets and fuzzy sets in MCDM. In: Gal, T., Hanne, T., Stewart, T.(Eds.), Advances in Multiple Criteria Decision Making. Kluwer Academic Publishers, Dordrecht, pp.203–211.
7. Greco, S., Matarazzo, B., Slowinski, R., 2000a. Extension of the rough set approach to multicriteria decision support. INFOR 38 (3), 161–196.
8. Greco, S., Matarazzo, B., Slowinski, R., 2000b. Dealing with missing values in rough set analysis of multi-attribute and multi-criteria decision problems. In: Zanakis, S.H., Doukidis, G., Zopounidis, C. (Eds.), Decision Making: Recent Developments and Worldwide Applications. Kluwer Academic Publishers, Dordrecht, pp. 295–316.
9. Greco, S., Matarazzo, B., Slowinski, R., 2002. Rough sets methodology for sorting problems in presence of multiple attributes and criteria. European Journal of Operational Research 138, 247-259.
10. Greco, S., Matarazzo, B., Slowinski, R., Zanakis, S., 1999. Rough set analysis of information tables with missing values. In: Despotis, D.,

Zopounidis, C. (Eds.), Integrating Technology & Human Decisions: Bridging into the 21st Century, Proceedings of the Fifth International Meeting of the Decision Sciences Institute, Vol. II. New Technologies Editions, Athens, pp. 1359–1362.

11. Green, P.E., Srinivasan, V., 1978. Conjoint analysis in consumer research: Issues and outlook. Journal of Consumer Research, 103–123.
12. Gregory, R., K. Wellman. 2001. Bringing stakeholder values into environmental policy choices: a community-based estuary case study. Ecological Economics 39 37-52.
13. Hammond, J.S., R.L. Keeney, H. Raiffa. 1998. Even swaps: A rational method for making trade-offs, Harvard Business Review 76(2) 137-149.
14. Hammond, J.S., R.L. Keeney, H. Raiffa. 1999. Smart Choices. A Practical Guide to Making Better Decisions. Harvard Business School Press, Boston, MA.
15. Jacquet-Lagreze, E., 1995. An application of the UTA discriminant model for the evaluation of R&D projects. In: Pardalos,P.M., Siskos, Y., Zopounidis, C. (Eds.), Advances in Multicriteria Analysis. Kluwer Academic Publishers, Dordrecht, pp.203–211.
16. Jacquet-Lagreze, E., Siskos, Y., 2001. Preference disaggregation: 20 years of MCDA experience. European Journal of Operational Research 130, 233–245.
17. Kajanus, M., J. Ahola, M. Kurttila, M. Pesonen. 2001. Application of even swaps for strategy selection in a rural enterprise. Management Decision 39(5) 394-402.
18. Keeney, R.L., Raiffa, H., 1976. Decisions with Multiple Objectives. Wiley, New York.
19. Keeney, R.L. 2002. Common mistakes in making value trade-offs. Oper. Res. 50 (6) 935-945.
20. Koksalan, M.M., 1989. Identifying and ranking a most preferred subset of alternatives in the presence of multiple criteria. Naval Research Logistics 36, 359–372.
21. Koksalan, M.M., Karwan, M.H., Zionts, S., 1984. An improved method

for solving multiple criteria problems involving discrete alternatives. IEEE Transactions on Systems, Man, and Cybernetics 14, 24–34.

22. Koksalan, M.M., Öden, Ö., 1993. Visual interactive approaches for bi-criteria decision making. Transactions on Operations Research (Yöneylem Araştırması Dergisi) 5, 27-44.
23. Koksalan, M.M., Rizi, O., 2001. A visual interactive approach for multiple criteria decision making with monotone utility functions. Journal of the Operational Research Society 52, 665-672
24. Koksalan, M.M., Sagala, P., 1995. Interactive approaches for discrete alternative multiple criteria decision making with monotone utility functions. Management Science 41, 1158–1171.
25. Koksalan, M.M., Taner, O.V., 1992. An approach for finding the most preferred alternative in the presence of multiple criteria. European Journal of Operational Research 60, 52–60.
26. Koksalan, M.M., Ulu, C., 2003. An interactive approach for placing alternatives in preference classes. European Journal of Operational Research 144 429–439
27. Korhonen, P., Soismaa, M., 1981. An interactive multiple criteria approach for ranking alternatives. Journal of the Operational Research Society 32, 577–585.
28. Korhonen, P., Wallenius, J., Zionts, S., 1984. Solving the discrete multiple criteria problem using convex cones. Management Science 30, 1336–1345.
29. Larichev, O.I., Moshkovich, H.M., 1994. An approach to ordinal classification problems. International Transactions in Operational Research 1, 375–385.
30. Malakooti, B., 1989. Identifying nondominated alternatives with partial information for multiple-objective discrete and linear programming problems. IEEE Transactions on SMC 19, 95–107.
31. Malakooti, B., 2000. Ranking and screening multiple criteria alternatives with partial information and use of ordinal and cardinal strength of preferences. IEEE Transactions on SMC, Part A 30, 355–36

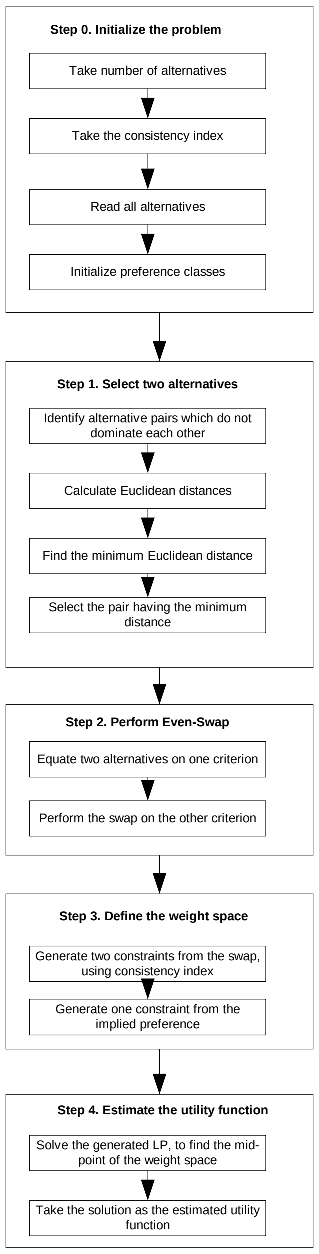
32. Malakooti, B., Raman, V., 2000. Clustering and selecting multiple criteria alternatives using unsupervised and supervised neural networks. Journal of Intelligent Manufacturing 11, 435–451.
33. McFadden, D., 1974. Conditional logit analysis in qualitative choice behavior. In: Zarembka, P. (Ed.), Frontiers in Econometrics.Academic Press, New York.
34. Michalowski, W., Rubin, S., Slowinski, R., Wilk, S., 2001. Triage of the child with abdominal pain: A clinical algorithm for emergency patient management. Paediatrics and Child Health 6 (1), 23–28.
35. Mousseau, V., Slowinski, R., Zielniewicz, P., 2000. A user-oriented implementation of the ELECTRE-TRI method integrating preference elicitation support. Computers and Operations Research 27 (7-8), 75
36. Mousseau, V., Slowinski, R., Zielniewicz, P., 2000. A user-oriented implementation of the ELECTRE-TRI method integrating preference elicitation support. Computers and Operations Research 27, 757–777.
37. Mustajoki, J., Hämäläinen, R.P., 2004. Making Even Swaps Even Easier. Manuscript. www.sal.hut.fi/Publications/
38. Preadki, L., Slowinski, R., Stefanowski, J., Susmaga, R., Wilk, Sz., 1998. ROSE – Software implementation of the rough set theory. In: Polkowski, L., Skowron, A. (Eds.), Rough Sets and Current Trends in Computing, Lecture Notes in Artificial Intelligence, Vol. 1424. Springer, Berlin, pp. 605–608.
39. Roy, B., 1971. Problems and methods with multiple objective functions. Mathematical Programming 1, 239–266.
40. Siskos, J., 1982. A way to deal with fuzzy preference sets in multi-criteria decision problems. European Journal of Operational Research 10, 314–324.
41. Siskos, Y., Grigoroudis, E., Zopounidis, C., Saurais, O., 1998. Measuring customer satisfaction using a survey based preference disaggregation model. Journal of Global Optimization 12 (2), 175–195
42. Stam, A., Ungar, D.R., 1995. RANGU: A microcomputer package for two-group mathematical programming-based nonparametric

classification. European Journal of Operational Research 86, 374–388.

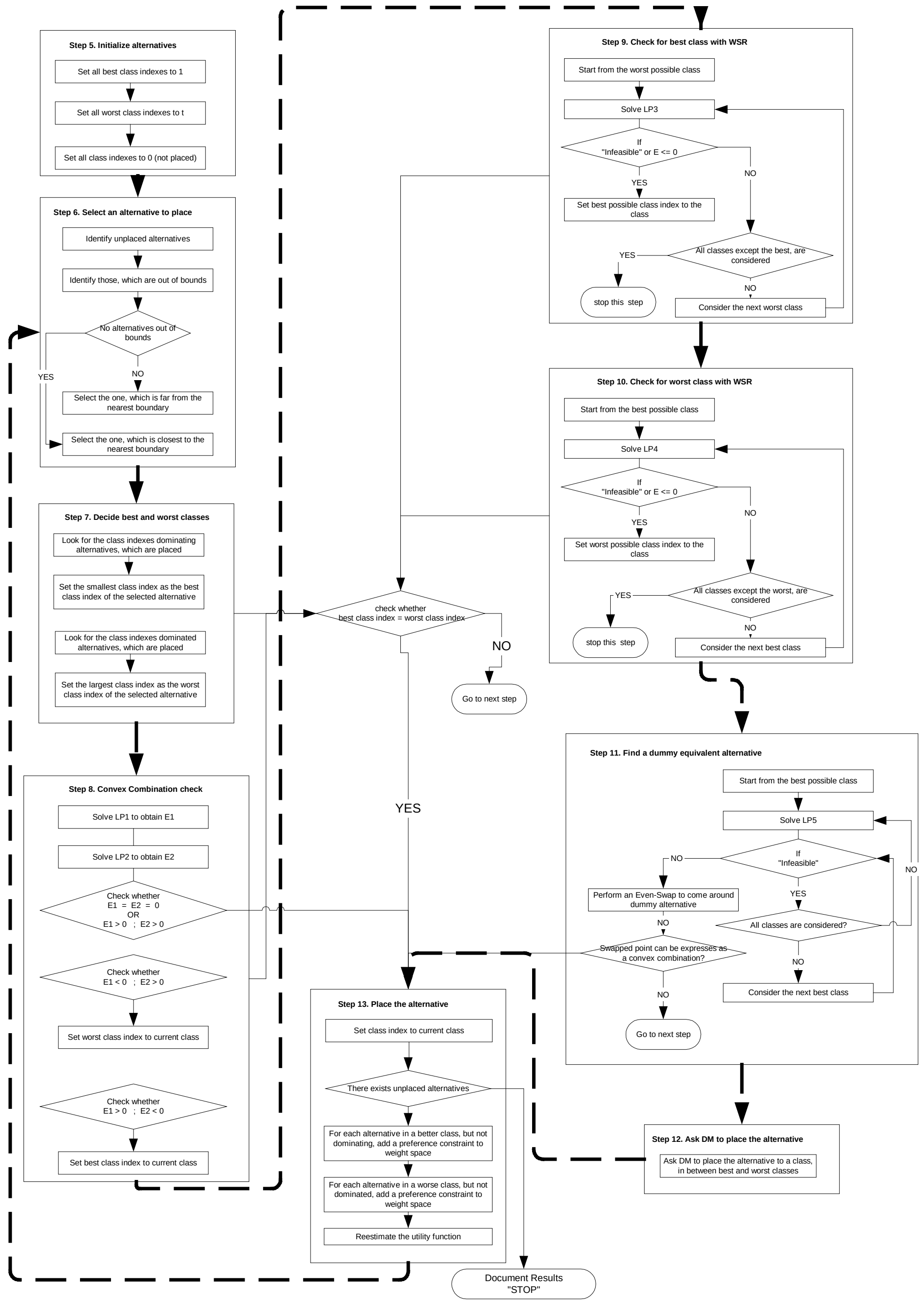
43. Ulu, C., Koksalan, M., 2001. An interactive procedure for selecting acceptable alternatives in the presence of multiple criteria. Naval Research Logistics 48, 592–606.
44. Zionts, S., 1981. A multiple criteria method for choosing among discrete alternatives. European Journal of Operational Research 7, 143–147.
45. Zopounidis, C., Doumpos, M., 1999. A multicriteria decision aid methodology for sorting decision problems: The case of financialdistress. Computational Economics 14 (3), 197–218.
46. Zopounidis, C., Doumpos, M., 2000a. PREFDIS: A multicriteria decision support system for sorting decision problems. Computers and Operations Research 27 (7-8), 779–797.
47. Zopounidis, C., Doumpos, M., 2002. Multicriteria classi.cation and sorting methods: A literature review. European Journal of Operational Research 138, 229–246

APPENDIX A – Detailed flow of the algorithm

Initializing Phase


Step 0. Initialize the problem
Take number of alternatives
Take the consistency index
Read all alternatives
Initialize preference classes
Step 1. Select two alternatives
Identify alternative pairs which do not dominate each other
Calculate Euclidean distances
Find the minimum Euclidean distance
Select the pair having the minimum distance
Step 2. Perform Even-Swap
Equate two alternatives on one criterion
Perform the swap on the other criterion
Step 3. Define the weight space
Generate two constraints from the swap, using consistency index
Generate one constraint from the implied preference
Step 4. Estimate the utility function
Solve the generated LP, to find the mid-point of the weight space
Take the solution as the estimated utility function

Placing Alternatives Phase



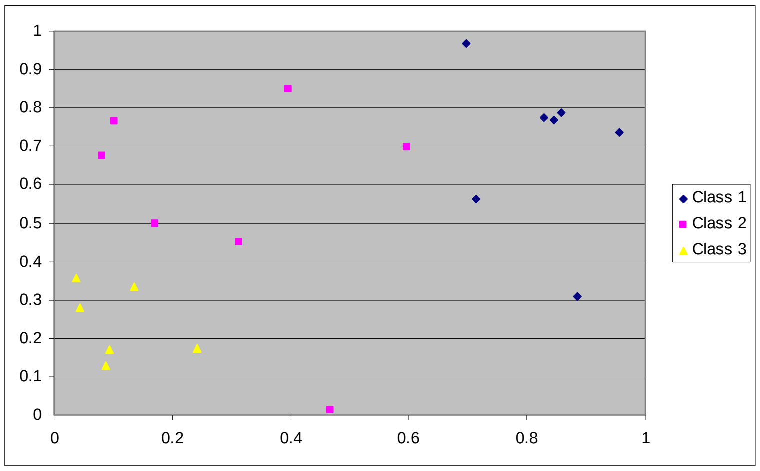
Step 5. Initialize alternatives
Set all best class indexes to 1
Set all worst class indexes to t
Set all class indexes to 0 (not placed)
Step 6. Select an alternative to place
Identify unplaced alternatives
Identify those, which are out of bounds
No alternatives out of bounds
YES
NO
Select the one, which is far from the nearest boundary
Select the one, which is closest to the nearest boundary
Step 7. Decide best and worst classes
Look for the class indexes dominating alternatives, which are placed
Set the smallest class index as the best class index of the selected alternative
Look for the class indexes dominated alternatives, which are placed
Set the largest class index as the worst class index of the selected alternative
Step 8. Convex Combination check
Solve LP1 to obtain E1
Solve LP2 to obtain E2
Check whether E1 = E2 = 0 OR E1 > 0 ; E2 > 0
Check whether E1 < 0 ; E2 > 0
Set worst class index to current class
Check whether E1 > 0 ; E2 < 0
Set best class index to current class
Step 9. Check for best class with WSR
Start from the worst possible class
Solve LP3
If "Infeasible" or E <= 0
YES
NO
Set best possible class index to the class
All classes except the best, are considered
YES
NO
stop this step
Consider the next worst class
Step 10. Check for worst class with WSR
Start from the best possible class
Solve LP4
If "Infeasible" or E <= 0
YES
NO
Set worst possible class index to the class
All classes except the worst, are considered
YES
NO
stop this step
Consider the next best class
check whether best class index = worst class index
NO
Go to next step
YES
Step 11. Find a dummy equivalent alternative
Start from the best possible class
Solve LP5
If "Infeasible"
NO
YES
NO
Perform an Even-Swap to come around dummy alternative
NO
All classes are considered?
Swapped point can be expresses as a convex combination?
NO
NO
Consider the next best class
Go to next step
Step 12. Ask DM to place the alternative
Ask DM to place the alternative to a class, in between best and worst classes
Step 13. Place the alternative
Set class index to current class
There exists unplaced alternatives
For each alternative in a better class, but not dominating, add a preference constraint to weight space
For each alternative in a worse class, but not dominated, add a preference constraint to weight space
Reestimate the utility function
Document Results "STOP"

APPENDIX B – Experimentation

<u>Experimentation with 20 alternatives:</u>

Alternative ID	Criterion 1 value	Criterion 1 value
1	0.1717	0.4979
2	0.8292	0.7750
3	0.0871	0.1286
4	0.7129	0.5627
5	0.044	0.2792
6	0.4681	0.0143
7	0.8454	0.7674
8	0.886	0.3096
9	0.2425	0.1729
10	0.6970	0.9681
11	0.3957	0.8499
12	0.0942	0.1718
13	0.5979	0.6971
14	0.1362	0.3332
15	0.8576	0.7886
16	0.0821	0.6767
17	0.0373	0.3582
18	0.313	0.4497
19	0.1013	0.7660
20	0.9571	0.7373

Class1			Class2			Class3		
Alt ID	Crit1	Crit2	Alt ID	Crit1	Crit2	Alt ID	Crit1	Crit2
2	0.8292	0.775	1	0.1717	0.4979	3	0.0871	0.1286
4	0.7129	0.5627	6	0.4681	0.0143	5	0.044	0.2792
7	0.8454	0.7674	11	0.3957	0.8499	9	0.2425	0.1729
8	0.886	0.3096	13	0.5979	0.6971	12	0.0942	0.1718
10	0.697	0.9681	16	0.0821	0.6767	14	0.1362	0.3332
15	0.8576	0.7886	18	0.313	0.4497	17	0.0373	0.3582
20	0.9571	0.7373	19	0.1013	0.766			



RUN 1:

Run Parameters	
# of Alternatives	20
Consistency Index	0.05
Underlying Utility Function (weight 1 / weight 2)	0.7 / 0.3
# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
	1		4	2
	6		9	3
	8		13	5
	10		16	7
	11			12
	14			15
	17			19
	18			20
	8		**4**	**8**

RUN 2:

Run Parameters	
# of Alternatives	20
Consistency Index	0.15
Underlying Utility Function (weight 1 / weight 2)	0.7 / 0.3
# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
	6	1	4	2
	8		9	3
	10		13	5
	11		16	7
	14			12
	17			15
	18			19
				20

	7	**1**	**4**	**8**

RUN 3:

Run Parameters	
# of Alternatives	20
Consistency Index	0.30
Underlying Utility Function (weight 1 / weight 2)	0.7 / 0.3
# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
18	8	1	4	2
	10	6	9	3
	11		13	5
	14		16	7
	17			12
				15
				19
				20
1	**5**	**2**	**4**	**8**

RUN 4:

Run Parameters	
# of Alternatives	20
Consistency Index	0.05
Underlying Utility Function (weight 1 / weight 2)	0.1 / 0.9
# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
	4		13	1
	6		14	2
	8		17	3
	9		19	5
	20			7
				10
				11
				12
				15
				16
				18
	5		**4**	**11**

RUN 5:

Run Parameters	
# of Alternatives	20
Consistency Index	0.15
Underlying Utility Function (weight 1 / weight 2)	0.1 / 0.9

# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
	4		13	1
	6		14	2
	8		17	3
	9		19	5
	20			7
				10
				11
				12
				15
				16
				18
	5		**4**	**11**

RUN 6:

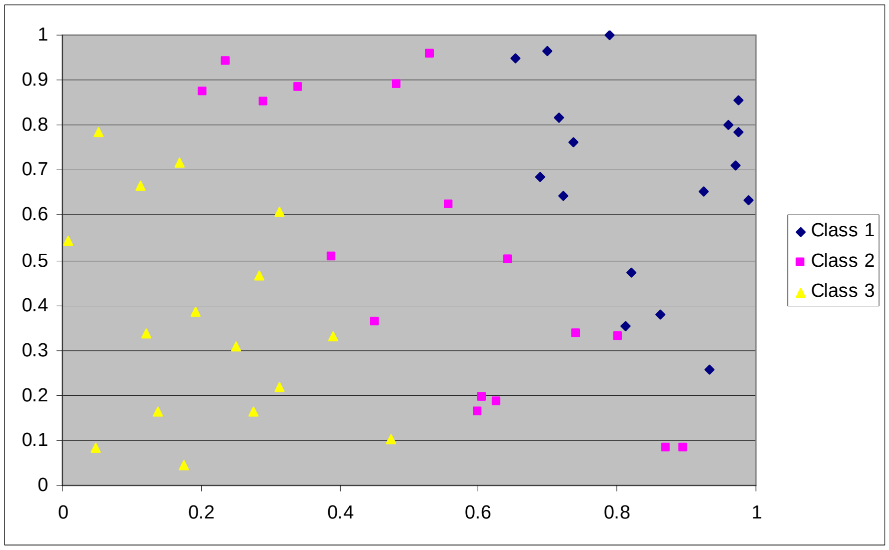
Run Parameters	
# of Alternatives	20
Consistency Index	0.30
Underlying Utility Function (weight 1 / weight 2)	0.1 / 0.9
# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
	4		13	1
	6		14	2
	8		17	3
	9		19	5
	20			7
				10
				11
				12
				15
				16
				18
	5		**4**	**11**

Experimentation with 50 alternatives:

Alternative ID	Criterion 1 value	Criterion 1 value	Alternative ID	Criterion 1 value	Criterion 1 value
1	0.3900	0.3296	26	0.3129	0.6086
2	0.1763	0.0446	27	0.1698	0.7177
3	0.1925	0.3864	28	0.1126	0.665
4	0.9750	0.7850	29	0.1379	0.1646
5	0.6266	0.1859	30	0.4732	0.1023
6	0.9751	0.8548	31	0.9704	0.7103
7	0.4504	0.3621	32	0.8698	0.0837
8	0.2843	0.4652	33	0.9333	0.2587
9	0.1218	0.3370	34	0.5299	0.958
10	0.6526	0.9496	35	0.6433	0.5016
11	0.2514	0.3081	36	0.9242	0.653
12	0.2035	0.8757	37	0.2897	0.8537
13	0.8618	0.3790	38	0.7000	0.9630
14	0.8026	0.3312	39	0.7228	0.6429
15	0.3876	0.5087	40	0.9595	0.8013
16	0.6889	0.6848	41	0.7171	0.8152
17	0.7401	0.3371	42	0.3404	0.8831
18	0.0472	0.0847	43	0.0084	0.5419
19	0.3137	0.2180	44	0.5986	0.1639
20	0.8961	0.0827	45	0.6055	0.1965
21	0.0523	0.7838	46	0.7369	0.7613
22	0.7899	0.9992	47	0.5571	0.6226
23	0.8131	0.3521	48	0.2759	0.1641
24	0.9899	0.6320	49	0.2366	0.9427
25	0.4817	0.8905	50	0.8195	0.472

Class1			Class2			Class3		
Alt ID	Crit1	Crit2	Alt ID	Crit1	Crit2	Alt ID	Crit1	Crit2
4	0.975	0.785	5	0.6266	0.1859	1	0.39	0.3296
6	0.9751	0.8548	7	0.4504	0.3621	2	0.1763	0.0446
10	0.6526	0.9496	12	0.2035	0.8757	3	0.1925	0.3864
13	0.8618	0.379	14	0.8026	0.3312	8	0.2843	0.4652
16	0.6889	0.6848	15	0.3876	0.5087	9	0.1218	0.337
22	0.7899	0.9992	17	0.7401	0.3371	11	0.2514	0.3081
23	0.8131	0.3521	20	0.8961	0.0827	18	0.0472	0.0847
24	0.9899	0.632	25	0.4817	0.8905	19	0.3137	0.218
31	0.9704	0.7103	32	0.8698	0.0837	21	0.0523	0.7838
33	0.9333	0.2587	34	0.5299	0.958	26	0.3129	0.6086
36	0.9242	0.653	35	0.6433	0.5016	27	0.1698	0.7177
38	0.7	0.963	37	0.2897	0.8537	28	0.1126	0.665
39	0.7228	0.6429	42	0.3404	0.8831	29	0.1379	0.1646
40	0.9595	0.8013	44	0.5986	0.1639	30	0.4732	0.1023
41	0.7171	0.8152	45	0.6055	0.1965	43	0.0084	0.5419
46	0.7369	0.7613	47	0.5571	0.6226	48	0.2759	0.1641
50	0.8195	0.472	49	0.2366	0.9427			



RUN 7:

Run Parameters	
# of Alternatives	50
Consistency Index	0.05
Underlying Utility Function (weight 1 / weight 2)	0.7 / 0.3
# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
17	1		12	2
35	5		14	3
37	7		23	4
46	10		26	6
47	15		34	8
	16			9
	20			11
	21			13
	22			18
	27			19
	30			24
	32			25
	33			28
	38			29
	39			31
	41			36
	44			40
	45			42
				43
				48
				49
				50

5	**18**		**5**	**22**

RUN 8:

Run Parameters	
# of Alternatives	50
Consistency Index	0.15
Underlying Utility Function (weight 1 / weight 2)	0.7 / 0.3
# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
17	1	7	14	2
35	5	12	15	3
46	10	16	20	4
47	22	34	23	6
	30		26	8
	32			9
	33			11
	37			13
	38			18
	39			19
	41			21
	42			24
	44			25
	45			27
	49			28
				29
				31
				36
				40
				43
				48
				50
4	**15**	**4**	**5**	**22**

RUN 9:

Run Parameters	
# of Alternatives	50
Consistency Index	0.30
Underlying Utility Function (weight 1 / weight 2)	0.7 / 0.3
# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
1	10	5	7	2
17	22	12	16	3
25	30	14	20	4
35	32	15	23	6
37	33	34	26	8
42	38	45	44	9

47	39		49	11
	41			13
	46			18
				19
				21
				24
				27
				28
				29
				31
				36
				40
				43
				48
				50
7	**9**	**6**	**7**	**21**

RUN 10:

Run Parameters	
# of Alternatives	50
Consistency Index	0.05
Underlying Utility Function (weight 1 / weight 2)	0.1 / 0.9
# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
8	1		3	2
15	10		7	4
17	12		16	5
25	14		36	6
26	21			9
37	23			11
	24			13
	27			18
	28			19
	33			20
	34			22
	42			29
	43			30
	49			31
				32
				35
				38
				39
				40
				41
				44
				45
				46

				47
				48
				50
6	**14**		**4**	**26**

RUN 11:

Run Parameters	
# of Alternatives	50
Consistency Index	0.05
Underlying Utility Function (weight 1 / weight 2)	0.1 / 0.9
# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
8	1		3	2
15	10		7	4
17	12		16	5
25	14		36	6
26	21			9
37	23			11
	24			13
	27			18
	28			19
	33			20
	34			22
	42			29
	43			30
	49			31
				32
				35
				38
				39
				40
				41
				44
				45
				46
				47
				48
				50
6	**14**		**4**	**26**

RUN 12:

Run Parameters	
# of Alternatives	50
Consistency Index	0.05
Underlying Utility Function (weight 1 / weight 2)	0.1 / 0.9

# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
8	9	3	1	2
15	10	27	7	4
17	12	33	14	5
25	21		16	6
26	24		36	11
37	28			13
	34			18
	42			19
	43			20
	49			22
				23
				29
				30
				31
				32
				35
				38
				39
				40
				41
				44
				45
				46
				47
				48
				50
6	**10**	**3**	**5**	**26**

Experimentation with 100 alternatives:

Alternative ID	Criterion 1 value	Criterion 1 value
1	0.832473245	0.955980422
2	0.947412859	0.895050087
3	0.753019334	0.49822847
4	0.614207422	0.313134137
5	0.3424168	0.501686154
6	0.08671645	0.27104183
7	0.853584041	0.113141229
8	0.260190032	0.37133937
9	0.537626481	0.135279453
10	0.241000145	0.790835886
11	0.073750323	0.010561168
12	0.513245474	0.932553728
13	0.795130693	0.821487671
14	0.189861305	0.041740431
15	0.839206625	0.144101518
16	0.059594292	0.432333708
17	0.486867394	0.160671516
18	0.735999703	0.166290046
19	0.464406083	0.345865531
20	0.143466196	0.956707613
21	0.192836693	0.036797936
22	0.060492734	0.603855139
23	0.310464731	0.672643931
24	0.285450622	0.24737219
25	0.621882788	0.653604652
26	0.722183838	0.262613704
27	0.778799285	0.340517335
28	0.799103113	0.432074287
29	0.202996158	0.612900643
30	0.836595514	0.508538558
31	0.415871906	0.568507938
32	0.48931535	0.527788795
33	0.777378849	0.625081627
34	0.848999086	0.137990519
35	0.231353816	0.416213454
36	0.337151451	0.843661416
37	0.375724187	0.810093271
38	0.198571633	0.137345423
39	0.383331991	0.080726606
40	0.914814546	0.027321941
41	0.604987791	0.540104749
42	0.796419794	0.580064143
43	0.850124341	0.021273596
44	0.282483332	0.139310251
45	0.480130608	0.377300004
46	0.574024729	0.674663997
47	0.708192047	0.086445481
48	0.365422299	0.19239134
49	0.023728097	0.686680671
50	0.244962971	0.102200394

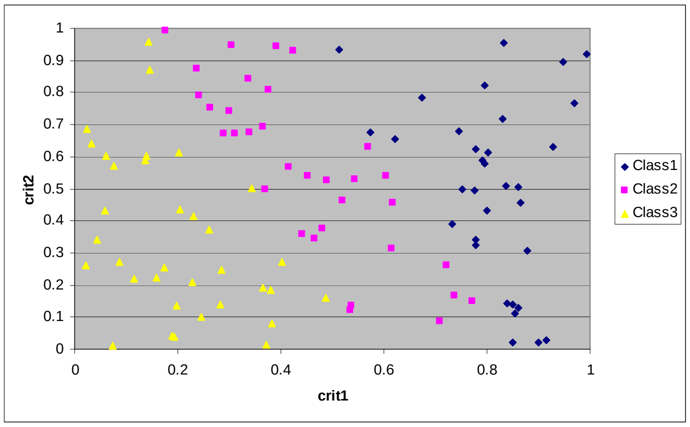
Alternative ID	Criterion 1 value	Criterion 1 value
51	0.992663346	0.921394289
52	0.15805215	0.224635798
53	0.441496201	0.359501055
54	0.14552186	0.871191621
55	0.381511131	0.184240931
56	0.032140635	0.641507591
57	0.864506978	0.457378111
58	0.534361588	0.121757557
59	0.17648324	0.992295065
60	0.453230764	0.541160228
61	0.390657796	0.945923677
62	0.861542781	0.127758741
63	0.422980653	0.929919058
64	0.304323174	0.94639595
65	0.96922068	0.765952414
66	0.927623558	0.629982704
67	0.402291509	0.271459831
68	0.116232643	0.218325976
69	0.732117643	0.39074074
70	0.33869598	0.676134621
71	0.544549909	0.529443227
72	0.235980527	0.873259861
73	0.776084535	0.49642373
74	0.137548599	0.588775218
75	0.076118645	0.572746004
76	0.772543958	0.149830007
77	0.365535945	0.691822652
78	0.13984392	0.601592939
79	0.618469837	0.455577213
80	0.203597813	0.435135888
81	0.745445192	0.680887682
82	0.228557863	0.209255101
83	0.878099092	0.305676716
84	0.022508035	0.262350081
85	0.262733829	0.751476624
86	0.520259	0.463249447
87	0.674965013	0.784149458
88	0.042721452	0.343153696
89	0.778707712	0.323776414
90	0.899764948	0.019489158
91	0.802880389	0.614349849
92	0.299628367	0.741196774
93	0.861517464	0.504841037
94	0.174336644	0.255147571
95	0.371507941	0.015617171
96	0.790817614	0.587568157
97	0.831112809	0.718197646
98	0.570222766	0.630412933
99	0.369114938	0.496737817
100	0.289137679	0.673427969

Placements

Class 1		
Alt ID	Crit1	Crit2
1	0.8325	0.956
2	0.9474	0.8951
3	0.753	0.4982
7	0.8536	0.1131
12	0.5132	0.9326
13	0.7951	0.8215
15	0.8392	0.1441
25	0.6219	0.6536
27	0.7788	0.3405
28	0.7991	0.4321
30	0.8366	0.5085
33	0.7774	0.6251
34	0.849	0.138
40	0.9148	0.0273
42	0.7964	0.5801
43	0.8501	0.0213
46	0.574	0.6747
51	0.9927	0.9214
57	0.8645	0.4574
62	0.8615	0.1278
65	0.9692	0.766
66	0.9276	0.63
69	0.7321	0.3907
73	0.7761	0.4964
81	0.7454	0.6809
83	0.8781	0.3057
87	0.675	0.7841
89	0.7787	0.3238
90	0.8998	0.0195
91	0.8029	0.6143
93	0.8615	0.5048
96	0.7908	0.5876
97	0.8311	0.7182

Class 2		
Alt ID	Crit1	Crit2
4	0.6142	0.3131
9	0.5376	0.1353
10	0.241	0.7908
18	0.736	0.1663
19	0.4644	0.3459
23	0.3105	0.6726
26	0.7222	0.2626
31	0.4159	0.5685
32	0.4893	0.5278
36	0.3372	0.8437
37	0.3757	0.8101
41	0.605	0.5401
45	0.4801	0.3773
47	0.7082	0.0864
53	0.4415	0.3595
58	0.5344	0.1218
59	0.1765	0.9923
60	0.4532	0.5412
61	0.3907	0.9459
63	0.423	0.9299
64	0.3043	0.9464
70	0.3387	0.6761
71	0.5445	0.5294
72	0.236	0.8733
76	0.7725	0.1498
77	0.3655	0.6918
79	0.6185	0.4556
85	0.2627	0.7515
86	0.5203	0.4632
92	0.2996	0.7412
98	0.5702	0.6304
99	0.3691	0.4967
100	0.2891	0.6734

Class 2		
Alt ID	Crit1	Crit2
5	0.3424	0.5017
6	0.0867	0.271
8	0.2602	0.3713
11	0.0738	0.0106
14	0.1899	0.0417
16	0.0596	0.4323
17	0.4869	0.1607
20	0.1435	0.9567
21	0.1928	0.0368
22	0.0605	0.6039
24	0.2855	0.2474
29	0.203	0.6129
35	0.2314	0.4162
38	0.1986	0.1373
39	0.3833	0.0807
44	0.2825	0.1393
48	0.3654	0.1924
49	0.0237	0.6867
50	0.245	0.1022
52	0.1581	0.2246
54	0.1455	0.8712
55	0.3815	0.1842
56	0.0321	0.6415
67	0.4023	0.2715
68	0.1162	0.2183
74	0.1375	0.5888
75	0.0761	0.5727
78	0.1398	0.6016
80	0.2036	0.4351
82	0.2286	0.2093
84	0.0225	0.2624
88	0.0427	0.3432
94	0.1743	0.2551
95	0.3715	0.0156



RUN 13:

Run Parameters	
# of Alternatives	100
Consistency Index	0.05
Underlying Utility Function (weight 1 / weight 2)	0.7 / 0.3
# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
3	9	5	10	1
4	12	20	17	2
7	18	26	46	6
15	19	43	76	8
23	40	58		11
25	45	98		13
27	47	99		14
28	57	100		16
29	59			21
30	61			22
31	62			24
32	63			35
33	64			36
34	66			37
41	72			38
42	83			39
53	90			44
54	93			48
60				49
67				50
69				51
70				52
71				55
73				56
77				65
79				68
85				74
86				75
89				78
91				80

92				81
96				82
				84
				87
				88
				94
				95
				97
32	**18**	**8**	**4**	**38**

* 291 LPs Executed

RUN 14:

Run Parameters	
# of Alternatives	100
Consistency Index	0.15
Underlying Utility Function (weight 1 / weight 2)	0.7 / 0.3
# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
3	1	17	5	2
4	9	20	10	6
8	12	46	43	7
15	13		67	11
23	18		76	14
26	19		100	16
27	25			21
28	33			22
29	42			24
30	47			31
34	58			32
35	59			36
41	61			37
45	63			38
53	64			39
54	69			40
73	72			44
79	81			48
80	87			49
85	89			50
86	90			51
96	91			52
97				55
99				56
				57
				60
				62
				65
				66
				68
				70
				71

				74
				75
				77
				78
				82
				83
				84
				88
				92
				93
				94
				95
				98
24	**22**	**3**	**6**	**45**

* 261 LPs Executed

RUN 15

Run Parameters	
# of Alternatives	100
Consistency Index	0.30
Underlying Utility Function (weight 1 / weight 2)	0.7 / 0.3
# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
3	2	7	5	1
4	18	9	10	6
8	47	15	12	11
13	51	17	26	14
19	57	20	59	16
23	61	28	67	21
25	65	34		22
27	66	40		24
29	93	43		31
30		46		32
33		58		36
35		63		37
41		69		38
42		76		39
45		83		44
53		90		48
54				49
62				50
64				52
70				55
72				56
73				60
79				68
80				71

81				74
85				75
86				77
87				78
89				82
91				84
92				88
96				94
97				95
99				98
100				
35	**9**	**16**	**6**	**34**

* 348 LPs Executed

RUN 16:

Run Parameters	
# of Alternatives	100
Consistency Index	0.05
Underlying Utility Function (weight 1 / weight 2)	0.1/ 0.9
# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
5	8		4	1
12	10		19	2
25	15		42	3
33	16		56	6
36	18		88	7
37	20		96	9
46	22			11
53	23			13
60	26			14
61	29			17
63	30			21
64	31			24
67	34			27
74	35			28
76	40			32
78	41			38
81	49			39
87	54			43
92	57			44
98	59			45
99	62			47
	70			48
	72			50
	75			51
	77			52

	80			55
	83			58
	85			65
	93			66
	100			68
				69
				71
				73
				79
				82
				84
				86
				89
				90
				91
				94
				95
				97
21	**30**	**0**	**6**	**43**

* 250 LPs Executed

RUN 17:

Run Parameters	
# of Alternatives	100
Consistency Index	0.15
Underlying Utility Function (weight 1 / weight 2)	0.1/ 0.9
# of Classes	3

Convex Combination	WSR	Even Swaps	DM Placement	Dominance
5	8	26	4	1
12	10		19	2
25	15		42	3
33	16		56	6
36	18		88	7
37	20		96	9
46	22			11
53	23			13
60	29			14
61	30			17
63	31			21
64	34			24
67	35			27
74	40			28
76	41			32
78	49			38
81	54			39
87	57			43

92	59			44
98	62			45
99	70			47
	72			48
	75			50
	77			51
	80			52
	83			55
	85			58
	93			65
	100			66
				68
				69
				71
				73
				79
				82
				84
				86
				89
				90
				91
				94
				95
				97
21	**29**	**1**	**6**	**43**

* 255 LPs Executed

RUN 18:

Run Parameters	
# of Alternatives	100
Consistency Index	0.30
Underlying Utility Function (weight 1 / weight 2)	0.1/ 0.9
# of Classes	3

Convex Combination	**WSR**	**Even Swaps**	**DM Placement**	**Dominance**
5	8	26	4	1
12	10	49	42	2
19	15		56	3
25	16		83	6
32	18		88	7
33	20		96	9
35	22			11
36	23			13
37	29			14
45	30			17
46	31			21

53	34			24
57	40			27
60	41			28
61	54			38
63	59			39
64	62			43
67	70			44
71	72			47
74	75			48
76	77			50
78	80			51
81	85			52
86	93			55
87	100			58
92				65
98				66
99				68
				69
				73
				79
				82
				84
				89
				90
				91
				94
				95
				97
28	**25**	**2**	**6**	**39**

* 257 LPs Executed