DECISION SUPPORT FOR MULTI-ATTRIBUTE AUCTIONS

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

BY

GÜLŞAH KARAKAYA

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
INDUSTRIAL ENGINEERING

DECEMBER 2013

Approval of the thesis:

DECISION SUPPORT FOR MULTI-ATTRIBUTE AUCTIONS

submitted by **GÜLŞAH KARAKAYA** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Industrial Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Murat Köksalan
Head of Department, **Industrial Engineering** _____

Prof. Dr. Murat Köksalan
Supervisor, **Industrial Engineering Dept., METU** _____

Examining Committee Members:

Assoc. Prof. Dr. Canan Sepil
Industrial Engineering Dept., METU _____

Prof. Dr. Murat Köksalan
Industrial Engineering Dept., METU _____

Assoc. Prof. Dr. Pelin Bayındır
Industrial Engineering Dept., METU _____

Assoc. Prof. Dr. Serhan Duran
Industrial Engineering Dept., METU _____

Assoc. Prof. Dr. Osman Alp
Industrial Engineering Dept., TEDU _____

Date: **26.12.2013**

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Gülşah KARAKAYA

Signature :

ABSTRACT

DECISION SUPPORT FOR MULTI-ATTRIBUTE AUCTIONS

Karakaya, Gülşah
Ph.D., Department of Industrial Engineering
Supervisor: Prof. Dr. Murat Köksalan

December 2013, 156 pages

In this study, we address multi-attribute, multi-item auction problems. In multi-attribute auctions there are additional attributes to the price and the comparison of bids is not straightforward. In multi-item auctions which are also known as combinatorial auctions, it is not so trivial to determine the winning bidders.

We develop an auction decision support system (ADSS) that supports sellers to bid on multiple items. We demonstrate our approach in a multi-attribute, multi-item reverse auction setting. The approach is also directly applicable to forward auctions. During the auction process, ADSS estimates the underlying preference function of the buyer and supports sellers providing them information based on these estimations. We first assume that the sellers do not share their cost functions with ADSS and develop interactive algorithms for underlying linear preference functions as well as for underlying quasiconvex preference functions. The aim of the developed approaches is to have the more competitive bidders eventually end

up winning the auction, with predetermined reasonable mark-up values. We demonstrate that our algorithms work well on a variety of test problems.

We also develop an interactive algorithm for the case that sellers explicitly make their cost functions available to ADSS. In this approach, ADSS tries to find the best possible combinations considering both the estimated preference function of the buyer and the cost functions of the sellers. The experiments show that our algorithm finds the optimal winners (achieved with exact parameters of the underlying preference function).

Keywords: multi-attribute auctions, multi-item auctions, interactive approach, combinatorial auctions

ÖZ

ÇOK ÖLÇÜTLÜ AÇIK ARTTIRMALAR İÇİN KARAR DESTEK

Karakaya, Gülşah
Doktora, Endüstri Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Murat Köksalan

Aralık 2013, 156 sayfa

Bu çalışmada çok ölçütlü, çok ürünlü ihale problemlerini ele alıyoruz. Çok ölçütlü ihalelerde fiyat dışında düşünülmesi gereken başka ölçütler de vardır ve tekliflerin değerlendirilmesi zordur. Kombinatoryal ihaleler olarak da bilinen çok ürünlü ihalelerde kazananları belirlemek özellikle zordur.

Satıcıların birden çok ürün için teklif vermelerini destekleyen bir ihale karar destek sistemi (İKDS) geliştirdik. Yaklaşımımızı çok ölçütlü, çok ürünlü açık eksiltme problemleri üzerinde gösterdik. Yaklaşımımız açık arttırma problemlerine de doğrudan uygulanabilir. İhalede, İKDS alıcının tercih fonksiyonunu tahmin edip satıcılara tahminler hakkında bilgi verir. İlk olarak satıcıların maliyet fonksiyonları bilgilerini İKDS ile paylaşmadığı durumu inceledik ve alıcının tercih fonksiyonunun hem doğrusal olduğu hem de doğrusal olmadığı durumlar için etkileşimli algoritmalar geliştirdik. Bu algoritmalar, maliyetleri daha rekabetçi olan satıcıların ihaleyi kazanmalarını hedeflemektedir.

Algoritmalarımızı test etmek için çözdüğümüz tüm problemlerde iyi sonuçlar elde ettik.

Ayrıca satıcıların maliyet fonksiyonları bilgisini İKDS'ye verdiği durum için de etkileşimli bir yaklaşım geliştirdik. Bu yaklaşımda, İKDS alıcının tahmin edilen tercih fonksiyonu ile satıcıların maliyet fonksiyonlarını göz önünde bulundurarak, en iyi teklifleri bulmaya çalışır. Yaptığımız testlerde algoritmamızın optimal kazananları (alıcının tercih fonksiyonunun açık olarak bilindiği durumda bulunan kazananlar) bulduğunu gördük.

Anahtar kelimeler: çok ölçütlü açık arttırmalar, çok ürünlü açık arttırmalar, etkileşimli yaklaşım, kombinatoryal açık arttırmalar

To my family...

ACKNOWLEDGMENTS

I would like to express my gratitude to my supervisor Professor Murat Köksalan not only for his invaluable guidance and supervision but also his understanding and encouragement throughout this study. I have learned a lot from him and it has been a pleasure for me to work with him.

I would like to thank to my families for their endless and unconditional love and trust throughout my life. I am thankful to my friends Diclehan Tezcaner Öztürk and Ceren Tuncer Şakar. I feel myself lucky to feel their existence whenever I need their ideas and support.

I owe thanks to Associate Professor Canan Sepil, Associate Professor Osman Alp, Associate Professor Pelin Bayındır and Associate Professor Serhan Duran for their valuable comments and suggestions.

I also would like to thank to TÜBİTAK for providing a financial support during my Ph.D. study.

Lastly, I am grateful to my husband Ahmet Fatih Karakaya for being in my life and for his support in my graduate studies. Nothing would be possible without the complimentary love and encouragement of him. Nothing would be meaningful without our daughter Zeynep.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

An auction is a way of buying and selling goods and services. The traditional auction process used to take place in a room or a square where an object was shown to the bidders by the auctioneer. The advances in the technological infrastructure and the Internet make it possible to conduct online auctions that eliminate the need for being present in the auction place physically. There are specialized websites that mediate between buyers and sellers and facilitate huge amounts of goods being traded between parties. With the online auction sites, people buy/sell various types of products/services.

Auctions are commonly used by companies and governments. Hohner et al. (2003) and Sandholm et al. (2006) report the implementation of auctions in Mars Inc. and Procter&Gamble, respectively. Metty et al. (2005) state that Motorola enjoy savings by implementing an online negotiation program. The government of Chile has used auctions for the procurement of school meals in Chile for many years (Catalán et al. 2009). Auctions have been commonly utilized in the transportation industry (Sheffi, 2004; De Vries et al. 2003).

Online auctions are becoming popular with the advances in the Internet and there is a growing amount of literature in this area. In single-attribute auctions generally the price is used as the attribute. In multi-attribute auctions, there are additional attributes and the comparison of bids is not straightforward. Multi-item auctions also bring additional complexity over single-item auctions. In the single-

item auctions, the winning bidder supplies the item with the committed attribute values. On the other hand, in multi-item case it is not trivial to determine the winning bidders. In these auctions, generally bidders offer a combination of items – a bundle – they wish to supply. They specify the attribute values of the bundle and make their bids. Multi-item auctions are known as combinatorial auctions (CAs).

In this thesis, we study multi-attribute multi-item (MAMI) auction problems. We first develop an exact approach that provides aid both to the buyer and the sellers for MAMI multi-round auctions where the buyer has an underlying linear preference function. The approach estimates the parameters of a preference function representing the buyer's preferences evaluated on multiple attributes and informs the sellers about the estimations to update their bids for the next round. We set different weight values to the attributes and generate different problems. We test the performance of the algorithm for both two and three attribute cases when the underlying preference function is linear. We also make a local linear approximation for nonlinear preference functions and report the results. We then develop an interactive method to support the buyer to find the best bid combination among the given bids for two attribute problems. This method decreases the number of comparisons made by the buyer. We use this method as an exact method for underlying linear preference functions, and as a heuristic for underlying nonlinear preference functions. We also develop an interactive method for underlying quasiconvex preference functions. We try different versions for this method and report the results for two attribute problems. Furthermore, in all of the mentioned methods above we assume that we do not know the cost functions of the sellers. We then address the case where sellers explicitly make their cost functions available to us (the independent party mediating the auction). By using their cost functions, we find favorable combinations to present the buyer. We refer to this case as "*Coordinated Bidding*" case. We also test the performance of the approach for this case. Lastly, we made modifications to improve the Evolutionary Algorithm (EA) developed in Karakaya (2009) for

MAMI reverse auctions in order to overcome the computational difficulties. We approximately generate the whole Pareto front using the EA. We test the EA on a number of randomly generated problems and report our findings.

The structure of the thesis is as follows: In Chapter 2, we give some definitions on multi-objective decision making, we present the background of the auction theory and relevant literature, and explain the general structure of our approach and define the problem specifications. In Chapter 3, we develop an approach that finds a set of efficient bid combinations to present the buyer. We develop an interactive method to support the buyer to find the best bid combination in Chapter 4. In Chapter 5, we develop an interactive method to find the most preferred bid combination of a buyer having a quasiconvex preference function. In Chapter 6, we describe the "*Coordinated Bidding*" case where we create good combinations to present the buyer by using the cost functions of sellers. We discuss extension we made to our previous work in Chapter 7. Lastly, we present future study issues and conclusive remarks in Chapter 8.

CHAPTER 2

DEFINITIONS, BACKGROUND AND PROBLEM CHARACTERISTICS

In this chapter we first give some definitions on multi-objective decision making. We then explain the auction process and summarize relevant literature. Lastly, we describe our approach and give the problem characteristics.

2.1 Definitions

In multi-objective optimization problems there are two or more, generally conflicting, objectives subject to a set of constraints. The general multi-objective optimization problem can be formulated as follows:

"Minimize" $\{z_1(\boldsymbol{x}), z_2(\boldsymbol{x}), \dots, z_J(\boldsymbol{x})\}$

subject to

$\boldsymbol{x} \in \boldsymbol{X}$

where,

$\boldsymbol{x}$: decision variable vector

$\boldsymbol{X}$: feasible decision space

z_j: j^{th} objective function

and the quotation marks are used to indicate that the minimization of a vector is not a well-defined mathematical operation.

A solution $\boldsymbol{x} \in \boldsymbol{X}$ is said to be *efficient*, if and only if there does not exist $\boldsymbol{x}' \in \boldsymbol{X}$ such that $z_j(\boldsymbol{x}') \leq z_j(\boldsymbol{x})$ for all j and $z_j(\boldsymbol{x}') < z_j(\boldsymbol{x})$ for at least one j. Otherwise,

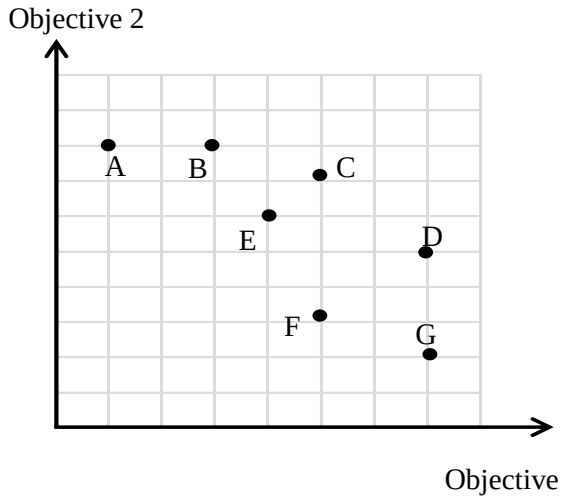
$\boldsymbol{x}$ is said to be *inefficient*. If $\boldsymbol{x}$ is efficient, then $\boldsymbol{z}(\boldsymbol{x}) = (z_1(\boldsymbol{x}), z_2(\boldsymbol{x}), \dots, z_J(\boldsymbol{x}))$ is said to be *nondominated*, whereas if $\boldsymbol{x}$ is inefficient, then $\boldsymbol{z}(\boldsymbol{x})$ is said to be *dominated*. $\boldsymbol{z}(\boldsymbol{x})$ is said to be *strictly dominated*, if and only if $z_j(\boldsymbol{x}') < z_j(\boldsymbol{x})$ for all *j* whereas $\boldsymbol{z}(\boldsymbol{x})$ is said to be *weakly nondominated*, if and only if there does not exist $\boldsymbol{x}' \in \boldsymbol{X}$ such that $\boldsymbol{z}(\boldsymbol{x}')$ strictly dominates $\boldsymbol{z}(\boldsymbol{x})$.

Consider distinct solutions $\boldsymbol{x}_i \in \boldsymbol{X}$, $i = 1,2, \dots, n$. Let $\boldsymbol{Y} = \{\boldsymbol{y}: \boldsymbol{y} = \sum_{i=1}^{n} \mu_i \boldsymbol{x}_i, \sum_{i=1}^{n} \mu_i = 1, \mu_i \geq 0\}$ be the set of all convex combinations of $\boldsymbol{x}_i$. A solution $\boldsymbol{x}_i$ is said to be convex dominated, if there exists $\boldsymbol{y} \in \boldsymbol{Y}, \boldsymbol{y} \neq \boldsymbol{x}_i$ such that $z_j(\boldsymbol{y}) \leq z_j(\boldsymbol{x}_i)$ for all *j*.

An efficient solution, $\boldsymbol{x}$, is said to be *unsupported efficient* if and only if there exists $\boldsymbol{y} \in \boldsymbol{Y}, \boldsymbol{y} \neq \boldsymbol{x}$ such that $z_j(\boldsymbol{y}) \leq z_j(\boldsymbol{x})$ for all *j* and $z_j(\boldsymbol{y}) < z_j(\boldsymbol{x})$ for at least one *j*. An efficient solution, $\boldsymbol{x}$, is said to be *nonextreme supported efficient* if and only if there exists $\boldsymbol{y} \in \boldsymbol{Y}, \boldsymbol{y} \neq \boldsymbol{x}$ such that $z_j(\boldsymbol{y}) = z_j(\boldsymbol{x})$ for all *j*. An efficient solution, $\boldsymbol{x}$, is said to be *extreme supported efficient* if and only if there does not exist $\boldsymbol{y} \in \boldsymbol{Y}, \boldsymbol{y} \neq \boldsymbol{x}$ such that $\boldsymbol{y}$ convex dominates $\boldsymbol{x}$.

It is well-known in the multi-objective literature that any supported nondominated solution can be found by using a suitable weighted linear combination of the objective functions. However, finding unsupported nondominated solutions is not straightforward.

In Figure 2.1 the classification of the solutions based on the domination rules where both objectives to be minimized are represented.



Dominated Solutions: B, C, D

Strictly Dominated Solutions: C, D

Weakly Nondominated but Dominated Solutions: B

Nondominated Solutions: A, E, F, G

Supported Nondominated Solutions: A, F, G

Unsupported Nondominated Solutions: E

Figure 2.1 Classification of the solutions

A pair of solutions are *adjacent efficient* to each other if their convex combinations are not dominated by the convex combinations of other solutions. In bi-objective problems, an extreme supported solution can have at most two distinct adjacent efficient solutions (see Ramesh et al., 1990). We specify these adjacent efficient solutions as *east* and *west* based on their positions relative to the reference solution. An adjacent efficient solution having a larger value than the reference solution in objective 1 is called its *east adjacent efficient solution*; whereas an adjacent solution having a larger value than the reference solution in objective 2 is called its *west adjacent efficient solution*. To demonstrate, consider alternative *F* in Figure 2.1. It has two adjacent efficient alternatives: *A* and *G*. We refer to *G* as the east adjacent efficient alternative of *F* and *A* as the west adjacent efficient alternative of *F*.

Let $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \boldsymbol{X}$. We will use the notation $\boldsymbol{x}_1 \succ \boldsymbol{x}_2$ to imply that the decision maker (DM) prefers $\boldsymbol{x}_1$ to $\boldsymbol{x}_2$ and $\boldsymbol{x}_1 \sim \boldsymbol{x}_2$ to imply that the DM is indifferent between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$.

Let $f: R^J \to R^1$ be a quasiconvex function. By definition $f(\sum_{i=1}^{n} \mu_i \boldsymbol{x}_i) \leq max_i f(\boldsymbol{x}_i)$ for $\sum_{i=1}^{n} \mu_i = 1, \mu_i \geq 0$ where $\boldsymbol{x}_i \in R^J$, $i = 1, \ldots, n$ are distinct

alternatives. The weighted L_α metric is a quasiconvex function that measures the weighted distance between two vectors $\boldsymbol{p}, \boldsymbol{q} \in R^J$ as follows:

$\|\boldsymbol{p}, \boldsymbol{q}\|_\alpha^w = \left(\sum_{j=1}^{J}\left(w_j|p_j - q_j|\right)^\alpha\right)^{1/\alpha}$ where $\alpha = \{1,2, \dots\} \cup \{\infty\}$ and $w_j > 0$.

Commonly used weighted L_α metrics are the weighted rectilinear distance, the weighted Euclidean distance, and the weighted Tchebycheff distance, corresponding to $\alpha = 1,\ \alpha = 2, \text{and}\ \ \alpha = \infty$, respectively.

Within the context of an auction, from a buyer's perspective, $z_j(\boldsymbol{x})$ refers to the value of attribute *j* of bid $\boldsymbol{x}$. The buyer's problem, then, is to choose the bid that minimizes his/her underlying preference function. In this thesis we use the terms bid combination, solution and alternative interchangeably to refer to a combination of bids that satisfy the whole requirements of the buyer. We also use the terms buyer and DM interchangeably.

2.2 Auction Process

In the literature, auctions are categorized with respect to different properties. For instance, they can be classified based on the number of buyers and sellers. If there is one buyer and one seller, it is called negotiation. If there are many buyers and many sellers it is called a double auction as in the case of a stock market. In forward auctions there are one seller and many buyers. The auction for art objects is an example of the forward auction. The last type is reverse auctions where the buyer is the auctioneer and the sellers are the bidders. It is a common auction type for procurement processes in the literature. To exemplify, a manufacturer selects the suppliers of some products where the manufacturer is the auctioneer who will buy the products and suppliers sell their products. Karakaya and Köksalan (2011) represent the classification of the auctions with respect to the number of sellers and buyers as follows:

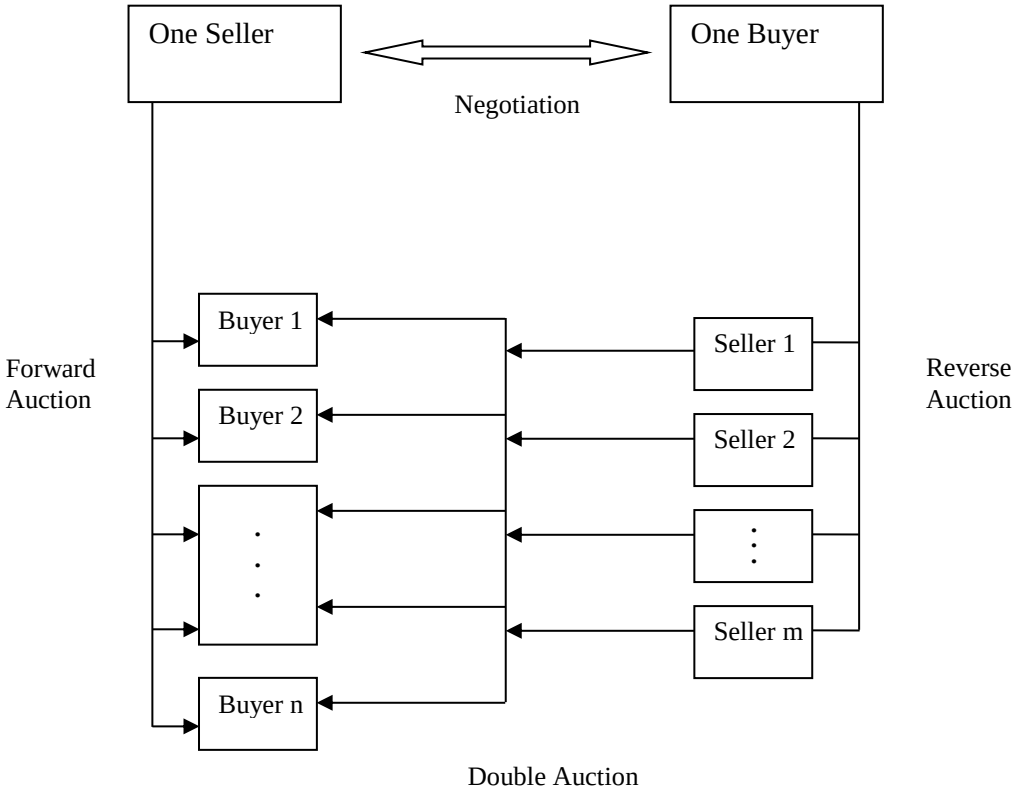


Figure 2.2 Auction types with respect to the number of sellers and buyers

McAfee and McMillan (1987) group the auction mechanisms into four: the English auction, the Dutch auction, the first-price sealed-bid auction and the second-price sealed-bid auction (Vickrey auction). In an English auction, bidders increase their bids during the action and the one who bids the highest price is the winner. In these auctions, all bidders know each others' bids. This is a property of open-cry auctions. In Dutch auction, the auction starts with a relatively high price and the auctioneer decreases it until a bidder accepts the current price. In the first-price sealed-bid auction, bidders do not know each other's bids. It is not an iterative process. The bidder offering the highest (lowest) price for the forward (reverse) process wins and he/she pays the highest (lowest) price. The second-price sealed-bid auction is similar to the first-price sealed-bid auction except that the winner pays the second highest (lowest) price.

Auctions are also classified with respect to the number of different items and the number of units for each item auctioned. In single-item, single-unit auctions, there

exists one unit of an item to be auctioned. If there are two or more units for the item auctioned, it is called a single-item, multi-unit auction. In single-item auctions, the bidder who values the item most is the winner. However, in the multi-item case where the items are complements or substitutes, it is not trivial to determine the winning bidder(s). These auctions are also known as CAs where bidders compose combinations of items, bundles, to sell/buy.

The number of attributes in the auction process is another classification. Price is a typical attribute in auctions and if only price is considered, it is a single-attribute auction. On the other hand, if there are additional attributes (quality, lead time, warranty, etc.) to the price, it is a multi-attribute auction. Multi-attribute auctions bring additional complexity over single-attribute auctions as the comparison of bids is not straightforward in multi-attribute auctions.

If the bidders submit their bids at different rounds during an auction, it is called a multi-round (iterative, progressive) auction. Multi-round combinatorial auctions have important advantages over single-round versions. Bidders do not have to submit bids for every possible combination in advance. It also allows bidders to behave in a dynamic manner. Moreover, additional information can be collected and utilized in a multi-round setting (see De Vries and Vohra, 2003). An application for multi-round combinatorial auction in Mars Inc. is reported by Hohner et al. (2003).

In this thesis, we deal with MAMI auction problems and the relevant literature is summarized next.

2.3 Literature Review

In single-attribute auctions there is one attribute, typically the price. Choosing the winner of such auctions is simple (Rothkopf and Park 2001). On the other hand, in multi-attribute auctions comparison of the bids is not so simple. To evaluate bids of multi-attribute auctions, typically a value or a scoring function is applied.

Commonly, such value functions are in the form of weighted linear functions. The winner of the auction is determined by solving the Winner Determination Problem (WDP) that maximizes the value/scoring function.

To evaluate bids of multi-attribute auctions, Bichler and Kalagnanam (2005) suggest a weighted-sum scoring function. Although using such functions is very common, as Bellosta et al. (2004) state, it has some drawbacks such as the difficulty of determining weights. Also, the solutions that can be found are limited with a weighted-sum scoring function. Bichler and Kalagnanam (2005) study multi-sourcing, i.e. demand can be supplied by multiple suppliers. They limit the number of winners by setting a lower and an upper bound on the number of winners.

Another approach to multi-attribute auctions is using the 'pricing out' technique as in Teich et al. (2006). In this technique, all attributes are converted into monetary values (see Keeney and Raiffa 1993, pp.125-127). Teich et al. (2006) solve the resulting problem with a single attribute, namely the price. They propose 'suggested price' tool for bidders. Bidders make the combination by deciding on the quantities. Then the best price that makes the bidder's bid among the provisional winners is determined by using the 'suggested price' tool. Leskelä et al. (2007) formulate a single-attribute auction problem and argue that the formulation can be extended to the multi-attribute case by the pricing out approach. They develop a Quantity Support Mechanism (QSM) that provides bidders not only the suggested price for a new bid, but also quantity decision support. They refer to a bid as "active" if it is among the provisional winners and "inactive" otherwise. An inactive bid can become active if an entering bid groups with it. Köksalan et al. (2009) improve the QSM and develop a Group Support Mechanism (GSM). In QSM only one incoming bid can complement the existing bids; whereas in the GSM a group of inactive bids can make a combination with active bids or with inactive bids. Sandholm and Suri (2006) propose a weighting function to evaluate bids in multi-attribute auctions. In the weighting

function$f(p_j, \vec{a}_j)$, p_j refers to the price and $\vec{a}_j$ refers to the vector of nonprice attributes of bid j. This function is introduced to represent all other attributes in terms of price, although the details are not explained. This approach is similar to the pricing out technique in Teich et al. (2006).

Talluri et al. (2007) use data envelopment analysis (DEA) to propose a decision support system tool for a multi-sourcing, single-round auction. They try to reflect the correlation between the attributes in the value function. They define weights for each attribute. To reflect the decision maker's (DM's) preference information for attributes, they define ranges instead of exact weights. They divide the DEA model into two stages. In stage I, scores of each bid are evaluated whereas in stage II, the winning bids are determined.

In the above approaches, simple functions that combine multiple attributes are used to estimate value functions. However, determining the weights and converting all attributes into a composite value are not easy.

Bellosta et al. (2004) suggest a multi-criteria model based on reference points for single-item auctions. The DM defines an aspiration point at the beginning of the auction. He/she also defines a dynamic reservation point based on which sellers update their bids. Bids are evaluated using the scaled deviations from the aspiration levels. Tchebycheff method is applied; the maximum scaled deviation among all attributes is the deviation of that bid. Baykal (2007) studies combinatorial auctions and applies a variation of Korhonen and Laakso's (1986) approach to the multi-attribute, multi-item auctions. She tries to find the best combination of bids for a single round. Determination of aspiration and/or reference points is not an easy task for the DM. Therefore, these methods may not well represent the preferences of the DM.

Karakaya and Köksalan (2011) propose an interactive method for multi-attribute, single-item reverse auctions. They estimate the underlying preference function of

the buyer considering his/her past preferences. At each round, they only require the buyer to select the most preferred bid among a set of bids. Then they inform the sellers about the estimations and facilitate the sellers to update their bids accordingly. They test the performance of the algorithm on a number of test problems and conclude that the suggested mechanism supports the sellers well. The buyer also benefits with the improvement in his/her preference function value over the progress of the auction.

Sandholm et al. (2002) study the complexity of winner determination in combinatorial auctions. They consider a single-attribute, price, and experiment on different types of combinatorial auctions using a general purpose mixed integer program solver, CPLEX. Sandholm (2002) proposes a tree search algorithm that branches on items to find the optimal solution for combinatorial auctions. The algorithm is a depth-first algorithm and allows finding feasible solutions quickly. Also several preprocessing methods are suggested to speed up the algorithm. Sandholm and Suri (2003) improve the algorithm in Sandholm (2002). They suggest to branch on bids (BOB) instead of branching on items as in Sandholm (2002). Besides computational advantages to the proposed algorithm in Sandholm (2002), BOB can also be used for multi-unit combinatorial auctions. Sandholm et al. (2005) suggest CABOB which is mainly based on the BOB algorithm. They compare CABOB and CPLEX and report results. They claim that CABOB is often drastically faster and seldom drastically slower than CPLEX.

Catalan et al. (2009) report the multi-attribute combinatorial auction for the procurement of school meals in Chile. The Chilean government is the auction owner and sets several criteria for the supply of foods. The bid selection is based on the fulfillment of those criteria. After applying a single-round auction, combination of bids supplying the whole demand at minimum cost is selected.

2.4 The Approach

Karakaya and Köksalan (2011) provide aid to both the buyer and the sellers in a multi-attribute, single-item, multi-round reverse auction environment. We extend their approach to multi-item auction problems. We develop an approach that supports sellers to bid on single items or bundles of items. The approach estimates the parameters of a preference function representing the buyer's preferences evaluated on multiple attributes and informs the sellers about the estimations to update their bids for the next round.

We present the approach for MAMI multi-round reverse auctions; however it is directly applicable to forward auctions. We consider an environment where each seller bids independently. We assume that no gaming issues are applicable and each seller bids based on his/her underlying cost function. We consider revenue maximization and allocative efficiency which are desirable properties of an auction mechanism. As stated in Ervasti and Leskelä (2010), in revenue maximization the buyer wants to maximize his/her revenue whereas in allocative efficiency the winners are the ones who have the lowest production cost. Since we consider multiple attributes in a reverse auction setting, we can consider preference function value minimization. We design our approach to support both the buyer and the sellers. We will refer to our approach as an auction decision support system (ADSS). This system is intended to act as a neutral third party independent from both the buyer and the sellers. During the auction process, we try to estimate the underlying preference function of the buyer and support sellers providing them information based on our estimations. At each round, the preference function value of the buyer is improved while the sellers update their bids to maximize their profits. Although sellers' profit may decrease as rounds progress, we expect sellers to update their bids in order to be among the winners. At the end of the auction, we expect the buyer to end up with a highly preferred solution and the competitive bidders to be the winners of the auction.

We consider two variations of the problem: single-round and multi-round. In the single-round case sellers compose their bids and the buyer selects the most preferred combination by evaluating the given bids. There would be no update of bids. On the other hand, in the multi-round case sellers compose their bids and the buyer selects the most preferred combination at each round. Then the sellers update their bids and the auction continues. In the single-round case, the buyer uses ADSS once to determine the most preferred combination. In the multi-round case we try to estimate the underlying preference function of the buyer based on his/her past preferences, without requiring any extra information. Then, we use this estimation to guide the sellers in updating their bids.

We summarize the stages of our approach for problems where *price* and *defect rate* are the two attributes in Figure 2.3. The process is similar for problems with more than two attributes.

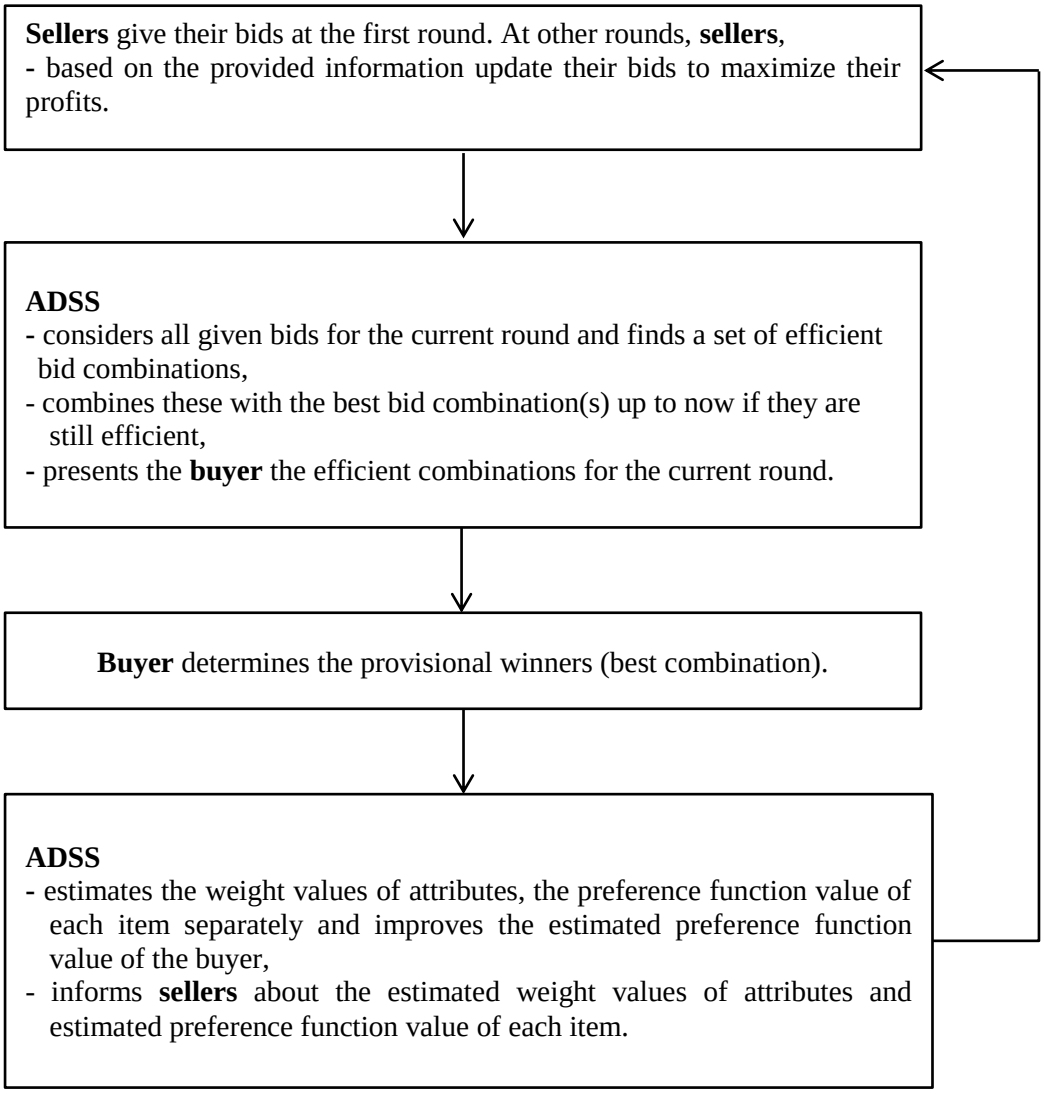


Figure 2.3 The stages of the approach

In the next chapters we explain the models that we solve in each stage in detail.

2.5 Problem Characteristics

We consider an environment where there are I sellers, M items, J attributes. We assume that all units of an item are supplied by a single seller. We use T_{ih} to represent the number of bids offered by seller *i* at round *h*. First M bids of each seller represents these singletons and the remaining T_{ih}-M bids represent seller *i*'s bundled bids at round *h*. We represent the bid of seller *i* as $\boldsymbol{b}_{it} = (a_{it1}, a_{it2}, \dots, a_{itj}, \dots, a_{itJ})$ where a_{itj} stands for the level of attribute *j*

offered by seller *i* for bid *t*. We use $\prod_{it} = (\pi_{it1}, \pi_{it2}, \dots, \pi_{itM})$ to represent the items that bid *t* of seller *i* consists of where π_{itm} takes the value of 1 if bid *t* of seller *i* includes item *m*; otherwise it is 0. We omit subscripts indicating rounds for simplicity. The preference function value of the buyer evaluated for bid *t* of seller *i* is depicted as $u(\boldsymbol{b}_{it})$. We use a weighted L_α metric to represent the underlying preference of the buyer for the bids. This function minimizes the weighted distance of a point from the ideal point in terms of an L_α metric. We estimate the weight values based on the past preferences of the buyer fitting the following preference function as an estimate of the preference to $\boldsymbol{b}_{it}$ at any round:

$$u(\boldsymbol{b}_{it}) = \left(\sum_{j=1}^{J}\left(w_j\left(a_{itj} - z_j^*\right)\right)^\alpha\right)^{1/\alpha}$$

where

w_j : weight of attribute j

z_j^*: ideal (best attainable) level of attribute j

α : parameter of the L_α metric

The preference function measures the weighted distance from the ideal point. Therefore, smaller $u(\boldsymbol{b}_{it})$ values are preferred by the buyer. If an attribute is of maximization type, we would simply replace $\left(a_{itj} - z_j^*\right)$ with $\left(z_j^* - a_{itj}\right)$ in the distance function. z_j^* values are typically the best attainable values for each attribute and can usually be extracted from the problem context. For simplicity we assume that $z_j^* = 0, \forall j$ and $u(\boldsymbol{b}_{it}) = \left(\sum_{j=1}^{J}\left(w_j a_{itj}\right)^\alpha\right)^{1/\alpha}$.

We note that the weights capture the relative importances of the attributes to the buyer and the scaling of attributes.

When we estimate the underlying preference function of the buyer with a linear preference function, we set α = 1 and estimate the weight value of each attribute. Otherwise, we estimate both α and the weight values of each attribute.

In the approach, sellers give their bids for single items as well as for bundles. We find some efficient combinations of bids that satisfy all the auctioned items and present these combinations to the buyer. We estimate the parameter values of the preference function of the buyer based on his/her preferences. We use a small positive constant threshold, "Δ" to represent a minimum preference difference by which the buyer can distinguish between bids as suggested by Karakaya and Köksalan (2011). For instance, if the buyer prefers *A* to *B*, then we require $u(B) \geq u(A) + \Delta$.

Moreover, at each round, we expect the sellers to improve their bids in such a way that the resulting combinations of the next round have improved preference function values of approximately "100γ" percent of the estimated value of the best combination of the current round as in Karakaya and Köksalan (2011). Therefore, after estimating a preference function based on the past preferences of the buyer, we provide information to the sellers about a possible way of improving their bids. According to these information and their cost functions, sellers update their bids for the next round. The auction continues until a termination condition is met. The possible termination conditions will be discussed later.

To demonstrate a simplified version of the approach, consider the following example. For the sake of simplicity, suppose that there are two sellers, one buyer, and one item to be auctioned with two attributes. Suppose each seller has two equally desirable bids as follows ($\boldsymbol{b}_{X1}$, $\boldsymbol{b}_{X2}$, for seller *X* and $\boldsymbol{b}_{Y1}$, $\boldsymbol{b}_{Y2}$ for seller *Y*):

$\boldsymbol{b}_{X1} = (2,5)$
$\boldsymbol{b}_{X2} = (4,1)$
$\boldsymbol{b}_{Y1} = (3.2,3.2)$
$\boldsymbol{b}_{Y2} = (7,0.5)$
where $i = X,Y$, $t = 1,2$ and $j = 1,2$.

Since we there is a single item to be auctioned, there is no item information in Table 2.1.

Table 2.1 Bids of sellers

	Seller *X*		Seller *Y*	
	$\boldsymbol{b}_{X1}$	$\boldsymbol{b}_{X2}$	$\boldsymbol{b}_{Y1}$	$\boldsymbol{b}_{Y2}$
Attribute 1	2.0	4.0	3.2	7.0
Attribute 2	5.0	1.0	3.2	0.5

Assume at the beginning the sellers have no information about the preference function of the buyer and seller *X* gives bid $\boldsymbol{b}_{X2}$ and seller *Y* gives bid $\boldsymbol{b}_{Y1}$ by considering a linear preference function with equal weights for the buyer. Suppose the buyer has a linear preference function with weights 0.9 and 0.1 for attributes 1 and 2, respectively. Then he/she prefers $\boldsymbol{b}_{Y1}$. We set Δ to zero and estimate the weights of attributes as 0.8 and 0.2 for attributes 1 and 2, respectively. Then seller *X* updates his/her bid and gives bid $\boldsymbol{b}_{X1}$ while seller *Y* keeps bidding on $\boldsymbol{b}_{Y1}$. Based on the underlying preference function of the buyer, he/she chooses $\boldsymbol{b}_{X1}$ and suppose the auction ends. The bids given in each round and the corresponding preference function values are given in Table 2.2.

Table 2.2 Buyer's preference function for the bids in each round

	Bid	Buyer's preference function value	Winner
Round 1	$\boldsymbol{b}_{X2}$ $\boldsymbol{b}_{Y1}$	3.7 **3.2**	$\boldsymbol{b}_{Y1}$
Round 2	$\boldsymbol{b}_{X1}$ $\boldsymbol{b}_{Y1}$	**2.3** 3.2	$\boldsymbol{b}_{X1}$

If there is no information, the auction may end at the first round and the winner would be seller *Y* with bid $\boldsymbol{b}_{Y1}$. The preference function value of the buyer would be 3.2. On the other hand, if the auction continues and ends at the second round, the winner would be seller *X* and the preference function value of the buyer would be 2.3. The estimated weights lead the sellers to converge the preferred bids. Normally, the situation is more complex with many possible potential bids for each seller. The process would continue for multiple iterations and more

preference information would be collected to guide the sellers. In such cases the estimated weights would help sellers update their bids and guide them to better bids in complex environments. The above example demonstrates that our approach can be useful even when implemented without its full potential. It does not use the improvement requirement that may be imposed on the sellers over the rounds. This requirement would lead to further improvements from the buyer's perspective and help reach allocative efficiency.

2.6 Auction Design

In our experiments we consider an environment where sellers have underlying cost functions based on which they bid and the buyer has an underlying preference function with which his/her preferences are consistent. As stated before, we assume that there are no strategic bidding or gaming and sellers always bid independently with their true valuations. Our mechanism aims to achieve allocative efficiency. It tries to improve the buyer's preference function value in each round and eventually to converge the most preferred combination. Therefore, neither the buyer nor the sellers can gain by acting against their true valuations. We also assume that there is no collaboration between sellers; i.e. sellers do not collude.

We assume that the attribute values of bid combinations are additive. For example, to represent the total price of a bid combination, we sum up the offered prices of singletons and bundle bids in the combination. When we consider defect rate as an attribute, we sum up the individual defect rates of singletons and bundle bids to determine the overall defect rate of the combination. This may not be realistic, especially in manufacturing environments. An alternative strategy could be to consider the maximum of the offered defect rates as the defect rate of the combination. In some situations weighted average could also be a viable option. Another alternative could be to penalize the larger defect rates more than proportionately. We should note that any implementation other than an additive aggregation of attribute values in a combination brings difficulties in the

disaggregation when trying to assign the buyer's preference information to the components of a combination. In 3-attribute problems, in addition to price and defect rate, we consider lead time as the third attribute and we calculate the lead time value for the combination by summing up the lead time values of singletons and bundle bids in the combination.

2.7 Implementation Issues

Our approach can be implemented in various settings including the procurement processes of companies in the automotive, food, and medical supplies industries. We consider a platform where the sellers place their bids until a certain deadline. The bids can be for single items or multiple items. ADSS would identify bid combinations that are expected to be desirable to the buyer. The buyer would select the best combination among the presented combinations and ADSS would provide updated preference information to the sellers to help them update their bids for the next round.

We suppose that all participants of the auction would use ADSS and relevant information (bids, estimated preference function values, etc.) is transferred via ADSS. As an alternative, we consider the case where sellers share their cost function information with ADSS. ADSS keeps creating desirable combinations using the sellers' cost functions and the buyer's estimated preference function throughout the auction process.

An example environment where our approach would be applicable would be the super market chains. The super market chains sell different varieties of products under different brands. They also sell some products with their own brands. They contract the production of these items to different firms. Dairy products (different cheese types, butter, yogurt) or oils (different types of vegetable and olive oils) produced by different firms are examples of these cases. Typically, these auctions are not conducted frequently and the supplying firms have the flexibility to offer different versions of their products. Therefore, in addition to the price, defect rate

can be considered as the second attribute in the auctions where the suppliers can differentiate their quality. For the auctions in the supply chain management where a manufacturer selects the suppliers of some products/services, lead time can also be considered as an additional attribute. Suppliers can make different bids with different attribute value combinations. The auction in the transportation industry may be another major area of application. On-time delivery performance can be considered as an attribute in such auctions.

Our decision support mechanism, ADSS, can be utilized in assigning product/service combinations to suppliers through online auctions.

CHAPTER 3

AN APPROACH FOR MULTI-ATTRIBUTE MULTI-ITEM AUCTIONS

In this chapter we develop an approach that provides aid both to the buyer and the sellers in a MAMI multi-round reverse auction environment. We first give an overview of the approach and the models used. We then discuss the details of the algorithm. We next provide the experimental setting and demonstrate the algorithm for the 2-attribute case numerically. We then discuss the algorithm for 3-attribute linear problems. Lastly, we present a heuristic approach in a 2-attribute nonlinear problem setting.

3.1 The Approach

We develop an auction decision support system (ADSS) that supports sellers to bid on multiple items. We assume a linear preference function and in each round we find all efficient bid combinations. For the sake of completeness, we provide the stages of the approach and the corresponding models used in each stage, in Figure 3.1.

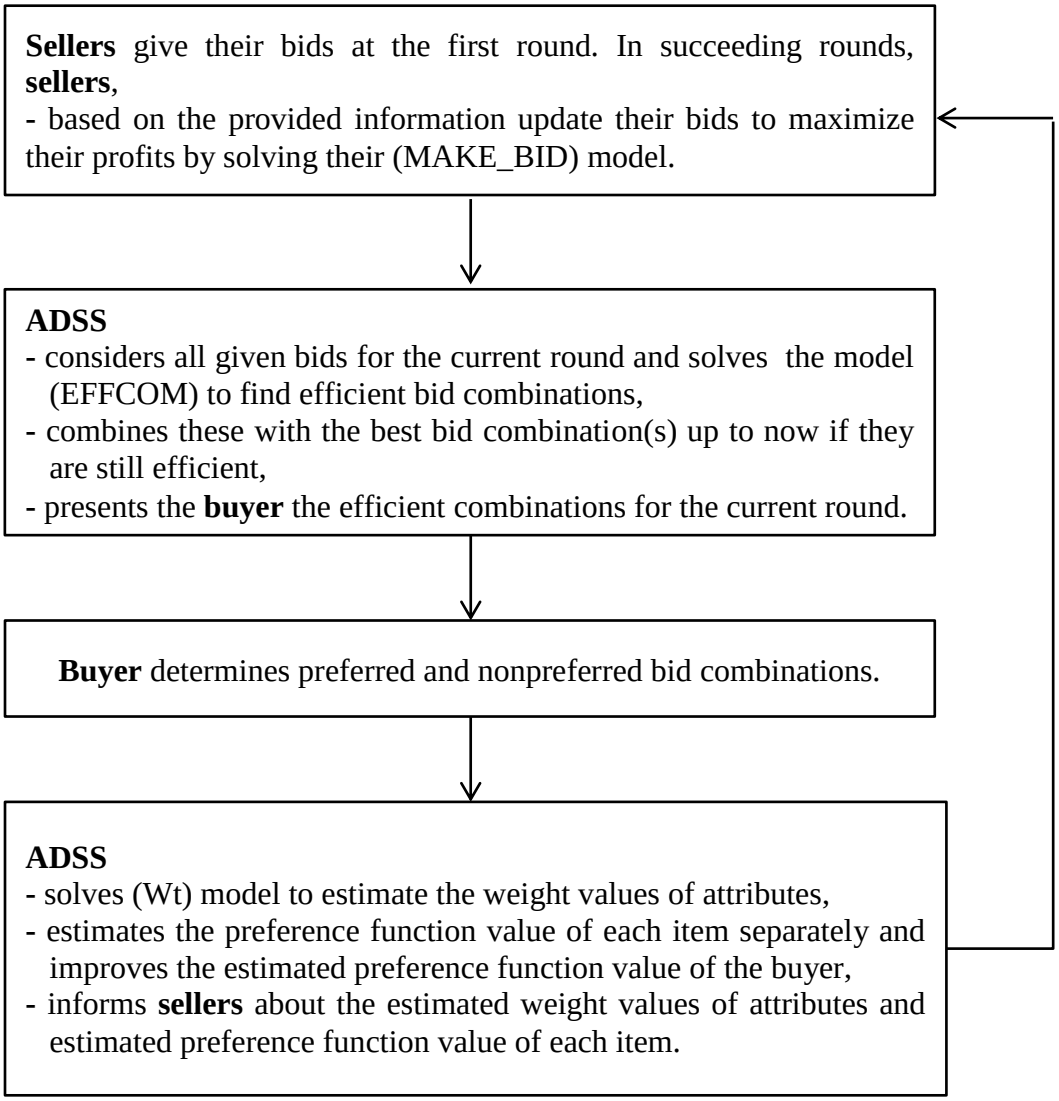


Figure 3.1 The stages of the approach with the corresponding models

3.2 The Efficient Combination Model

After taking the updated bids from the sellers, we solve the following (EFFCOM) model to find the efficient combination(s) where all the auctioned items are supplied. We apply a variation of the ε-constraint method of Haimes et al. (1971).

Parameters:

a_{itj}: level of attribute *j* offered by seller *i* in bid *t*

π_{itm}: 1 if bid *t* of seller *i* includes item *m*; 0 otherwise

T_i: the number of bids offered by seller *i*

ρ: a small positive constant

ε_j: upper bound level for attribute *j* and it changes from solution to solution

Decision Variables:

y_{it}: 1 if bid *t* of seller *i* is selected to be in the efficient combination; 0 otherwise

Price is a typical attribute in auctions and we define it as attribute 1 for convenience of notation, without loss of generality.

Model (EFFCOM)

$$\text{Min} \sum_{i=1}^{I} \sum_{t=1}^{T_i} a_{it1} y_{it} + \rho \sum_{i=1}^{I} \sum_{t=1}^{T_i} \sum_{j=2}^{J} a_{itj} y_{it} \qquad (1.1)$$

s.to

$$\sum_{i=1}^{I} \sum_{t=1}^{T_i} \pi_{itm} y_{it} \geq 1 \qquad \forall m \qquad (1.2)$$

$$\sum_{i=1}^{I} \sum_{t=1}^{T_i} a_{itj} y_{it} \leq \varepsilon_j \qquad j \neq 1 \qquad (1.3)$$

$$y_{it} \in \{0,1\} \qquad (1.4)$$

We optimize one objective and in (1.3) restrict the other to some upper bound value. In order to guarantee an efficient solution, we augment the objective function of the standard ε-constraint method. We multiply sum of the constrained objectives with a small positive constant as the augmented part. Constraint set (1.2) guarantees satisfying the demand for each item. We use (1.4) to enforce that bids are indivisible.

We systematically change ε_j and solve (EFFCOM) repeatedly to obtain different efficient solutions. If there are more than two attributes, finding a representative set of efficient solutions is more cumbersome by systematically changing ε_j. For the two-attribute case, we systematically change ε_j values to generate all efficient solutions. We derive the suitable ρ value that does not cause any trade-offs with the first term, in the objective function and only has an effect of breaking ties (see Appendix A for details of the reduction in ε_j values and setting ρ). We solve the

(EFFCOM) by using GAMS 23.8 for two attribute problems. For three attribute problems we use the algorithm suggested by Lokman and Köksalan (2012) to find all the efficient solutions for three attribute problems.

We assume that the attribute values of combinations of bids are additive. Here, to find the efficient bid combinations we need to consider different combinations of the available bids. When the total number of bids is large, the computational burden may become excessive.

We find the efficient combinations by solving (EFFCOM) and assign them index values to keep track of them. Let the index set of efficient combinations for the current round be E. We assume that the buyer determines the preferred and nonpreferred bid combinations in E. We recognize the fact that the buyer may not be able to state very precise preference statements when bid combinations are close to each other in terms of buyer's preferences. In such cases, the buyer could indicate indifference between such bid combinations. It is sufficient for our purposes that the buyer determines only the preferred and nonpreferred combinations. To illustrate, consider four alternatives (bid combinations) $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ and $\boldsymbol{D}$ and assume the buyer is indifferent between $\boldsymbol{A}$ and $\boldsymbol{B}$, $\boldsymbol{B}$ and $\boldsymbol{C}$, and prefers $\boldsymbol{A}$ to $\boldsymbol{C}$, $\boldsymbol{B}$ to $\boldsymbol{D}$, and $\boldsymbol{C}$ to $\boldsymbol{D}$. The buyer may not identify alternative $\boldsymbol{A}$ as the best alternative and we assume that he/she provides us with the information that $\boldsymbol{A}$ and $\boldsymbol{B}$ are the preferred combinations and combination $\boldsymbol{D}$ is worse than both $\boldsymbol{A}$ and $\boldsymbol{B}$.

Based on the preferences of the buyer, we solve the parameter estimation model (Wt) explained in the following section.

3.3 The Parameter Estimation Model

We assume a linear preference function. Therefore the weighted L_α metric that represents the underlying preferences of the buyer approximately (explained in Section 2.5) is reduced to $u(\boldsymbol{b}_{it}) = \sum_{j=1}^{J}(w_j a_{itj})$. We estimate the parameters (weights) of the preference function by solving the following (Wt) model.

For simplicity, we omit the subscripts indicating rounds. An efficient combination is composed of bids of several sellers. For the sake of simplicity, we introduce the notation $\boldsymbol{e}_k$ to represent the k^{th} efficient combination, $k \in E$, where E is the index set of efficient combinations presented to the buyer for the corresponding round.

Parameters:

Δ: predetermined threshold level by which the buyer can distinguish between bid combinations

ρ: a small positive constant

e_{kj}: level of attribute j in efficient combination k

E: index set of efficient combinations

Decision Variables:

μ : an auxiliary variable (to measure the estimated value difference between alternatives and bound the weights)

$\widehat{w}_j$: estimated weight of attribute j

Model (Wt)

Max μ (2.1)

s.to

$\sum_{j=1}^{J} \widehat{w}_j = 1$ (2.2)

$\widehat{w}_j \geq \mu$ (2.3)

$u(\boldsymbol{e}_k) = \sum_{j=1}^{J} \widehat{w}_j e_{kj}$ $k \in E$ (2.4)

$u(\boldsymbol{e}_n) \geq u(\boldsymbol{e}_p) + \Delta + \mu$ for each $\boldsymbol{e}_p \succ \boldsymbol{e}_n$ (2.5)

$u(\boldsymbol{e}_l) - \Delta + \rho + \mu \leq u(\boldsymbol{e}_r) \leq u(\boldsymbol{e}_l) + \Delta - \rho - \mu$ for each $\boldsymbol{e}_l \sim \boldsymbol{e}_r$ (2.6)

$\mu \geq 0$ (2.7)

The objective (2.1) is to find the maximum μ value that satisfies the constraints. This leads the solution to be at a central point of the feasible weight space; i.e. it tries to locate the weights as far from each preference constraint as possible. We

use normalized weights by constraint (2.2). (2.3) imposes lower bounds on $\widehat{w}_j$. Bid combinations are evaluated in terms of a weighted linear preference function (2.4). The past preferences of the buyer are modeled by (2.5) and (2.6). We make sure in (2.5) that the value difference between the preferred alternative and the inferior alternative is at least as big as the threshold, Δ. Similarly, (2.6) guarantees that the value difference between indifferent alternatives do not exceed Δ. The Δ value has to be positive in order to make sure that there will be a difference in the estimated preference values of the preferred and inferior combinations.

We provide sellers with the weight values found from (Wt), $\widehat{w}_j$ as well as the estimated preference function value of each item separately. Since some of the items in the preferred bid combination are given as bundles, in our simulations we determine the preference function value of the items in each bundle. To estimate the preference function value of the items in the bundle, we use the estimated preference function value of the items given by the seller of that bundle for the current round. We assign preference function values to the items in the bundle proportional to their estimated preference function values as singletons. We take the average of the estimations of the current and the previous rounds. We impose an improvement to the new estimated values and provide the resulting information to the sellers. Assume, for example, that seller *i*'s bundled bid given for items 1 and 2 is in the winning combination of the current round. Let the estimated preference function value of the bundle be 12 and the estimated preference function value of items 1 and 2 proposed by seller *i* be 5 and 10, respectively for the current round. Then we estimate the preference function value of item 1 to be 4 and item 2 to be 8 for the current round. Suppose that in the previous round we inform the sellers that the estimated preference function values for items 1 and 2 should be at most 5 and 9, respectively. Suppose that the current round is round 3. Then we inform the sellers that the estimated preference function value for items 1 and 2 should be at most, $\left(\frac{4+2*5}{3}\right)(1-\gamma)$ and $\left(\frac{8+2*9}{3}\right)(1-\gamma)$, respectively. In this method, besides the estimations in the current round we also consider the previous rounds' estimations. We then apply

$100\gamma\%$ improvement in order to end up with an improved combination in the next round. Different techniques could also be devised to estimate the preference function value of the items in the bundle.

3.4 Sellers' Model

We assume that each seller determines a minimum mark-up percentage that he/she uses and he/she solves the MAKE_BID model below to find the most profitable bids (bundle and singletons) for him/her based on our estimations. If there are feasible bids with extra profit, then the seller gives the best possible bids with his/her predetermined mark-up. Suppose that seller *i* has a minimum mark-up of $v_i\%$, that is, if the cost of a bid at a specified defect rate is *C*, then the price of the corresponding bid should be at least $(1 + v_i/100)C$. We assume that sellers do not incur losses and therefore, we use nonnegative mark-ups.

We first provide the seller's model for the 2-attribute case where the attributes are *price* and *defect rate*.

Parameters:

w: estimated weight of price

EP_t: estimated preference function value for bid *t* (ADSS provides the estimated preference function value for each item separately and the estimated preference function of a bundle is calculated by summing up the estimated preference function values of the items in the bundle)

v_i^{perc}: minimum mark-up percentage for seller *i*; if it is 0, then seller *i* may bid with zero profit. For the sake of simplicity let $v_i = v_i^{perc}/100$.

f_{it}: the cost function of seller *i* for bid *t*

$$f_{it}(d_{it}) = \left(1 - \left(1 - \left(1 - \frac{d_{it} - D_{it}^-}{D_{it}^+ - D_{it}^-}\right)^{q_i}\right)^{1/q_i}\right)(C_{it}^+ - C_{it}^-) + C_{it}^-$$

where

D_{it}^+: maximum defect rate value that can be offered by seller *i* for bid *t*

D_{it}^-: minimum defect rate value that can be offered by seller *i* for bid *t*

C_{it}^{+}: maximum cost of bid *t* for seller *i* (at D_{it}^{-})

C_{it}^{-}: minimum cost of bid *t* for seller *i* (at D_{it}^{+})

q_i: parameter of the L_q cost function for seller *i*

Decision Variables:

p_{it} = price offered by seller *i* for bid *t*

d_{it} = defect rate offered by seller *i* for bid *t*

Model (MAKE_BID$_{it}$)

Max $p_{it} - (1 + v_i) f_{it}(d_{it})$ (3.1)

s.to

$D_{it}^{-} \leq d_{it} \leq D_{it}^{+}$ (3.2)

$w.p_{it} + (1 - w) d_{it} \leq EP_t$ (3.3)

The objective (3.1) is to maximize the profit value for bid *t* of seller *i* considering his/her mark-up value. We consider the feasible defect rate range for the corresponding bid in (3.2). (3.3) guarantees that the estimated preference function value with the updated bids does not exceed the estimated preference function value of bid *t*.

In our experiments we simulate the cost functions of the sellers using convex functions, specifically L_q functions (see Köksalan, 1999). We show the L_q function for different *q* values in Figure 3.2.

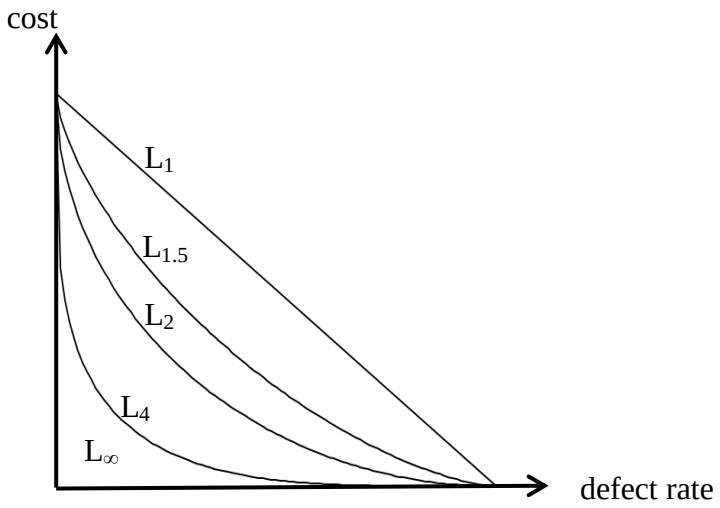


Figure 3.2 L_q function for different q values

In these functions, there is an inverse relation between cost and defect rate. That is less defect rate (better quality) costs more. For larger q values, the extreme values in each attribute gets harder to achieve. To achieve better values in one attribute, the amount you need to sacrifice from other attribute increases as you are close to the more preferred values (extremes).

The model is nonlinear due to the objective function which is equivalent to

$$\text{Max } z(d_{it}) = \frac{EP_t-(1-w)d_{it}}{w} - (1+v_i)f_{it}(d_{it}).$$

The first term in the objective function is linear and linear functions are special cases of concave functions. In the second term, the convex cost function is multiplied with a negative constant and $-(1+v_i)f_{it}(d_{it})$ is concave. Since the weighted sum of concave functions, with positive weights, is concave, $z(d_{it})$ is concave. Instead of using a solver, we utilize the properties of concave functions and determine the updated bids as follows:

The objective function can be rewritten as follows:

$$\text{Max } z(d_{it}) = \frac{EP_t-(1-w)d_{it}}{w} - (1+v_i)\left(1-\left(1-\left(1-\frac{d_{it}-D_{it}^-}{D_{it}^+-D_{it}^-}\right)^{q_i}\right)^{1/q_i}\right)(C_{it}^+ - C_{it}^-) - (1+v_i)C_{it}^-$$

Since $z(d_{it}^*)$ is concave, to find the optimal defect rate value, d_{it}^*, we check the stationary and the boundary points.

At stationary point: $z(d_{it}^*) = 0$

$$z' = -\frac{(1-w)}{w} + \frac{(1+v_i)(C_{it}^+ - C_{it}^-)}{D_{it}^+ - D_{it}^-}(D_{it}^+ - d_{it})^{q_i-1}((D_{it}^+ - D_{it}^-)^{q_i} - (D_{it}^+ - d_{it})^{q_i})^{(1-q_i)/q_i} = 0$$

After some manipulations, we obtain

$$d_{it}^* = D_{it}^+ - \frac{(1-w)^{1/(q_i-1)}(D_{it}^+ - D_{it}^-)^{q_i/(q_i-1)}}{\left(\left((1-w)(D_{it}^+ - D_{it}^-)\right)^{q_i/(q_i-1)} + \left(w(C_{it}^+ - C_{it}^-)(1+v_i)\right)^{q_i/(q_i-1)}\right)^{1/q_i}}.$$

d_{it}^* is the only stationary point of the function and the boundary points are D_{it}^- and D_{it}^+. Indeed, from the equation above we see that $D_{it}^- < d_{it}^* < D_{it}^+$. Therefore, there is no need to check the boundary points and the optimal defect rate value is d_{it}^* since $z'(d_{it}^*) = 0$.

If $\frac{EP_t - (1-w)d_{it}^*}{w} > (1+v_i)f_{it}(d_{it}^*)$, $p_{it}^* = \frac{EP_t - (1-w)d_{it}^*}{w}$ and seller bids profitably. Otherwise, $p_{it}^* = (1+v_i)f_{it}(d_{it}^*)$ and seller bids with his/her predetermined mark-up only and no extra profit is possible.

We discuss the 2-attribute case here explicitly. The procedure is also directly applicable for more than two attributes. For more than two attribute case, we assume that all non-price attributes are reflected in the cost function and therefore cost function is depicted as $f_{it}(\boldsymbol{a}_{it})$ where $\boldsymbol{a}_{it}$ is the vector of non-price attribute values of bid *t* of seller *i*, $\boldsymbol{a}_{it} = \left(a_{it2}, a_{it3}, \dots, a_{itj}, \dots, a_{itJ}\right)$. The optimal non-price attribute values, $\left(a_{it2}^*, a_{it3}^*, \dots, a_{itj}^*, \dots, a_{itJ}^*\right)$, can be found by setting $\nabla z(\boldsymbol{a}_{it}^*) = 0$. The 3-attribute case is discussed in Section 3.9 in detail.

After obtaining the updated bids from the sellers, we find the set of efficient bid combinations of the current round using the estimated weights found at the end of the previous round and continue.

3.5 The Algorithm (*ALL-e*)

We provide the steps of the algorithm below:

Recall that E denotes the index set of efficient bid combinations presented to the buyer for the corresponding round, $\boldsymbol{e}_k$ denote the k^{th} efficient bid combination, $k \in E$, and $u(\boldsymbol{e}_k)$ be the estimated preference function value of the buyer for $\boldsymbol{e}_k$. Let X_h denote the set of constraints derived from the preferences of the buyer in round h and let $X_0 = \emptyset$. We assume that the buyer's underlying preference function is linear and sellers are informed about this.

Step 1: Sellers place initial bids. Set the round counter h = 1.

Step 2: Solve (EFFCOM) to find the efficient bid combination(s) for round h and also consider the best combination(s) up to current round. Place the efficient combinations in set E. Present the buyer all combinations in set E and ask him/her to choose the preferred and nonpreferred bid combinations. If a termination condition is met, go to *Step 5*. Otherwise go to *Step 3*.

Step 3: Update the preference constraint set;

$$X_h = X_{h-1} \cup \begin{Bmatrix} u(\boldsymbol{e}_n) \geq u(\boldsymbol{e}_p) + \Delta + \mu \text{ for each } \boldsymbol{e}_p \succ \boldsymbol{e}_n \text{ and } p, n \in E \\ u(\boldsymbol{e}_l) - \Delta + \rho + \mu \leq u(\boldsymbol{e}_r) \leq u(\boldsymbol{e}_l) + \Delta - \rho - \mu \text{ for each } \boldsymbol{e}_l \sim \boldsymbol{e}_r \text{ and } l, r \in E \end{Bmatrix}$$

Solve (Wt) to fit a preference function that satisfies the constraint set X_h. Let the estimated preference function value of the best bid combination of the current round be u^*.

Step 4: Move to a 100γ% improved contour with an estimated preference function value of u^h, i.e., $u^h = u^*(1-\gamma)$. Find the preference function value of each item separately and provide the bidders with this information in addition to the current estimated weight values. Let sellers update their bids by solving their own (MAKE_BID) problem. Set $h \leftarrow h + 1$ and go to *Step 2*.

Step 5: Stop. The preferred combination(s), $\boldsymbol{e}_p$, is (are) the winning combination(s) for $p \in E$. The sellers providing items in the winning combination are the winning sellers.

If there are more than one winning combinations in *Step 5*, the buyer selects one of them using additional information.

Sellers can give bids for single items or for bundles. They update their bids and the auction continues until a termination condition is met. For multi-item case, we modify *Steps 2* and *4* of the original algorithm in Karakaya and Köksalan (2011). In *Step 2*, we find all efficient bid combinations before presenting to the buyer. In *Step 4*, we find the preference function value of each item to support the sellers.

3.6 The Modified Algorithm (*ELIM-e*)

We make some modifications to *ALL-e* and develop *ELIM-e*. In *ALL-e*, we find all efficient combinations and present them to the buyer. In the modified version, we eliminate the combinations that would be considered inferior by the buyer based on his/her past preferences, before presenting those to the buyer.

Elimination by Weight Space Reduction Models

We keep the most preferred combination up to the current round and add it to the set of efficient combinations for the current round. Let the index set of efficient combinations for the current round be G. We solve (ELIMINATION) model to eliminate combinations that would be inferior based on the information derived from the buyer's previous selections and based on the assumed form of his/her preference function.

Parameters:

Δ: predetermined threshold level by which the buyer can distinguish between bid combinations

G : index set of efficient combinations for the current round before elimination

e_{kj}: level of attribute j in efficient combination k

Decision Variable:

μ : an auxiliary variable (to measure the estimated value difference between alternatives)

w_j': possible weight of attribute *j*

Model (ELIMINATION$_v$)

Min μ (4.1)

s.to

$\sum_{j=1}^{J} w_j' = 1$ (4.2)

$u(\boldsymbol{e}_k) = \sum_{j=1}^{J} w_j' e_{kj}$ $k \in G$ (4.3)

$u(\boldsymbol{e}_v) \leq u(\boldsymbol{e}_s) - \mu$ $\forall s \neq v$ (4.4)

$u(\boldsymbol{e}_n) \geq u(\boldsymbol{e}_p) + \Delta$ for each $\boldsymbol{e}_p \succ \boldsymbol{e}_n$ (4.5)

$u(\boldsymbol{e}_l) - \Delta + \rho \leq u(\boldsymbol{e}_r) \leq u(\boldsymbol{e}_l) + \Delta - \rho$ for each $\boldsymbol{e}_l \sim \boldsymbol{e}_r$ (4.6)

$w_j' \geq 0$ $\forall j$ (4.7)

In the model, combination *v* is compared to other combinations in *G*. In each comparison we try to find a weight vector in the feasible weight space that makes combination *v* better than each of the remaining combinations. The feasible weight space is determined based on the past preferences of the buyer (constraint sets 4.5 and 4.6). If we can find a feasible solution with positive μ value, we say that there exist weights in the feasible weight space that makes combination *v* to be preferred by the buyer. Otherwise, we say that combination *v* cannot be preferred by the buyer.

We solve (ELIMINATION$_v$) model for each *v* in *G*. If we can find a feasible solution to (ELIMINATION$_v$) with positive μ value, we place *v* in set *E* where *E* is the index set of efficient combinations for the current round that will be presented to the buyer. In the two attribute case, we can construct *E* by using the following procedure where ρ is a small positive constant, subscript 1 and 2 refer to *price* and *defect rate,* respectively.

Since there are only two attributes, it is sufficient to specify bounds on the weight of attribute 1. Let WL and WU be the estimated lower and upper bounds for the weight of attribute 1, respectively, based on the past preference of the buyer, and let *K* be the cardinality of set *G*.

In the procedure, for each pairwise comparison of combination *v* with other combinations in *G*, we check whether the resulting weight space is feasible or not. If it is feasible we place *v* in set *E*; otherwise we delete *v* from *G* due to Theorem 1 and the procedure is continued.

Theorem 1: Let $\boldsymbol{e}_A$ and $\boldsymbol{e}_B$ be two alternatives such that $w^{temp} = \frac{e_{B2} - e_{A2}}{e_{A1} - e_{A2} - e_{B1} + e_{B2}}$ be the weight of attribute 1 that makes $\boldsymbol{e}_A$ and $\boldsymbol{e}_B$ have equal preference values. If $e_{A1} < e_{B1}$ and $w^{temp} + \rho > WU$ or if $e_{A1} > e_{B1}$ and $w^{temp} - \rho < WL$, then $\boldsymbol{e}_A$ cannot be preferred to $\boldsymbol{e}_B$ by the DM based on his/her past preferences, where ρ is a sufficiently small positive constant.

Proof: If $e_{A1} < e_{B1}$, then $w^{temp} + \rho$ is the smallest possible value of weight of attribute 1 that makes $\boldsymbol{e}_A$ preferred to $\boldsymbol{e}_B$; whereas if $e_{A1} > e_{B1}$, then $w^{temp} - \rho$ is the largest possible value of weight of attribute 1 that makes $\boldsymbol{e}_A$ preferred to $\boldsymbol{e}_B$. Based on the past preferences of the DM, the estimated lower and upper bounds of weight of attribute 1 are WL and WU, respectively. If $e_{A1} < e_{B1}$ and $w^{temp} + \rho > WU$ or if $e_{A1} > e_{B1}$ and $w^{temp} - \rho < WL$, then there are no weights in the feasible weight space that make $\boldsymbol{e}_A$ preferred to $\boldsymbol{e}_B$. □

The procedure used to solve (ELIMINATION$_v$)

Step 0: Set $E = \emptyset$, $v = 1$, $s = 1$ and $K = |G|$.

Step 1: Set $s \leftarrow s + 1$. If $s \in G$, go to *Step 2*, else if $s > K$ go to *Step 4*; otherwise go to *Step 1*.

Step 2: If $s = v$ go to *Step 1*. Otherwise, set $w^{temp} = \frac{e_{s2} - e_{v2}}{e_{v1} - e_{v2} - e_{s1} + e_{s2}}$ and if $e_{v1} - e_{v2} - e_{s1} + e_{s2} > 0$ go to Step 2.1; otherwise go Step 2.2.

Step 2.1: If $w^{temp} - \rho < WL$, go to *Step 3*. Otherwise go to *Step 1*.

Step 2.2: If $w^{temp} + \rho > WU$, go to *Step 3*. Otherwise go to *Step 1*.

Step 3: Set $G = G - \{v\}, v \leftarrow v + 1$ and go to *Step 5*.

Step 4: Set $E = E \cup \{v\}, v \leftarrow v + 1$ and go to *Step 5*.

Step 5: If $v \in G$, set $s = 0$ and set go to *Step 1*; otherwise stop.

We apply the procedure for each *v* in *G* and after the eliminations, we present all combinations in *E* to the buyer. The rest of the procedure is the same as *ALL-e*.

3.7 Experimental Setting

To test the performance of the algorithm we generate test problems. We consider two versions of the test problems in terms of the number of attributes: 2-attribute and 3-attribute cases. In the 2-attribute case, we consider two specific attributes, *price*, and *defect rate*. In the 3-attribute case, we include *lead time* as the third attribute. In both cases, all attributes are to be minimized.

We consider two different versions of the test problems in terms of the underlying preference function of the buyer: *linear* and *nonlinear* cases.

3.7.1 Test Problem Generation

We demonstrate the performance of our algorithm by generating test problems. In the literature combinatorial auction test suites are available for single attribute auctions (Fujishima et al., 1999; Leyton-Brown et al., 2000; Sandholm et al., 2005). Buer and Pankratz (2010), generate their test instances for two attribute transportation problems. To the best of our knowledge, all the combinatorial auction test problems are generated to determine the winner of the single round auctions. We modify the technique proposed by Buer and Pankratz (2010) to generate the cost functions of the sellers. We use these generated cost functions during the multi-round auction process.

As mentioned before, we consider two specific attributes, *price*, and *defect rate*, where smaller values are preferred in both attributes by the buyer. We suppose that each seller has his/her own competitive item combinations and he/she makes bids for these combinations. We use TS_i, TB_i and T_i to denote the set of singleton bids, bundled bids and total bids, respectively, given by seller *i* where $T_i = TS_i \cup TB_i$. We replace the original notation of the non-price attribute, defect rate, with $\boldsymbol{d}_i = (d_{i1}, d_{i2}, \dots, d_{it}, \dots, d_{iT_i})$ where d_{it} is the defect rate value of bid *t* offered by seller *i*. Each seller identifies a price and defect rate for each bid he/she offers. Here defect rate is used as an indicator of quality; smaller defect rate values stand for higher quality. We later discuss how we evaluate the quality of a bundle.

As Leyton-Brown et al. (2000) state, some items may be more suitable to group together and this may differ from seller to seller. To capture this property, we generate a relation matrix for each seller like the synergy matrix in Buer and Pankratz (2010). This matrix consists of 0's and 1's. "1" indicates that grouping the corresponding items decreases the cost of the seller, whereas "0" indicates that grouping the corresponding items does not have an effect on the seller's cost. We consider a cost function that takes economies of scope into account.

We randomly generate a relation matrix for each seller. For each seller-item pair we assign defect rate and *resource requirement values* that will determine which items the seller can combine to create bundles. Buer and Pankratz (2010) use resource requirement as an indicator of the difficulty of supplying an item for a seller. This difficulty may reflect itself in the cost of the item. The resource requirement of an item may differ from seller to seller due to reasons such as the differences in the technologic infrastructures of the firms. Defect rate levels and resource requirement values are generated randomly from uniform distributions in the ranges $[dl, du]$ and $[rl, ru]$ respectively. We assume that the cost and defect rate are inversely proportional, whereas the cost and resource requirement are directly proportional. That is, smaller defect rates and higher resource requirement levels will result in higher cost values.

Generating cost values of singletons

Recall that d_{it} denotes the defect rate value of singleton bid *t* for seller *i*. Let r_{it} and cst_{it} be the resource requirement and cost of singleton *t*, respectively. Let $R = (rl + ru)/2$ be the mean of the uniform distribution used to generate r_{it} values and $D = \frac{\ln(du)-\ln(dl)}{du-dl}$ be the mean of the distribution that generates $1/d_{it}$ values. Let *max_cost* and *min_cost* be the maximum and minimum values that a singleton *t* can take, respectively. We generate the costs of bids as follows:

Step 0: Set $i = 1$.

Step 1: Set $t = 1$.

Step 2: $\mu_{it} = 1 + \frac{r_{it}}{R}\frac{1/d_{it}}{D}$

Step 3: Generate a random variate, *v*, from normal distribution with mean μ_{it} and variance 1.

Step 4: If *min_cost* $\leq v \leq$ *max_cost*, set $cst_{it} = v$. Otherwise go to *Step 3*.

Step 5: If $t < M$, then set $t \leftarrow t + 1$ and go to *Step 2*.

Step 6: If $i < I$, then set $i \leftarrow i + 1$ and go to *Step 1*. Otherwise, stop.

The cost value of singleton *t* for seller *i*, cst_{it}, is randomly generated based on its defect rate value and resource requirement. We set rl=0.1, ru=0.5, *min_cost* = 0.5 and *max_cost* = 1.5 as in Buer and Pankratz (2010). Buer and Pankratz (2010) generate quality values directly, whereas we generate defect rate as an indicator of quality using dl = 0.2 and du=1.

As stated above, each seller is assumed to have his/her own suitable item combinations. While computing the defect rate and the resource requirement levels of a bundle, we sum the defect rate and the resource requirement levels of the items that the bundle consists of, respectively. As stated before, the resource requirement levels are somewhat artificial values generated to restrict the item combinations that can be bundled together. They also have impact on costs. We assume that the resource requirement of a bid cannot exceed 1, that is the sum of

the resource requirement levels of the items in a bundle should be less than or equal to 1 as suggested by Buer and Pankratz (2010). We assign a lower and upper bound to a bundled bid according to the costs of the singletons and the bundles it consists of. Then we randomly generate a cost value for the bundled bid between its lower and upper bounds.

Generating cost values for bundles

Recall that TB_i and T_i denote the set of bundled bids and total bids given by seller *i*. Let n_{it} be the number of items in bid *t* of seller *i*. Let H_{it} be the set of singletons and bundled bids that contains all possible subsets of bundled bid *t* of seller *i*. Let P_{it} be the set of bid compositions whose unions constitute bundled bid *t* of seller *i* and whose intersections are empty. For example, let bundled bid *t* composed of items 1, 2 and 3. Then, $H_{it} = \{(1), (2), (3), (1,2), (1,3), (2,3)\}$ and $P_{it} = \{[(1), (2), (3)], [(1), (2,3)], [(2), (1,3)], [(3), (1,2)]\}$. The cost value of a composition in P_{it} is the sum of cost values of the bids it contains. We describe how we generate the cost values for the bundles next.

Step 0: Set $i = 1$

Step 1: Set $z = 2$.

Step 2: $\forall t \in TB_i$ for which $n_{it} = z$, set $LB_{it} = max_{k \in H_{it}}\{cst_{ik}\}$ and $UB_{it} = min_{m \in P_{it}}\{cst_{im}\}$. Generate a random variate, v, from uniform distribution in the interval (LB_{it}, UB_{it}). Set $cst_{it} = v$.

Step 3: Set $z \leftarrow z + 1$. If there are no bundled bids in T_i with size z, then stop. Otherwise, go to *Step 2*.

Step 4: If $i < I$, then set $i \leftarrow i + 1$ and go to *Step 1*. Otherwise, stop.

After generating the sellers' bids and their corresponding attribute values, we use L_q functions (see Köksalan, 1999) to generate cost and defect rate pairs for which the seller would be indifferent. The L_q function can be written as:

$(1-d'_{it})^q+(1-c'_{it})^q=1$ where $q \geq 1$, d'_{it} and c'_{it} refer to the scaled defect rate and cost values.

Commonly used L_q functions use q values of 1, 2, and ∞. $q=1$, $q=2$, and $q=\infty$ correspond to rectilinear, Euclidean, and Tchebycheff distance functions, respectively.

To generate different cost functions for the sellers, we randomly generate a q value to represent the cost function of each seller. After generating the cost function we randomly assign initial mark-up percentages (explained in Section 3.4) between *mpl* and *mpu* to the sellers for the first round.

We develop above the general framework of the cost function to experiment with. We also consider cost functions with three attributes (Section 3.9).

3.7.2 Parameter Setting

We present our approach in an environment where there are three sellers and five items where seller 1 has 11, seller 2 has 7 and seller 3 has 10 preferred combinations. Based on our preliminary experiments, we set the threshold, $\Delta=0.01$ and $\gamma=0.1$ implying a required improvement of 10% in each round. We use an ideal point for each attribute in our estimated preference function. Since we minimize all attributes, we set the ideal point to the point where both attributes are zero in this thesis. Without loss of generality, we set minimum mark-up values to zero for each seller. That is, all sellers can bid with zero profits. We terminate at the round when each seller bids with zero profit in all his/her bids.

There may be alternative termination conditions. For instance the buyer could decide to terminate, for example, when he/she finds the improvement between rounds to be small. It is also possible for some sellers to stop bidding if he/she cannot be among provisional winners for a number of rounds.

We use this problem setting to test the performance of the algorithms throughout this thesis.

3.8 A Numerical Example for the 2-attribute Case

We provide a numerical example for *ELIM-e*. Both *ALL-e* and *ELIM-e* have similar steps and since *ELIM-e* also includes an elimination procedure, we give an example for this version only. In our experiments, we assume that the buyer has a specific underlying true preference function, which we use to simulate his/her preferences. In this example, we assume that the weights of attributes of the buyer's underlying preference function are 0.55 and 0.45 for price and defect rate, respectively.

We round the values to four significant digits. We use the buyer's underlying preference function to report the preference values. We also report the estimated preference values. In this chapter, we present the buyer efficient bid combinations and we assume that he/she identifies the preferred and nonpreferred bid combinations. Initial bids are given in Table 3.1.

Table 3.1 Initial bids

Round 1					
Seller	Bid	Item	Price	Defect Rate	Profitable
1	1	1	0.7707	0.8325	1
1	2	2	1.9295	0.6873	1
1	3	3	1.7187	0.65	1
1	4	4	1.2476	0.5149	1
1	5	5	0.7534	0.4734	1
1	6	1,3	2.1041	1.4825	1
1	7	2,5	2.5209	1.1607	1
1	8	3,5	2.2412	1.1234	1
1	9	1,3,5	2.563	1.9559	1
1	10	2,3,5,	3.3081	1.8107	1
1	11	1,2,3,5	3.9188	2.6432	1
2	1	1	0.9578	0.3709	1
2	2	2	1.4626	0.5018	1
2	3	3	0.8558	0.7844	1
2	4	4	1.8825	0.2289	1
2	5	5	1.4358	0.5573	1
2	6	1,3	1.4015	1.1553	1
2	7	2,5	2.0246	1.0591	1
3	1	1	1.6854	0.8493	1
3	2	2	1.7669	0.2768	1
3	3	3	1.7157	0.6529	1
3	4	4	1.6948	0.4581	1
3	5	5	2.0264	0.4346	1
3	6	1,3	3.181	1.5022	1
3	7	1,4	2.9701	1.3074	1
3	8	2,4	3.2017	0.7349	1
3	9	2,5	2.8067	0.7114	1
3	10	1,3,4	3.8325	1.9603	1

In the tables, the items constituting a bid are given under the “Item” column. The “Profitable” column represents whether the bid is profitable for the seller or not. As soon as all sellers have zero in this column, indicating all sellers bid unprofitably, the algorithm stops.

Table 3.2 Efficient bid combinations for Round 1

Round 1						
Combination	Seller	Bid	Item	Price	Defect Rate	Buyer's True Pref. Fn. Value
1	1	3	3	1.7187	0.6500	4.9337
	2	1	1	0.9578	0.3709	
	2	4	4	1.8825	0.2289	
	3	9	2,5	2.8067	0.7114	
2	1	8	3,5	2.2412	1.1234	4.6666
	2	1	1	0.9578	0.3709	
	2	4	4	1.8825	0.2289	
	3	2	2	1.7669	0.2768	
3	2	4	4	1.8825	0.2289	4.2929
	2	6	1,3	1.4015	1.1553	
	3	9	2,5	2.8067	0.7114	
4	1	5	5	0.7534	0.4734	4.1528
	2	4	4	1.8825	0.2289	
	2	6	1,3	1.4015	1.1553	
	3	2	2	1.7669	0.2768	
5	1	4	4	1.2476	0.5149	3.9324
	1	5	5	0.7534	0.4734	
	2	6	1,3	1.4015	1.1553	
	3	2	2	1.7669	0.2768	
6	**1**	**4**	**4**	**1.2476**	**0.5149**	3.7987
	2	**6**	**1,3**	**1.4015**	**1.1553**	
	2	**7**	**2,5**	**2.0246**	**1.0591**	

We highlight the provisional winners and their corresponding bids in bold. In Round 1, based on the bids in Table 3.1, we find 10 efficient combinations, however, after applying (ELIMINATION) procedure we are left with 6 efficient combinations given in Table 3.2. Among them, the buyer selects combination 6 as best (remaining combinations are nonpreferred combinations). Based on this information we estimate the weight values of the attributes. In fact estimating one attribute's weight is sufficient since the other weight can be estimated by subtracting the estimated weight from 1. By solving (Wt) we find the following values where w_p and w_d are the estimated weights of price and defect rate, respectively:

$w_p = 0.7309$ and $w_d = 0.2691$

The estimated preference function values are calculated using the estimated weight values and given under "Estimated Pref. Fn. Value" column in the tables. The estimated preference function values for Round 1 are provided in Table 3.3.

Table 3.3 Efficient bid combinations with estimated preference function values for Round 1

Round 1							
Combination	Seller	Bid	Item	Price	Defect Rate	Buyer's True Pref. Fn. Value	Estimated Pref. Fn. Value
1	1	3	3	1.7187	0.6500	4.9337	5.9111
	2	1	1	0.9578	0.3709		
	2	4	4	1.8825	0.2289		
	3	9	2,5	2.8067	0.7114		
2	1	8	3,5	2.2412	1.1234	4.6666	5.5435
	2	1	1	0.9578	0.3709		
	2	4	4	1.8825	0.2289		
	3	2	2	1.7669	0.2768		
3	2	4	4	1.8825	0.2289	4.2929	5.0154
	2	6	1,3	1.4015	1.1553		
	3	9	2,5	2.8067	0.7114		
4	1	5	5	0.7534	0.4734	4.1528	4.8166
	2	4	4	1.8825	0.2289		
	2	6	1,3	1.4015	1.1553		
	3	2	2	1.7669	0.2768		
5	1	4	4	1.2476	0.5149	3.9324	4.4295
	1	5	5	0.7534	0.4734		
	2	6	1,3	1.4015	1.1553		
	3	2	2	1.7669	0.2768		
6	**1**	**4**	**4**	**1.2476**	**0.5149**	3.7987	4.1504
	2	**6**	**1,3**	**1.4015**	**1.1553**		
	2	**7**	**2,5**	**2.0246**	**1.0591**		

$u^* = 4.1504$ and after improvement $u^{(1)} = 4.1504\ (1 - 0.1) = 3.7353$. For each item we estimate the following preference function values:

Table 3.4 Estimated preference function values at the end of Round 1

Round 1	
Item	Estimated Pref. Fn. Value
1	0.5874
2	0.7957
3	0.6143
4	0.9454
5	0.7926

We inform the sellers about the estimated weights of attributes. We recommend each seller that the updated preference function value of a combination should not exceed 3.7353 and we also give the estimated preference function value for each item separately.

Afterwards, each seller solves his/her own (MAKE_BID) model and the resulting bids are provided in Table 3.5.

Table 3.5 Bids for Round 2

Round 2					
Seller	Bid	Item	Price	Defect Rate	Profitable
1	1	1	0.5034	0.955	0
1	2	2	0.7234	0.9919	1
1	3	3	0.6485	0.9829	0
1	4	4	1.0786	0.5836	1
1	5	5	1.0059	0.2133	1
1	6	1,3	0.9218	1.9618	1
1	7	2,5	1.4563	1.9467	1
1	8	3,5	1.2753	1.7644	1
1	9	1,3,5	1.7199	2.7394	1
1	10	2,3,5,	1.9562	2.8716	1
1	11	1,2,3,5	2.3916	3.8717	1
2	1	1	0.6582	0.4002	0
2	2	2	1.0349	0.2013	0
2	3	3	0.4818	0.9743	1
2	4	4	0.9988	0.8003	1
2	5	5	0.7325	0.9558	1
2	6	1,3	1.1734	1.2786	1
2	7	2,5	1.6442	1.4365	1
3	1	1	0.4372	0.9951	1
3	2	2	0.8156	0.7416	1
3	3	3	1.0789	0.8557	0
3	4	4	0.9625	0.8988	1
3	5	5	0.7388	0.9386	1
3	6	1,3	1.0699	1.9563	0
3	7	1,4	1.3862	1.9307	1
3	8	2,4	1.7743	1.6508	1
3	9	2,5	1.5662	1.6482	1
3	10	1,3,4	1.8852	2.8582	1

Table 3.6 Efficient bid combinations for Round 2

Round 2							
Combination	Seller	Bid	Item	Price	Defect Rate	Buyer's True Pref. Fn. Value	Estimated Pref. Fn. Value
1	**1**	**4**	**4**	**1.0786**	**0.5836**	3.3850	3.3713
	1	**5**	**5**	**1.0059**	**0.2133**		
	2	**2**	**2**	**1.0349**	**0.2013**		
	2	**6**	**1,3**	**1.1734**	**1.2786**		
2	1	4	4	1.0786	0.5836	3.5081	3.4992
	1	5	5	1.0059	0.2133		
	2	6	1,3	1.1734	1.2786		
	3	2	2	0.8156	0.7416		
3	1	2	1	0.7234	0.9919	3.5700	3.5636
	1	4	4	1.0786	0.5836		
	1	5	5	1.0059	0.2133		
	2	6	1,3	1.1734	1.2786		
4	1	2	2	0.7234	0.9919	3.6480	3.6446
	1	5	5	1.0059	0.2133		
	2	6	1,3	1.1734	1.2786		
	3	4	4	0.9625	0.8988		
5	1	10	2,3,5	1.9562	2.8716	3.9902	4.0002
	3	1	1	0.4372	0.9951		
	3	4	4	0.9625	0.8988		
6	1	2	2	0.7234	0.9919	3.9985	4.0088
	2	3	3	0.4818	0.9743		
	3	1	1	0.4372	0.9951		
	3	4	4	0.9625	0.8988		
	3	5	5	0.7388	0.9386		
7	1	2	2	0.7234	0.9919	4.0076	4.0183
	2	3	3	0.4818	0.9743		
	3	5	5	0.7388	0.9386		
	3	7	1,4	1.3862	1.9307		
8	1	2	2	0.7234	0.9919	4.0119	4.0227
	2	3	3	0.4818	0.9743		
	2	5	5	0.7325	0.9558		
	3	7	1,4	1.3862	1.9307		

In Round 2, using the bids in Table 3.5, we find 176 efficient combinations in addition to the best combination saved from the previous round. The best combination up to the current round turns out to be dominated by some efficient combinations of the current round and we end up with 176 efficient combinations. By applying (ELIMINATION) procedure we find out that 168 of the 176 combinations would not be preferred by the buyer and we present the remaining 8

bids to the buyer. The buyer prefers combination 1 to the remaining seven combinations. Incorporating this information into (Wt) and find the following values:

$w_p = 0.5429$ and $w_d = 0.4571$

For each item we estimate the preference function values given in Table 3.7.

Table 3.7 Estimated preference function values at the end of Round 2

Round 2	
Item	Estimated Pref. Fn. Value
1	0.5024
2	0.6523
3	0.5880
4	0.8090
5	0.6463

The updated bids for Round 3 are given in Table 3.8.

Table 3.8 Bids for Round 3

Round 3					
Seller	Bid	Item	Price	Defect Rate	Profitable
1	1	1	0.5638	0.8568	0
1	2	2	0.5687	0.9736	0
1	3	3	0.6723	0.9445	0
1	4	4	1.2328	0.3056	1
1	5	5	1.0205	0.2018	1
1	6	1,3	0.9459	1.8759	0
1	7	2,5	0.8531	1.8278	1
1	8	3,5	1.5091	1.3247	0
1	9	1,3,5	1.6990	2.2149	0
1	10	2,3,5,	1.4081	2.5910	0
1	11	1,2,3,5	1.5741	3.5873	0
2	1	1	0.7368	0.2534	0
2	2	2	1.0354	0.2003	0
2	3	3	0.2908	0.9410	1
2	4	4	0.9911	0.5927	1
2	5	5	0.4333	0.8993	1
2	6	1,3	1.3444	0.7888	1
2	7	2,5	1.5883	0.9545	1
3	1	1	0.1875	0.9892	0
3	2	2	0.9139	0.5186	0
3	3	3	1.1712	0.7010	0
3	4	4	0.8296	0.7845	1
3	5	5	0.4605	0.8670	1
3	6	1,3	1.1011	1.9045	0
3	7	1,4	0.8589	1.8489	1
3	8	2,4	1.6024	1.2937	1
3	9	2,5	1.3065	1.2892	1
3	10	1,3,4	1.2394	2.6916	0

Table 3.9 Efficient bid combinations for Round 3

Round 3							
Combination	Seller	Bid	Item	Price	Defect Rate	Buyer's True Pref. Fn. Value	Estimated Pref. Fn. Value
1	1	4	4	1.2328	0.3056	3.2216	3.2140
	1	5	5	1.0205	0.2018		
	2	2	2	1.0354	0.2003		
	2	6	1,3	1.3444	0.7888		
2	1	4	4	1.2328	0.3056	3.2130	3.2079
	2	6	1,3	1.3444	0.7888		
	2	7	2,5	1.5883	0.9545		
3	1	5	5	1.0205	0.2018	3.2100	3.2059
	2	6	1,3	1.3444	0.7888		
	3	8	2,4	1.6024	1.2937		
4	**2**	**3**	**3**	**0.2908**	**0.941**	3.1865	3.1904
	3	**7**	**1,4**	**0.8589**	**1.8489**		
	3	**9**	**2,5**	**1.3065**	**1.2892**		
5	**1**	**7**	**2,5**	**0.8531**	**1.8278**	3.1795	3.1859
	2	**3**	**3**	**0.2908**	**0.941**		
	3	**7**	**1,4**	**0.8589**	**1.8489**		

In Round 3, using the bids in Table 3.8, we find 50 efficient combinations and from the previous rounds we have one more combination as the best combination up to the current round which is dominated by some efficient combinations of the current round. We end up with 50 efficient combinations. By applying (ELIMINATION) procedure we eliminate 45 combinations and we are left with 5 efficient combinations. Among these 5 combinations the buyer finds combinations 4 and 5 as best because the preference function value of the buyer for these combinations are smallest and are within the threshold Δ value of each other. Combinations 1,2 and 3 are found as nonpreferred. By solving (Wt), we find the following values:

$w_p = 0.5476$ and $w_d = 0.4524$

Table 3.10 Estimated preference function values at the end of Round 3

Round 3	
Item	Estimated Pref. Fn. Value
1	0.4601
2	0.5993
3	0.5283
4	0.7188
5	0.5688

The updated bids for Round 4 are given in Table 3.11.

Table 3.11 Bids for Round 4

Round 4					
Seller	Bid	Item	Price	Defect Rate	Profitable
1	1	1	0.5608	0.8604	0
1	2	2	0.5681	0.9743	0
1	3	3	0.6711	0.946	0
1	4	4	1.0694	0.3096	0
1	5	5	0.8719	0.2019	1
1	6	1,3	0.9431	1.8792	0
1	7	2,5	0.6488	1.8322	0
1	8	3,5	1.4975	1.3386	0
1	9	1,3,5	1.6841	2.2328	0
1	10	2,3,5,	1.3995	2.6014	0
1	11	1,2,3,5	1.5652	3.5979	0
2	1	1	0.7352	0.2553	0
2	2	2	1.0354	0.2003	0
2	3	3	0.2865	0.9421	0
2	4	4	0.8182	0.5984	1
2	5	5	0.2942	0.9011	1
2	6	1,3	1.1996	0.7987	0
2	7	2,5	1.4869	0.9658	0
3	1	1	0.1873	0.9894	0
3	2	2	0.9092	0.5242	0
3	3	3	1.1674	0.7056	0
3	4	4	0.6615	0.7881	1
3	5	5	0.3205	0.8693	1
3	6	1,3	1.0997	1.9062	0
3	7	1,4	0.6827	1.8515	0
3	8	2,4	1.4622	1.3039	0
3	9	2,5	1.1379	1.2994	0
3	10	1,3,4	1.2349	2.6969	0

Table 3.12 Efficient bid combinations for R4

Round 4							
Combination	Seller	Bid	Item	Price	Defect Rate	Buyer's True Pref. Fn. Value	Estimated Pref. Fn. Value
1	**1**	**7**	**2,5**	**0.6488**	**1.8322**	2.9198	2.9212
	2	**4**	**4**	**0.8182**	**0.5984**		
	2	**6**	**1,3**	**1.1996**	**0.7987**		
2	**1**	**7**	**2,5**	**0.6488**	**1.8322**	2.9190	2.9212
	2	**6**	**1,3**	**1.1996**	**0.7987**		
	3	**4**	**4**	**0.6615**	**0.7881**		

In Round 4, using the bids in Table 3.11, we find 31 efficient combinations some of which dominate the best combinations saved from the previous round. From these 31 combinations, 29 combinations are eliminated by using (ELIMINATION) procedure. We are left with two efficient combinations. The buyer states indifference between these two combinations and with this additional information; the estimated weight values turn out to be the same as those found in Round 3. By providing improvement for each item we estimate the preference function values given in Table 3.13.

Table 3.13 Estimated preference function values at the end of Round 4

Round 4	
Item	Estimated Pref. Fn. Value
1	0.4184
2	0.5562
3	0.4779
4	0.6469
5	0.4987

Table 3.14 Bids for Round5

Round 5					
Seller	Bid	Item	Price	Defect Rate	Profitable
1	1	1	0.5608	0.8604	0
1	2	2	0.5681	0.9743	0
1	3	3	0.6711	0.946	0
1	4	4	1.0694	0.3096	0
1	5	5	0.7439	0.2019	1
1	6	1,3	0.9431	1.8792	0
1	7	2,5	0.6488	1.8322	0
1	8	3,5	1.4975	1.3386	0
1	9	1,3,5	1.6841	2.2328	0
1	10	2,3,5,	1.3995	2.6014	0
1	11	1,2,3,5	1.5652	3.5979	0
2	1	1	0.7352	0.2553	0
2	2	2	1.0354	0.2003	0
2	3	3	0.2865	0.9421	0
2	4	4	0.6870	0.5984	1
2	5	5	0.1662	0.9011	1
2	6	1,3	1.1996	0.7987	0
2	7	2,5	1.4869	0.9658	0
3	1	1	0.1873	0.9894	0
3	2	2	0.9092	0.5242	0
3	3	3	1.1674	0.7056	0
3	4	4	0.6311	0.7881	0
3	5	5	0.2533	0.8693	0
3	6	1,3	1.0997	1.9062	0
3	7	1,4	0.6827	1.8515	0
3	8	2,4	1.4622	1.3039	0
3	9	2,5	1.1379	1.2994	0
3	10	1,3,4	1.2349	2.6969	0

We set the termination condition to the event that all sellers bid with zero profit in all their bids. Therefore, seller 3 continues bidding although he bids with zero profit for each item.

Table 3.15 Efficient bid combinations for Round 5

Round 5							
Combination	Seller	Bid	Item	Price	Defect Rate	Buyer's True Pref. Fn. Value	Estimated Pref. Fn. Value
1	**1**	**5**	**5**	**0.7439**	**0.2019**	2.8259	2.8214
	2	**2**	**2**	**1.0354**	**0.2003**		
	2	**4**	**4**	**0.6870**	**0.5984**		
	2	**6**	**1,3**	**1.1996**	**0.7987**		
2	**2**	**2**	**2**	**1.0354**	**0.2003**	2.8228	2.8214
	2	**4**	**4**	**0.6870**	**0.5984**		
	2	**5**	**5**	**0.1662**	**0.9011**		
	2	**6**	**1,3**	**1.1996**	**0.7987**		

In Round 5, we find 30 efficient combinations using the updated bids in addition to the best combinations saved from the previous round. From these 32 combinations, two combinations are dominated, two had been previously presented, and 26 are eliminated using (ELIMINATION) procedure. The buyer states indifference between the remaining two combinations. We incorporate this information to solve (Wt). The estimated preference function values for each item and the updated bids for Round 6 are given in Tables 3.16 and 3.17, respectively.

Table 3.16 Estimated preference function values at the end of Round 5

Round 5	
Item	Estimated Pref. Fn. Value
1	0.3875
2	0.5188
3	0.4411
4	0.5822
5	0.4488

Table 3.17 Bids for Round 6

Round 6					
Seller	Bid	Item	Price	Defect Rate	Profitable
1	1	1	0.5608	0.8604	0
1	2	2	0.5681	0.9743	0
1	3	3	0.6711	0.946	0
1	4	4	1.0694	0.3096	0
1	5	5	0.6528	0.2019	1
1	6	1,3	0.9431	1.8792	0
1	7	2,5	0.6488	1.8322	0
1	8	3,5	1.4975	1.3386	0
1	9	1,3,5	1.6841	2.2328	0
1	10	2,3,5,	1.3995	2.6014	0
1	11	1,2,3,5	1.5652	3.5979	0
2	1	1	0.7352	0.2553	0
2	2	2	1.0354	0.2003	0
2	3	3	0.2865	0.9421	0
2	4	4	0.6366	0.5984	0
2	5	5	0.1627	0.9011	0
2	6	1,3	1.1996	0.7987	0
2	7	2,5	1.4869	0.9658	0
3	1	1	0.1873	0.9894	0
3	2	2	0.9092	0.5242	0
3	3	3	1.1674	0.7056	0
3	4	4	0.6311	0.7881	0
3	5	5	0.2533	0.8693	0
3	6	1,3	1.0997	1.9062	0
3	7	1,4	0.6827	1.8515	0
3	8	2,4	1.4622	1.3039	0
3	9	2,5	1.1379	1.2994	0
3	10	1,3,4	1.2349	2.6969	0

Table 3.18 Efficient bid combinations for Round 6

Round 6							
Combination	Seller	Bid	Item	Price	Defect Rate	Buyer's True Pref. Fn. Value	Estimated Pref. Fn. Value
1	**1**	**5**	**5**	**0.6528**	**0.2019**	2.7481	2.7439
	2	**2**	**2**	**1.0354**	**0.2003**		
	2	**4**	**4**	**0.6366**	**0.5984**		
	2	**6**	**1,3**	**1.1996**	**0.7987**		

In Round 6, we find 29 efficient combinations using the updated bids in addition to the two best combinations saved from the previous round. From these 31 combinations, two are dominated, five had been previously presented, and 23 are eliminated using (ELIMINATION) procedure. As the possible weight range is

narrowed considerably (lower and upper bounds of the weight of price are 0.5459 and 0.5530, respectively), the model can anticipate the best of 24 combinations without asking the buyer. Since there is only one efficient combination, this will not bring us any new information for estimation of weights. Therefore, we use estimated weight values found in Round 5 and by providing improvement for each item we estimate the preference function values given in Table 3.19.

Table 3.19 Estimated preference function values at the end of Round 6

Round 6	
Item	Estimated Pref. Fn. Value
1	0.3625
2	0.4877
3	0.4117
4	0.5295
5	0.4039

The updated bids for Round 7 are given in Table 3.20.

Table 3.20 Bids for Round 7

Round 7					
Seller	Bid	Item	Price	Defect Rate	Profitable
1	1	1	0.5608	0.8604	0
1	2	2	0.5681	0.9743	0
1	3	3	0.6711	0.946	0
1	4	4	1.0694	0.3096	0
1	5	5	0.6158	0.2019	0
1	6	1,3	0.9431	1.8792	0
1	7	2,5	0.6488	1.8322	0
1	8	3,5	1.4975	1.3386	0
1	9	1,3,5	1.6841	2.2328	0
1	10	2,3,5,	1.3995	2.6014	0
1	11	1,2,3,5	1.5652	3.5979	0
2	1	1	0.7352	0.2553	0
2	2	2	1.0354	0.2003	0
2	3	3	0.2865	0.9421	0
2	4	4	0.6366	0.5984	0
2	5	5	0.1627	0.9011	0
2	6	1,3	1.1996	0.7987	0
2	7	2,5	1.4869	0.9658	0
3	1	1	0.1873	0.9894	0
3	2	2	0.9092	0.5242	0
3	3	3	1.1674	0.7056	0
3	4	4	0.6311	0.7881	0
3	5	5	0.2533	0.8693	0
3	6	1,3	1.0997	1.9062	0
3	7	1,4	0.6827	1.8515	0
3	8	2,4	1.4622	1.3039	0
3	9	2,5	1.1379	1.2994	0
3	10	1,3,4	1.2349	2.6969	0

Table 3.21 Efficient bid combinations for Round 7

Round 7							
Combination	Seller	Bid	Item	Price	Defect Rate	Buyer's True Pref. Fn. Value	Estimated Pref. Fn. Value
1	**1**	**5**	**5**	**0.6158**	**0.2020**	2.7278	2.7237
	2	**2**	**2**	**1.0354**	**0.2003**		
	2	**4**	**4**	**0.6366**	**0.5984**		
	2	**6**	**1,3**	**1.1996**	**0.7987**		

In Round 7, we find 29 efficient combinations using the updated bids in addition to the best combination saved from the previous round. From these 30 combinations, one is dominated, 17 had been previously presented and 11 are eliminated using (ELIMINATION) procedure. We are left with a single efficient

combination. We see that all profit values are zero at this time, indicating that no seller can bid profitably. Therefore, the algorithm stops and the winners of the auction are the winners of Round 7.

Table 3.22 The results of the *ELIM-e*

Winners of the Auction (*ELIM-e*)				
Seller	Bid	Item	Price	Defect Rate
1	5	5	0.6158	0.2019
2	2	2	1.0354	0.2003
2	4	4	0.6366	0.5984
2	6	1,3	1.1996	0.7987

The true preference function value of each round is reported In Table 3.22.

Table 3.23 Buyer's preference function value in each round

Round No	Buyer's True Pref. Fn. Value	Improvement (%)
1	3.7987	-
2	3.3856	10.8748
3	3.1795	6.0875
4	2.9190	8.1931
5	2.8228	3.2956
6	2.7481	2.6463
7	2.7278	0.7387

In Table 3.24, we provide the number of combinations before and after (ELIMINATION) procedure for each round.

Table 3.24 Number of bid combinations

Round No	Number of Combinations before Elimination	Number of Combinations found previously	Elimination due to Domination	Elimination due to Weight Space Reduction
1	10	0	0	4
2	177	0	1	168
3	51	0	1	45
4	33	0	2	29
5	32	2	2	26
6	31	5	2	23
7	30	17	1	11

We also report the estimated weights and estimated bounds for the weight of the price attribute in each round in Table 3.25.

Table 3.25 Estimated bounds for the weight of the price and estimated weights in each round

Round No	Estimated LB for Weight of Price	Estimated UB for Weight of Price	Estimated Weight of Price	Estimated Weight of Defect Rate
1	0.3963	1.0000	0.7309	0.2691
2	0.3963	0.6981	0.5429	0.4571
3	0.5459	0.5530	0.5476	0.4524
4	0.5459	0.5530	0.5476	0.4524
5	0.5459	0.5530	0.5476	0.4524
6	0.5459	0.5530	0.5476	0.4524
7	0.5459	0.5530	0.5476	0.4524

To show the performance of the algorithm, we compare the results of the algorithm with the ones found using the exact parameter values. We find what the results would have been had the sellers known the buyer's true preference function explicitly, assuming, without loss of generality, that the sellers would bid zero profit as in the case of the algorithm's solution. We also assume that sellers bid independently of each other as in our experiments. Recall that, by construction, each seller has bid combinations in which he/she is competitive, based on his/her cost function. Now that we make calculations with full information (on the sellers' cost functions and buyer's preference function), we can find the optimal attribute values for all bids of all sellers in which they are competitive by solving the (MAKE_BID) problem under the zero profit assumption. Then using these bids we solve the (EFFCOM) problem and find the possible best combination(s). This case is reported as "Decentralized." The results for Decentralized case for the numerical example are provided in Table 3.26.

Table 3.26 The results for the Decentralized case

Winners of the Auction (Decentralized)							
Seller	Bid	Item	Price	Defect Rate	Combination Price	Combination Defect Rate	Buyer's True Pref. Fn. Value
1	5	5	0.6158	0.2020	3.4807	1.8075	2.7278
2	2	2	1.0354	0.2003			
2	4	4	0.6341	0.6014			
2	6	1,3	1.1954	0.8038			

When we compare the results found by the algorithm and the Decentralized case, we see that we find the same winners with the same bids as we estimate the true weights very closely. This is expected, and the estimations would keep improving with the amount of preference information and converge to the true weights with sufficient information.

We also compare the preference function values of the buyer for the winning bidders found with the algorithm and in the Decentralized case. We check the percent deviations of the algorithm from Decentralized using the following formula:

$$\%\,\text{deviation} = \frac{u(final\_combination\_algorithm) - u(Decentralized)}{u(Decentralized)} 100$$

where $u(final\_combination\_algorithm)$ and $u(Decentralized)$ refer to the preference function value s of the final bid combination found by the *ELIM-e* and Decentralized, respectively.

We use 10 different weight values for the price attribute (λ) to generate different problems for both linear and nonlinear cases. The percentage deviations for the linear case are reported in Table 3.27.

Table 3.27 Percentage deviations between the results of *ELIM-e* and the decentralized optimal solution

λ=0.05	λ=0.15	λ=0.25	λ=0.35	λ=0.45	λ=0.55	λ=0.65	λ=0.75	λ=0.85	λ=0.95
0.0084	0.0064	0.0000	0.0000	0.0040	0.0000	0.0075	0.0043	0.0000	0.0245

In all problems the winning bidders found by the algorithm and Decentralized are the same, i.e. allocative efficiency is satisfied. As can be seen from Table 3.27, the percentage deviations are very small, i.e. for all problems the buyer's preference function found with the algorithm is close to that found by Decentralized. Moreover, with (ELIMINATION) procedure we substantially decrease the number of alternatives presented to the buyer. These imply that the estimation and guidance mechanisms of our approach worked well in all the test problems.

3.9 The 3-attribute Case

We consider *lead time* as the third attribute. In this case, we construct the cost function in such a way that improvements in the defect rate and lead time both increase the cost in different magnitudes. The relation of the defect rate with cost is the same as that of the two attribute case. For the three attribute case, we generate a convex cost function and apply a procedure similar to MAKE_BID procedure to determine the updated bids. For the sake of completeness, we provide the whole procedure below:

Parameters:

w_p: estimated weight of price where $w_p > 0$

w_d: estimated weight of defect rate where w_d>0

w_l: estimated weight of lead time where w_l>0

EP_t: estimated preference function value for bid *t* (ADSS provides the estimated preference function value for each item separately and the estimated preference function of a bundle is calculated by adding up the estimated preference function values of the items in the bundle)

v_i^{perc}: minimum mark-up percentage for seller *i*; if it is 0, then seller *i* may bid with zero profit. For the sake of simplicity let $v_i = v_i^{perc}/100$.

f_{it}: the cost function of seller *i* for bid *t*

$$f_{it}(d_{it}, l_{it}) = \left(1 - \left(1 - \left(1 - \frac{d_{it} - D_{it}^-}{D_{it}^+ - D_{it}^-}\right)^{q_i}\right)^{1/q_i}\right)(C_{it}^+ - C_{it}^-) + C_{it}^- + C_{it}^- \frac{(L_{it}^+ - L_{it}^-)}{K(l_{it} - L_{it}^-)}$$

where

D_{it}^+: maximum defect rate value that can be offered by seller *i* for bid *t*

D_{it}^-: minimum defect rate value that can be offered by seller *i* for bid *t*

C_{it}^+: maximum cost of bid *t* for seller *i* (at D_{it}^-)

C_{it}^-: minimum cost of bid *t* for seller *i* (at D_{it}^+)

L_{it}^+: maximum lead time value that can be offered by seller *i* for bid *t*

L_{it}^-: minimum lead time value that can be offered by seller *i* for bid *t*

q_i: parameter of the L_q cost function for seller *i*

K: positive constant

Decision Variables:

p_{it} = price offered by seller *i* for bid *t*

d_{it} = defect rate offered by seller *i* for bid *t*

l_{it} = lead time value offered by seller *i* for bid *t*

Each seller tries to maximize his/her profit by solving the following problem.

Problem (MAKE_BID$_{it}$)

Max $p_{it} - (1 + v_i) f_{it}(d_{it}, l_{it})$ (5.1)

s.to

$D_{it}^- \leq d_{it} \leq D_{it}^+$ (5.2)

$L_{it}^- \leq l_{it} \leq L_{it}^+$ (5.3)

$w_p . p_{it} + w_d . d_{it} + w_l . l_{it} \leq EP_t$ (5.4)

$w_p + w_d + w_l = 1$ (5.5)

The objective function is equivalent to

$$\text{Max } z(d_{it}, l_{it}) = \frac{EP_t - w_d d_{it} - w_l l_{it}}{w_p} - (1 + v_i) f_{it}(d_{it}, l_{it}).$$

Similar to Section 3.4, we conclude that $z(d_{it}, l_{it})$ is concave and we utilize the properties of concave functions and determine the updated bids as follows:

The objective function can be rewritten as follows:

$$\text{Max } z(d_{it}, l_{it}) = \frac{EP_t - w_d d_{it} - w_l l_{it}}{w_p} - (1 + v_i)\left(1 - \left(1 - \left(1 - \frac{d_{it} - D_{it}^-}{D_{it}^+ - D_{it}^-}\right)^{q_i}\right)^{\frac{1}{q_i}}\right)(C_{it}^+ - C_{it}^-) - (1 + v_i)C_{it}^- - (1 + v_i)C_{it}^- \frac{(L_{it}^+ - L_{it}^-)(l_{it} - L_{it}^-)^{-1}}{K}$$

Since $z(d_{it}^*, l_{it}^*)$ is concave, to find the optimal defect rate and lead time values we check the stationary and boundary points.

At stationary point: $\nabla z = \mathbf{0}$

$$\nabla z = \begin{pmatrix} -\frac{w_d}{w_p} + \frac{(1 + v_i)(C_{it}^+ - C_{it}^-)}{D_{it}^+ - D_{it}^-}(D_{it}^+ - d_{it})^{q_i - 1}((D_{it}^+ - D_{it}^-)^{q_i} - (D_{it}^+ - d_{it})^{q_i})^{(1 - q_i)/q_i} \\ -\frac{w_l}{w_p} + (1 + v_i)C_{it}^- \frac{(L_{it}^+ - L_{it}^-)(l_{it} - L_{it}^-)^{-2}}{K} \end{pmatrix} = \mathbf{0}$$

After some manipulations, we obtain

$$d_{it}^* = D_{it}^+ - \frac{(1 - w_d)^{1/(q_i - 1)}(D_{it}^+ - D_{it}^-)^{q_i/(q_i - 1)}}{\left(\left(w_d(D_{it}^+ - D_{it}^-)\right)^{q_i/(q_i - 1)} + \left(w_p(C_{it}^+ - C_{it}^-)(1 + v_i)\right)^{q_i/(q_i - 1)}\right)^{1/q_i}}$$

$$l_{it}^* = L_{it}^- + \left(\frac{w_p(1 + v_i)C_{it}^-(L_{it}^+ - L_{it}^-)}{K w_l}\right)^{1/2}$$

d_{it}^* is same with that of in 2-attribute case. For l_{it}^*, we also check the upper bound.

If $\frac{EP_t - w_d d_{it}^* - w_l l_{it}^*}{w_p} > (1 + v_i) f_{it}(d_{it}^*, l_{it}^*)$, $p_{it}^* = \frac{EP_t - w_d d_{it}^* - w_l l_{it}^*}{w_p}$ and seller bids profitably. Otherwise, $p_{it}^* = (1 + v_i) f_{it}(d_{it}^*, l_{it}^*)$ and seller bids with his/her predetermined mark-up only and no extra profit is possible.

All the steps are the same as the 2-attribute case except that to find the efficient solutions for 3-attribute problems we use the algorithm suggested by Lokman and Köksalan (2012).

We set *K=10*, and set minimum and maximum lead time values for each item to 1 and 2, respectively. We calculate the lead time value of a bundle by summing up the lead time values of the items that constitute the combination. Other parameters take the same values as in the 2-attribute case. We use 10 different weight values for the attributes to generate different 3-attribute problems. We use *ELIM-e* and report the results for 7 problems in Tables 3.28 and 3.29.

Table 3.28 Preference function values of the combinations found by *ELIM-e* and the decentralized optimal solution (3-attribute case)

	λ_p=0.1 λ_d=0.1 λ_l=0.8	λ_p=0.1 λ_d=0.8 λ_l=0.1	λ_p=0.8 λ_d=0.1 λ_l=0.1	λ_p=0.1 λ_d=0.45 λ_l=0.45	λ_p=0.45 λ_d=0.1 λ_l=0.45	λ_p=0.45 λ_d=0.45 λ_l=0.1	λ_p=0.33 λ_d=0.33 λ_l=0.33
ELIM-e	5.0957	2.0285	2.6424	3.6972	4.1667	3.3715	4.2205
Decentralized	5.094	2.0284	2.6422	3.6953	4.1667	3.371	4.2038

Table 3.29 Percentage deviations between the results of *ELIM-e* and the decentralized optimal solution (3-attribute case)

λ_p=0.1 λ_d=0.1 λ_l=0.8	λ_p=0.1 λ_d=0.8 λ_l=0.1	λ_p=0.8 λ_d=0.1 λ_l=0.1	λ_p=0.1 λ_d=0.45 λ_l=0.45	λ_p=0.45 λ_d=0.1 λ_l=0.45	λ_p=0.45 λ_d=0.45 λ_l=0.1	λ_p=0.33 λ_d=0.33 λ_l=0.33
0.0334	0.0049	0.0076	0.0514	0.0000	0.0148	0.3973

In all problems, the winning bidders found by the algorithm and Decentralized are the same. The largest percentage deviation of the preference values obtained by the winning bidders of the algorithm from that of Decentralized is 0.3973%. Therefore, we say that our algorithm works well in all the problems solved for the 3-attribute case.

3.10 Local Linear Approximation for Nonlinear Preference Functions

So far, we tested the algorithm for underlying linear preference functions. In this section, we assume that the buyer has an underlying decreasing quasiconvex preference function. We locally approximate the buyer's preference function with a linear function. We estimate the weight values using a variation of (Wt) model. If the model is feasible, we use the weights that solve (Wt). However, if there are no weights satisfying the constraints, we relax the constraints that cause infeasibility and solve the problem again. There are many ways of choosing which constraints to remove (see for example Chinneck 2008). In our infeasibility reduction heuristic, we solve the following (IR) problem to identify a set of constraints causing infeasibility.

Recall that $\boldsymbol{e}_k$ represents the k^{th} efficient combination presented to the buyer for the current round. $u(\boldsymbol{e}_k)$ denotes the estimated preference function value of the buyer for $\boldsymbol{e}_k$ and e_{kj} denotes the level of attribute j of $\boldsymbol{e}_k$.

Parameters:

Δ: predetermined threshold level by which the buyer can distinguish between bid combinations

ρ: a small positive constant

e_{kj}: level of attribute j in efficient combination k

Decision Variables:

z: maximum amount of infeasibility

$\widehat{w}$: estimated weight of attribute 1

g_{pn}: amount of infeasibility in the preference constraint for each $\boldsymbol{e}_p$ preferred to each $\boldsymbol{e}_n$

g_{lr}, g_{rl}: amount of infeasibility in the preference constraint for each $\boldsymbol{e}_l$ and $\boldsymbol{e}_r$ the buyer is indifferent between

Model (IR)

$$\text{Min } z + \rho \sum_{\boldsymbol{e}_{p} \succ \boldsymbol{e}_{n}, \boldsymbol{e}_{l} \sim \boldsymbol{e}_{r}} \left(g_{pn} + g_{lr} + g_{rl} \right) \tag{6.1}$$

s.to

$$u(\boldsymbol{e}_k) = \hat{w} e_{k1} + (1 - \hat{w}) e_{k2} \qquad k \in E \tag{6.2}$$

$$u(\boldsymbol{e}_n) \geq u(\boldsymbol{e}_p) + \Delta - g_{pn} \qquad \text{for each } \boldsymbol{e}_p \succ \boldsymbol{e}_n \tag{6.3}$$

$$u(\boldsymbol{e}_l) - \Delta + \rho - g_{lr} \leq u(\boldsymbol{e}_r) \leq u(\boldsymbol{e}_l) + \Delta - \rho + g_{rl} \qquad \text{for each } \boldsymbol{e}_l \sim \boldsymbol{e}_r \tag{6.4}$$

$$z \geq g_{pn} \qquad \text{for each } \boldsymbol{e}_p \succ \boldsymbol{e}_n \tag{6.5}$$

$$z \geq g_{lr} \qquad \text{for each } \boldsymbol{e}_l \sim \boldsymbol{e}_r \tag{6.6}$$

$$z \geq g_{rl} \qquad \text{for each } \boldsymbol{e}_l \sim \boldsymbol{e}_r \tag{6.7}$$

$$g_{pn}, g_{lr}, g_{rl} \geq 0 \qquad \text{for each } \boldsymbol{e}_p \succ \boldsymbol{e}_n, \boldsymbol{e}_l \sim \boldsymbol{e}_r \tag{6.8}$$

$$\hat{w} \geq 0 \tag{6.9}$$

In the original (Wt) model, we assume that all constraints are feasible and we try to maximize μ. In the (IR) model we remove variable μ and add nonnegative g_{pn}, g_{lr} and g_{rl} for each $\boldsymbol{e}_p \succ \boldsymbol{e}_n, \boldsymbol{e}_l \sim \boldsymbol{e}_r$. We try to minimize the maximum contribution to infeasibility. We also use an augmented part that is the sum of infeasibility contributions to break the ties. The (IR) model is always feasible as for each preference constraint there is a variable that captures the amount of violation, while keeping the constraint feasible. If the objective is strictly positive, the constraints with strictly positive $g_{pn}, g_{lr},$ or g_{rl} contribute to infeasibility. Instead of deleting these constraints, we allow them to be violated while minimizing the violation of the constraint that has the maximum violation.

In the nonlinear case, we apply ELIMINATION procedure to eliminate some of the efficient combinations in the initial phases since we assume linearity for the buyer's preference function. We stop using ELIMINATION when the (IR) problem has a positive objective function value. The reason is that when the (IR) problem has a positive objective function value, it is discovered that the linearity assumption for the buyer's preference function is violated. Continuing with ELIMINATION can lead to the elimination of some of the efficient combinations that might be preferred by the buyer. Therefore, until the (IR) model has a positive objective function value, we apply ELIMINATION but afterwards we

present the buyer all the efficient combinations of the succeeding rounds without applying ELIMINATION. In this approach, as we apply ELIMINATION procedure at the beginning of the rounds, we may eliminate good solutions before observing the violation of the linearity assumption. To avoid such deficiencies some repairing methods can be used. For instance, when the linearity assumption failed, the alternatives eliminated at the beginning of the corresponding round may also be presented to the buyer.

We test the performance of the algorithm by simulating the preferences of the buyer having an L_α preference function; specifically we use the weighted Euclidean ($\alpha = 2$) and the weighted Tchebycheff ($\alpha = \infty$) functions.

We use 10 different weight values for the attributes to generate different 2-attribute problems. The results are given in Table 3.30.

Table 3.30 Percentage deviations between the results of the algorithm and decentralized optimal solution under weighted Euclidean preference function

λ=0.05	λ=0.15	λ=0.25	λ=0.35	λ=0.45	λ=0.55	λ=0.65	λ=0.75	λ=0.85	λ=0.95
0.0306	0.0090	-0.0869	0.2911	-0.3593	-1.0399	-0.0052	0.1155	0.0208	0.0000

Table 3.31 Percentage deviations between the results of the algorithm and decentralized optimal solution under weighted Tchebycheff preference function

λ=0.05	λ=0.15	λ=0.25	λ=0.35	λ=0.45	λ=0.55	λ=0.65	λ=0.75	λ=0.85	λ=0.95
0.0421	8.2693	-3.3448	-2.0733	-0.4471	-0.1517	-4.8684	6.1857	0.0000	0.0000

When we look at the results in Table 3.30 the percent deviations are very small. In Table 3.31, however, the highest percent deviation is 8.2693. When we check the reason for such a high percent deviation, we see that between rounds we eliminate the best of the previous round as we apply ELIMINATION procedure until discovering that the underlying preference function is not linear. As suggested before some repairing operations may be applied. Even in this case, the average percent deviation is 0.3612. Moreover, in both tables we see in some test

problems that our algorithm performed better than Decentralized. We conclude that the estimation and guidance mechanisms of our approach worked well when the buyer has a nonlinear preference function. Although Decentralized finds better preference function values for each bid separately, when a combination is constructed, the preference function of a combination for the Decentralized case may be worse than that of ours for the considered nonlinear preference functions. This result is not surprising when the underlying preference function is nonlinear.

We demonstrate this situation with a simple example with two bids. Suppose the buyer has a weighted Euclidean preference function with equal weights for price and defect rate. Consider the attribute values of the bids presented in Table 3.32.

Table 3.32 Attribute values for each bid separately

	Decentralized			Algorithm		
	price	defect rate	Buyer's preference function value	price	defect rate	Buyer's preference function value
Item 1	1.0	2.0	1.1180	1.7	1.5	1.1336
Item 2	1.0	2.0	1.1180	1.2	1.9	1.1236
Combination	2.0	4.0	1.2361	2.9	3.4	1.2344

In Table 3.32, we see that Decentralized finds better preference function values for each bid separately. However, when a combination is constructed, the preference function of a combination for the Decentralized case is worse than that of the algorithm.

CHAPTER 4

AN INTERACTIVE METHOD TO FIND THE BEST BID COMBINATION

In Chapter 3, we used ELIMINATION procedure to decrease the number of efficient combinations presented to the buyer. However, even after applying this procedure, the number of remaining efficient combinations may be high and it might be difficult for the buyer to evaluate them all. Therefore, we apply a multi-criteria decision making (MCDM)-based method to support the buyer to find the best bid combination among the given bid combinations. In this chapter we develop an interactive algorithm for the case where the underlying preference function is linear. We first give an overview of the approach. We then explain the algorithm and discuss the results. Lastly, we present a heuristic approach for underlying nonlinear preference functions.

4.1 The Interactive Approach

The stages of the approach are similar to those in Section 3.1. The main difference is that in *ELIM-e*, in each round we first find a set of efficient combinations and then ask the buyer to compare them; whereas in this method these processes are not sequential but interactive. For the sake of completeness, we provide the stages of the new interactive approach in Figure 4.1.

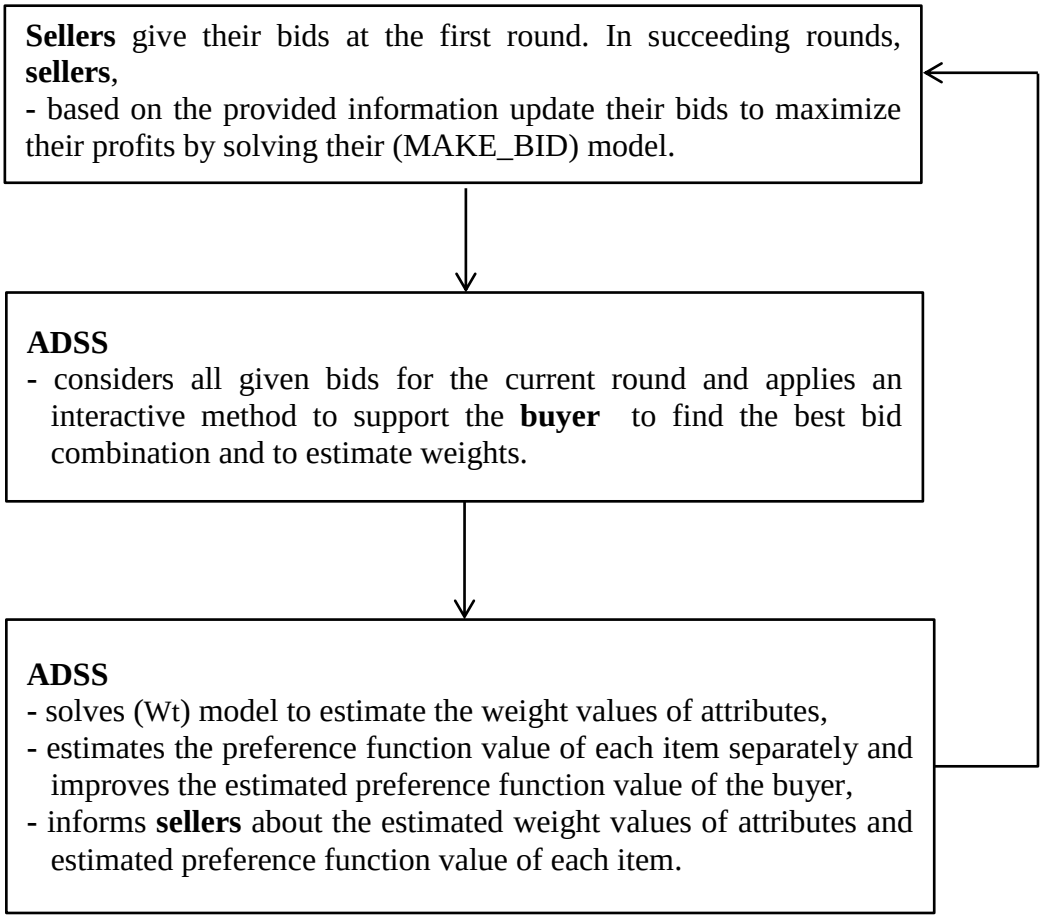


Figure 4.1 The stages of the interactive approach

4.2 An Interactive Algorithm (*LIN-u*)

We introduce an interactive algorithm, *LIN-u*, to find the most preferred combination of a buyer for the 2-attribute case. We assume an underlying linear preference function and apply a variation of the algorithm developed by Zionts (1981). Since we assume a linear preference function, the most preferred solution of the buyer is a supported efficient solution. Thus we deal with only the extreme supported efficient solutions. In the algorithm, the buyer compares an incumbent efficient solution with its adjacent efficient alternatives. We reduce the weight space based on the preferences of the buyer. If an adjacent efficient alternative is preferred to the current incumbent, a new incumbent is generated and the algorithm continues until an incumbent is preferred to all its adjacent efficient alternatives. Since we assume an underlying linear preference function, the

resulting solution at the end of the algorithm is the best solution of the current round (see Zionts 1981). Zionts (1981) does not consider the case where the buyer expresses indifference between two alternatives. In our algorithm, we consider such cases as well. Moreover, unlike Zionts (1981), we do not have all solutions at hand at the outset but we rather generate each solution to be presented to the buyer as needed.

We note that it is also possible to find all efficient combinations first and then apply the interactive method.

LIN-u algorithm is a generalization of Zionts' method and is applicable for the general bi-objective integer programming problems. As mentioned before, we use the terms DM and buyer interchangeably. The algorithm for the single-round linear case follows:

LIN-u

In our indifference relations we do not assume transitivity; that is, if the DM indicates indifference between alternatives *A* and *B* as well as between alternatives *B* and *C*, we do not automatically assume indifference between *A* and *C*. However, we construct a set, *IN*, that contains indifferent alternatives based on direct comparisons and transitivity relations. Let $\boldsymbol{e}^1$ and $\boldsymbol{e}^2$ be the alternatives having the maximum values in attributes 1 and 2 among the alternatives in *IN*, respectively. These alternatives correspond to the extremes of set *IN* in the two attributes. Since an alternative can have at most two distinct adjacent efficient alternatives in the two attribute case, we classify them as the east and west adjacent efficient alternatives. Let $\boldsymbol{e}_E$ and $\boldsymbol{e}_W$ be the east and west adjacent efficient alternatives to the incumbent, respectively. Select an arbitrary direction (*DIR*) to search an adjacent efficient alternative to the incumbent. Let *DIR=east* correspond to searching for $\boldsymbol{e}_E$ and *DIR=west* correspond to searching for $\boldsymbol{e}_W$. Set iteration counter $i = 1$. Without loss of generality set DIR=*east* and $\boldsymbol{e}^1 = \boldsymbol{e}^2 = \boldsymbol{0}$. Let e_{k1} and e_{k2} be the values of attributes 1 and 2 of the corresponding

solution, $\boldsymbol{e}_k$, respectively, $\boldsymbol{w} = (w, 1-w)$ where w is the estimated weight of attribute 1 and ρ be a small positive constant. Recall that WL and WU are the estimated lower and upper bounds for the weight of attribute 1, respectively.

In the algorithm, after the DM expresses his/her preferences, we construct the corresponding preference constraint(s) which constitute the feasible weight space of the (Wt) model explained in Section 3.3.

We first discuss the details of various aspects of the algorithm and then give the steps.

At the beginning, we select arbitrary weights to find an alternative. In the auction example, at the beginning of Round 1, we assume equal weights for attributes since we have no information. We solve the following (ALT) model to find a new alternative with the estimated weights:

Parameters:

a_{itj}: level of attribute *j* offered by seller *i* in bid *t*

π_{itm}: 1 if bid *t* of seller *i* includes item *m*; 0 otherwise

T_i: the number of bids offered by seller *i*

w: estimated weight of attribute 1

Decision Variables:

y_{it}: 1 if bid *t* of seller *i* is selected to be in the efficient combination; 0 otherwise

$\hat{e}_j$: level of attribute *j* of the optimal alternative

Model (ALT)

$$\text{Min } w\hat{e}_1 + (1-w)\hat{e}_2 \qquad (7.1)$$

s.to

$$\sum_{i=1}^{I}\sum_{t=1}^{T_i} \pi_{itm} y_{it} \geq 1 \qquad \forall m \qquad (7.2)$$

$$\hat{e}_j = \sum_{i=1}^{I}\sum_{t=1}^{T_i} a_{itj} y_{it} \qquad j = 1,2 \qquad (7.3)$$

$$y_{it} \in \{0,1\} \qquad (7.4)$$

In (ALT), we aim to minimize the estimated preference function value by using the estimated weights. Constraint set (7.2) guarantees satisfying the demand for each item.

After finding an incumbent, we then find the adjacent efficient alternative of the incumbent based on the search direction. We adapt the method proposed by Aneja and Nair (1979). They find all supported efficient solutions of a bi-objective problem by minimizing a linear function of the two objectives whose weights they systematically change. We next explain the procedure to find east adjacent efficient alternative of an incumbent. The procedure for finding the west adjacent efficient alternative is similar.

Finding east adjacent efficient alternative (ADJ_E)

Recall that $\boldsymbol{e}^*$, $\boldsymbol{e}_E$ and $\boldsymbol{e}_W$ be the incumbent, east and west adjacent efficient alternatives, respectively, and WL and WU be the estimated lower and upper bounds for estimated weight of attribute 1, respectively. Let $\boldsymbol{e}^{EM}$ be the eastmost alternative having the maximum price value that bounds the search region. If $\boldsymbol{e}^{EM}$ has not been found previously, set $w = \varepsilon$ where ε is a small positive constant and solve (ALT). Set the solution of (ALT) to $\boldsymbol{e}^{EM}$. Afterwards, as the DM expresses his/her preferences the value of $\boldsymbol{e}^{EM}$ will be updated if necessary. For instance if the DM prefers $\boldsymbol{e}_W$ to $\boldsymbol{e}^*$, then set $\boldsymbol{e}^{EM} = \boldsymbol{e}^*$.

We note that if at the beginning $\boldsymbol{e}^{EM} = \boldsymbol{e}^*$, it indicates that there is no east adjacent efficient alternative of $\boldsymbol{e}^*$. In this case, we skip the following steps.

Step 1: Set $w = \frac{e_2^* - e_2^{EM}}{e_1^{EM} - e_2^{EM} - e_1^* + e_2^*}$. Solve (ALT) using w and considering the following additional constraints to bound the search region:

$$\hat{e}_1 \geq e_1^* + \rho \quad (7.5)$$

$$\hat{e}_2 \geq e_2^{EM} + \rho \quad (7.6)$$

$$w\hat{e}_1 + (1-w)\hat{e}_2 + s = we_1^* + (1-w)e_2^* \quad (7.7)$$

where ρ is a small positive constant and s is the slack variable of the corresponding constraint. If there is a feasible solution, $\hat{\boldsymbol{e}}$, having a positive s value, set $\boldsymbol{e}^{EM} = \hat{\boldsymbol{e}}$ and go to *Step 1*. Otherwise go to *Step 2*.

Step 2: Set $\boldsymbol{e}_E = \boldsymbol{e}^{EM}$. Stop.

In Step 1, we calculate the weights of the linear function passing through the incumbent and the eastmost alternative. Then we find the solution that minimizes the estimated preference function with the corresponding weights applying (ALT) and additional constraints. Although (ALT) itself is always feasible, with the additional constraints we cannot guarantee to find feasible solutions. Since both $\boldsymbol{e}^*$ and $\boldsymbol{e}^{EM}$ are supported efficient solutions, we use ρ and bound the search region with constraints (7.5) and (7.6). Although these two constraints are redundant, to emphasize the search region we keep them in the model. In (7.7) we check the slack variable value. If it is zero or negative, then the solution, $\hat{\boldsymbol{e}}$, is convex dominated by the points on the line passing through the incumbent and $\boldsymbol{e}^{EM}$. We continue to search if there is a feasible solution with a positive slack variable value.

After finding an adjacent alternative we check whether the adjacent alternative can be preferred to the incumbent based on past preference information. To do this, we check whether the weights making the adjacent alternative preferred to the incumbent are in the feasible weight region or not. If the weights favoring the adjacent alternative are in the feasible weight space we ask the DM to compare the incumbent and the adjacent alternative. Otherwise, we conclude that the adjacent alternative cannot be preferred by the DM based on his/her past preferences due to Theorem 1 in Chapter 3. In the algorithm, we find the

incumbent using the estimated (feasible) weights. However, we do not know whether the weights favoring the adjacent alternative are in the feasible weight space or not. Therefore, we check w^{temp} which refers to the weight of attribute 1 that makes $\boldsymbol{e}_{adj}$ equivalent to $\boldsymbol{e}^*$. If the direction is *west* and $w^{temp} + \rho$ is greater than the upper bound of the weight of attribute 1 or if the direction is *east* and $w^{temp} - \rho$ is smaller than the lower bound of the weight of attribute 1, we say that there does not exist weights in the feasible weight space that make the adjacent alternative preferable to the incumbent.

We next provide the steps of the algorithm *LIN-u*.

Step 1: Select an arbitrary set of weights, find an incumbent, $\boldsymbol{e}^*$, and go to *Step 3*.
Step 2: Set $i \leftarrow i + 1$. Find a solution, $\boldsymbol{e}_i$, using $\boldsymbol{w}$. If $\boldsymbol{e}^*, \boldsymbol{e}_i \in IN$ or $\boldsymbol{e}_i = \boldsymbol{e}^*$, go to *Step 3*. Else, if $IN = \emptyset$, go to *Step 2.1*. Else go to *Step 2.2*.

Step 2.1: If $e_{i1} < e_1^*$, set *DIR=west*; otherwise set *DIR=east*. Ask the DM $\boldsymbol{e}^*$ versus $\boldsymbol{e}_i$. If

- $\boldsymbol{e}_i$ is preferred to $\boldsymbol{e}^*$, add a constraint $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_i) \geq \Delta + \mu$. Set $\boldsymbol{e}^* = \boldsymbol{e}_i$ and go to *Step3*.
- $\boldsymbol{e}^*$ is preferred to $\boldsymbol{e}_i$, add a constraint $\boldsymbol{w}(\boldsymbol{e}_i - \boldsymbol{e}^*) \geq \Delta + \mu$. Switch the value of DIR (i.e. set *DIR=east* if it is equal to *west* and set *DIR=west* if it is equal to *east*) and go to *Step 3*.
- the DM is indifferent between $\boldsymbol{e}^*$ and $\boldsymbol{e}_i$, add constraints $\boldsymbol{w}(\boldsymbol{e}_i - \boldsymbol{e}^*) \leq \Delta - \rho - \mu$ and $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_i) \leq \Delta - \rho - \mu$. Set $IN = \{\boldsymbol{e}^*, \boldsymbol{e}_i\}$, update $\boldsymbol{e}^1, \boldsymbol{e}^2$ and go to *Step 3*.

Step 2.2: If $e_{i1} < e_1^*$, set $\boldsymbol{e}^* = \boldsymbol{e}^1$ and *DIR=west*. Else set $\boldsymbol{e}^* = \boldsymbol{e}^2$ and *DIR=east*.

Ask the DM $\boldsymbol{e}^*$ versus $\boldsymbol{e}_i$. If

- $\boldsymbol{e}_i$ is preferred to $\boldsymbol{e}^*$, add a constraint $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_i) \geq \Delta + \mu$. Set $IN = IN - \{\boldsymbol{e}^*\}$. If $\boldsymbol{e}^* = \boldsymbol{e}^1$, set $\boldsymbol{e}^* = \boldsymbol{e}^2$; otherwise set $\boldsymbol{e}^* = \boldsymbol{e}^1$. Ask the DM $\boldsymbol{e}^*$ versus $\boldsymbol{e}_i$. If

- $\boldsymbol{e}_i$ is preferred to $\boldsymbol{e}^*$, add a constraint $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_i) \geq \Delta + \mu$. Set $\boldsymbol{e}^* = \boldsymbol{e}_i$, $IN = \emptyset$, $\boldsymbol{e}^1 = \boldsymbol{e}^2 = \boldsymbol{0}$ and go to *Step 3*.
- the DM is indifferent between $\boldsymbol{e}^*$ and $\boldsymbol{e}_i$, add constraints $\boldsymbol{w}(\boldsymbol{e}_i - \boldsymbol{e}^*) \leq \Delta - \rho - \mu$ and $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_i) \leq \Delta - \rho - \mu$. Set $IN = IN \cup \{\boldsymbol{e}_i\}$, check $\boldsymbol{e}^1/\boldsymbol{e}^2$ (i.e. if $\boldsymbol{e}^* = \boldsymbol{e}^1$ update $\boldsymbol{e}^2$, otherwise update $\boldsymbol{e}^1$ if necessary **)** and go to *Step 3*.

- $\boldsymbol{e}^*$ is preferred to $\boldsymbol{e}_i$, add a constraint $\boldsymbol{w}(\boldsymbol{e}_i - \boldsymbol{e}^*) \geq \Delta + \mu$. Switch the value of *DIR*.
- the DM is indifferent between $\boldsymbol{e}^*$ and $\boldsymbol{e}_i$, add constraints $\boldsymbol{w}(\boldsymbol{e}_i - \boldsymbol{e}^*) \leq \Delta - \rho - \mu$ and $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_i) \leq \Delta - \rho - \mu$. Set $IN = IN \cup \{\boldsymbol{e}_i\}$ and check $\boldsymbol{e}^1/\boldsymbol{e}^2$.

Step 3: If *DIR=east*, find $\boldsymbol{e}_E$ and set $\boldsymbol{e}_{adj} = \boldsymbol{e}_E$; otherwise find $\boldsymbol{e}_W$ and set $\boldsymbol{e}_{adj} = \boldsymbol{e}_W$. If there does not exist $\boldsymbol{e}_{adj}$ or $(\boldsymbol{e}^*, \boldsymbol{e}_{adj})$ pair has been compared previously, go to *Step 3.1*. Otherwise, set $w^{temp} = \frac{e_2^* - e_{adj,2}}{e_{adj,1} - e_{adj,2} - e_1^* + e_2^*}$ and if *DIR=east* go to *Step 3.2;* otherwise go to *Step 3.3.*

Step 3.1: If both $\boldsymbol{e}_E$ and $\boldsymbol{e}_W$ have been evaluated before, go to *Step 5*. Otherwise, switch the value of *DIR* and go to *Step 3*.

Step 3.2: If $w^{temp} - \rho < WL$, go to *Step 3.1*; otherwise go to *Step 4*.

Step 3.3: If $w^{temp} + \rho > WU$, go to *Step 3.1*; otherwise go to *Step 4*.

Step 4: If $IN = \emptyset$, go to *Step 4.1*.Else go to *Step 4.2*.

Step 4.1: Ask the DM $\boldsymbol{e}^*$ versus $\boldsymbol{e}_{adj}$. If

- $\boldsymbol{e}_{adj}$ is preferred to $\boldsymbol{e}^*$, add a constraint $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_{adj}) \geq \Delta + \mu$. Set $\boldsymbol{e}^* = \boldsymbol{e}_{adj}$ and go to *Step 6*.
- $\boldsymbol{e}^*$ is preferred to $\boldsymbol{e}_{adj}$, add a constraint $\boldsymbol{w}(\boldsymbol{e}_{adj} - \boldsymbol{e}^*) \geq \Delta + \mu$. If both $\boldsymbol{e}_E$ and $\boldsymbol{e}_W$ have been evaluated go to *Step 5*. Otherwise switch the value of *DIR* and go to *Step 3*.
- the DM is indifferent between $\boldsymbol{e}^*$ and $\boldsymbol{e}_{adj}$, add constraints $\boldsymbol{w}(\boldsymbol{e}_{adj} - \boldsymbol{e}^*) \leq \Delta - \rho - \mu$ and $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_{adj}) \leq \Delta - \rho - \mu$. Set $IN = \{\boldsymbol{e}^*, \boldsymbol{e}_{adj}\}$ and update $\boldsymbol{e}^1, \boldsymbol{e}^2$. If both $\boldsymbol{e}_E$ and $\boldsymbol{e}_W$ have been evaluated go to *Step 5*. Otherwise switch the value of *DIR* and go to *Step 3*.

Step 4.2: If *DIR=west*, set $\boldsymbol{e}^* = \boldsymbol{e}^1$; otherwise set $\boldsymbol{e}^* = \boldsymbol{e}^2$. Ask the DM $\boldsymbol{e}^*$ versus $\boldsymbol{e}_{adj}$. If

- $\boldsymbol{e}_{adj}$ is preferred to $\boldsymbol{e}^*$, add a constraint $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_{adj}) \geq \Delta + \mu$. Set $IN = IN - \{\boldsymbol{e}^*\}$, if *DIR=west*, set $\boldsymbol{e}^* = \boldsymbol{e}^2$; otherwise set $\boldsymbol{e}^* = \boldsymbol{e}^1$. Ask the DM $\boldsymbol{e}^*$ versus $\boldsymbol{e}_{adj}$. If
    - $\boldsymbol{e}_{adj}$ is preferred to $\boldsymbol{e}^*$, add a constraint $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_{adj}) \geq \Delta + \mu$. Set $\boldsymbol{e}^* = \boldsymbol{e}_{adj}$, $IN = \emptyset$, $\boldsymbol{e}^1 = \boldsymbol{e}^2 = \boldsymbol{0}$ and go to *Step 6.*
    - the DM is indifferent between $\boldsymbol{e}^*$ and $\boldsymbol{e}_{adj}$, add constraints $\boldsymbol{w}(\boldsymbol{e}_{adj} - \boldsymbol{e}^*) \leq \Delta - \rho - \mu$ and $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_{adj}) \leq \Delta - \rho - \mu$. Set $IN = IN \cup \{\boldsymbol{e}_{adj}\}$ and check $\boldsymbol{e}^1/\boldsymbol{e}^2$. If both $\boldsymbol{e}_E$ and $\boldsymbol{e}_W$ have been evaluated go to *Step 5.* Otherwise switch the value of *DIR* and go to *Step 3*.
- $\boldsymbol{e}^*$ is preferred to $\boldsymbol{e}_{adj}$, add a constraint $\boldsymbol{w}(\boldsymbol{e}_{adj} - \boldsymbol{e}^*) \geq \Delta + \mu$. If both $\boldsymbol{e}_E$ and $\boldsymbol{e}_W$ have been evaluated go to *Step 5.* Otherwise switch the value of *DIR* and go to *Step 3*.
- the DM is indifferent between $\boldsymbol{e}^*$ and $\boldsymbol{e}_{adj}$, add constraints $\boldsymbol{w}(\boldsymbol{e}_{adj} - \boldsymbol{e}^*) \leq \Delta - \rho - \mu$ and $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_{adj}) \leq \Delta - \rho - \mu$. Set $IN = IN \cup \{\boldsymbol{e}_{adj}\}$ and check $\boldsymbol{e}^1/\boldsymbol{e}^2$. If both $\boldsymbol{e}_E$ and $\boldsymbol{e}_W$ have been evaluated go to *Step 5*. Otherwise switch the value of *DIR* and go to *Step 3*.

Step 5: If any of the following three conditions are satisfied, go to *Step 7*. Otherwise go to *Step 6*.

- incumbent has no adjacent efficient alternative
- $IN = \emptyset$ and $\boldsymbol{e}^*$ is preferred to all its adjacent efficient alternatives
- $\boldsymbol{e}^* \in IN$ and both $\boldsymbol{e}^1$ and $\boldsymbol{e}^2$ are at least as preferable as their adjacent efficient alternatives

Step 6: Find a feasible set of weights satisfying all constraints corresponding to past preferences of the DM by solving the (Wt) and go to *Step 2*.

Step 7: If $IN = \emptyset$, $\boldsymbol{e}^*$ is the most preferred solution. Otherwise, present the DM the solutions in set IN. Stop.

In Step 2.2, if $\boldsymbol{e}_i$ is preferred to $\boldsymbol{e}^*$ after making necessary updates, we then consider 2 possibilities: "$\boldsymbol{e}_i$ is preferred to $\boldsymbol{e}^*$" or "the DM is indifferent between $\boldsymbol{e}^*$ and $\boldsymbol{e}_i$". Using the following theorem, we omit "$\boldsymbol{e}^*$ is preferred to $\boldsymbol{e}_i$" case which is not possible. In Step 4.2, a similar situation exists. For the sake of completeness, we show this result with the following theorem.

Theorem 2: Let Δ be the minimum value difference between alternatives to warrant preference between two alternatives. Consider three alternatives $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$, and assume the DM is indifferent between $\boldsymbol{A}$ and $\boldsymbol{B}$, and prefers $\boldsymbol{C}$ to $\boldsymbol{A}$. Then $\boldsymbol{B}$ cannot be preferred to $\boldsymbol{C}$.

Proof: If the DM is indifferent between $\boldsymbol{A}$ and $\boldsymbol{B}$, then $u(\boldsymbol{B}) - \Delta < u(\boldsymbol{A}) < u(\boldsymbol{B}) + \Delta$. If $\boldsymbol{C}$ is preferred to $\boldsymbol{A}$, then $u(\boldsymbol{C}) \leq u(\boldsymbol{A}) - \Delta$. $u(\boldsymbol{C}) \leq u(\boldsymbol{A}) - \Delta < u(\boldsymbol{B}) \Rightarrow u(\boldsymbol{C}) < u(\boldsymbol{B})$. Therefore, $\boldsymbol{B}$ cannot be preferred to $\boldsymbol{C}$. □

So far, we have discussed the interactive *LIN-u* algorithm for a single round. In the next section, we discuss the multi-round case.

4.3 Interactive *LIN-u* for Multi-round

In the multi-round case, at each round, we expect the sellers to improve their bids in such a way that the resulting combinations of the next round have improved preference function values of approximately "100γ" percent of the estimated value of the best combination of the current round as in Karakaya and Köksalan (2011). Therefore, after estimating a preference function based on past preferences of the buyer, we provide information to the sellers about a possible way of improving their bids. Using the information together with their cost functions, sellers update their bids for the next round. The auction continues until a termination condition is met.

4.3.1 Bid Update

At the end of each round, sellers update their bids for the next round according to their cost functions. Each seller solves his/her own mathematical model (explained in Section 3.4) to update his/her bids. If there are feasible bids with extra profit, then the seller gives the best possible bids with his/her predetermined mark-up.

After taking the updated bids from the sellers, we find a new bid combination as the incumbent of the current round using the estimated weights found at the end of the previous round and continue.

We next discuss checking the status of the previous best solution (whether it is extreme or nonextreme supported, or unsupported nondominated, or dominated).

4.3.2 Status of the Solution (*SoS*)

We guarantee to find the most preferred bid combination of the current round by applying *LIN-u* under the linear preference function assumption. In Step 1 of *LIN-u* we select arbitrary weights from the feasible weight space and find an incumbent. Then we progress by checking the adjacent efficient alternatives of the incumbent. In the multi-round case we keep the best alternative(s) up to the current round to be compared with the new incumbent of the current round. In this section, we explain the method for the multi-round case. For the sake of simplicity, we assume that we have a single best alternative at the end of any round. That is in Step 7 of *LIN-u*, we assume $IN = \emptyset$.

Let $\boldsymbol{e}^{PREV}$ be the best alternative up to the current round. At the beginning of the current round, after finding an incumbent with the estimated weights and updated bids, we check whether $\boldsymbol{e}^{PREV}$ is extreme supported or not.

Here, we utilize the notation of *LIN-u*. We consider an additional step, Step 0, at the beginning of the algorithm. The remaining steps are the same as *LIN-u*.

Step 0: Find a solution, $\boldsymbol{e}_i$, using $\boldsymbol{w}$. Check whether $\boldsymbol{e}^{PREV}$ is extreme supported or not by applying *SoS* procedure (below). If it is extreme supported, set $\boldsymbol{e}^* = \boldsymbol{e}^{PREV}$ and go to *Step 2.1 (of LIN-u)*. Otherwise, set $\boldsymbol{e}^* = \boldsymbol{e}_i$ and go to *Step 3 (of LIN-u)*.

We consider Figure 4.2 to visualize the possible regions where bid $\boldsymbol{e}_i$ may be located.

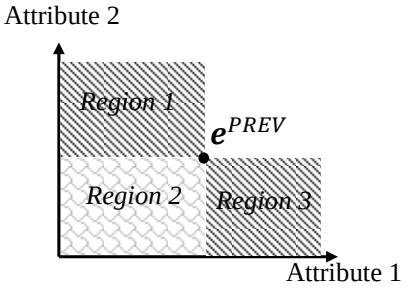


Figure 4.2 The possible regions for $\boldsymbol{e}_i$

We check whether an alternative is extreme supported or not by applying the following *SoS* procedure:

The steps of SoS:

Step 1: Check the dominance relations between $\boldsymbol{e}_i$ and $\boldsymbol{e}^{PREV}$. If $\boldsymbol{e}_i$ is in Region 2, go to *Step 5*. Otherwise, if there exists $\hat{\boldsymbol{e}}$ such that $\hat{\boldsymbol{e}}$ and $\boldsymbol{e}_i$ convex dominate $\boldsymbol{e}^{PREV}$, go to *Step 5*.

Step 2: If $\boldsymbol{e}_i$ is in Region 1 find the east adjacent alternative of $\boldsymbol{e}^{PREV}$, otherwise find the west adjacent alternative of $\boldsymbol{e}^{PREV}$. If there does not exist an adjacent alternative, go to *Step 4*. Otherwise go to *Step 3*.

Step 3: Check whether there exists $\hat{\boldsymbol{e}}$ such that $\hat{\boldsymbol{e}}$ and the adjacent alternative convex dominate $\boldsymbol{e}^{PREV}$. If there does not exist $\hat{\boldsymbol{e}}$, go to *Step 4*. Otherwise go to *Step 5*.

Step 4: Set $\boldsymbol{e}^* = \boldsymbol{e}^{PREV}$ and stop.

Step 5: Set $\boldsymbol{e}^* = \boldsymbol{e}_i$ and stop.

In Step 1, if $\boldsymbol{e}_i$ is in Region 2, $\boldsymbol{e}_i$ dominates $\boldsymbol{e}^{PREV}$. Therefore we stop and continue with the new found alternative, $\boldsymbol{e}_i$. If $\boldsymbol{e}_i$ is in Region 1 or Region 3 we need an iterative procedure. We explain the procedure for the case that $\boldsymbol{e}_i$ is in Region 1 and the procedure is similar for Region 3.

If $\boldsymbol{e}_i$ is in Region 1, we search for an $\hat{\boldsymbol{e}}$ such that $\hat{\boldsymbol{e}}$ and $\boldsymbol{e}_i$ convex dominate $\boldsymbol{e}^{PREV}$. We set $w = \frac{e_{i2} - e_2^{PREV}}{e_1^{PREV} - e_2^{PREV} - e_{i1} + e_{i2}}$, bound the search region as in Figure 4.3 and solve model (SoS_S1).

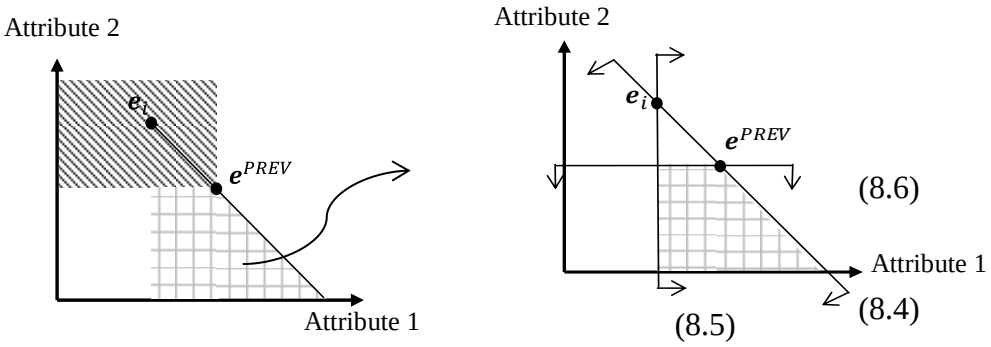


Figure 4.3 The search region in Step 1

Model (SoS_S1)

Min $w\hat{e}_1 + (1-w)\hat{e}_2 + \rho\,\hat{e}_2$ (8.1)

s.to

$\sum_{i=1}^{I}\sum_{t=1}^{T_i}\pi_{itm}y_{it} \geq 1 \qquad \forall m$ (8.2)

$\hat{e}_j = \sum_{i=1}^{I}\sum_{t=1}^{T_i} a_{itj}y_{it} \qquad \forall j$ (8.3)

$w\hat{e}_1 + (1-w)\hat{e}_2 \leq we_1^{PREV} + (1-w)e_2^{PREV}$ (8.4)

$\hat{e}_1 \geq e_{i1} + \rho$ (8.5)

$\hat{e}_2 \leq e_2^{PREV}$ (8.6)

$y_{it} \in \{0,1\}$ (8.7)

where ρ is a small positive constant.

The augmented part ($\rho\hat{e}_2$) in the objective function is used to break ties between a candidate solution on the line passing through $\boldsymbol{e}^{PREV}$ and $\boldsymbol{e}_i$, and $\boldsymbol{e}^{PREV}$. First two constraints are the same as the first two constraints of (ALT). (8.4), (8.5) and (8.6) bound the search region. Since $\boldsymbol{e}_i$ is a supported solution, (8.5) is redundant; however to emphasize the search region we keep it in the model. We note that instead of using an augmented part, the problem can also be solved in a lexicographic manner. First, the problem can be solved to minimize $w\hat{e}_1 + (1-w)\hat{e}_2$. If $\hat{e}_2 = e_2^{PREV}$, then the problem can be solved to minimize $\hat{e}_2$, without sacrificing from the optimal value of the former objective. If there is a feasible solution, $\hat{\boldsymbol{e}}$, different than $\boldsymbol{e}^{PREV}$, we conclude that $\boldsymbol{e}^{PREV}$ is unsupported or nonextreme supported, and we continue with $\boldsymbol{e}_i$. Otherwise we keep searching with Step 2 of *SoS*.

In Step 2, we search for an east adjacent efficient alternative of $\boldsymbol{e}^{PREV}$ in the shaded region in Figure 4.4 by using *ADJ_E*. If there does not exist an adjacent alternative on the east of $\boldsymbol{e}^{PREV}$, we conclude that $\boldsymbol{e}^{PREV}$ is extreme supported and continue with $\boldsymbol{e}^{PREV}$. Otherwise we keep searching with Step 3 of *SoS*.

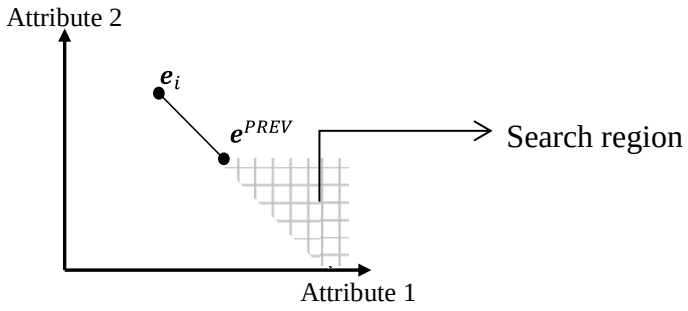


Figure 4.4 The search region in Step 2

In Step 3, we search for an $\hat{\boldsymbol{e}}$ such that $\hat{\boldsymbol{e}}$ and $\boldsymbol{e}_E$ convex dominate $\boldsymbol{e}^{PREV}$. We set $w = \frac{e_{E2} - e_2^{PREV}}{e_1^{PREV} - e_2^{PREV} - e_{E1} + e_{E2}}$, bound the search region as in Figure 4.5 and solve model (SoS_S2).

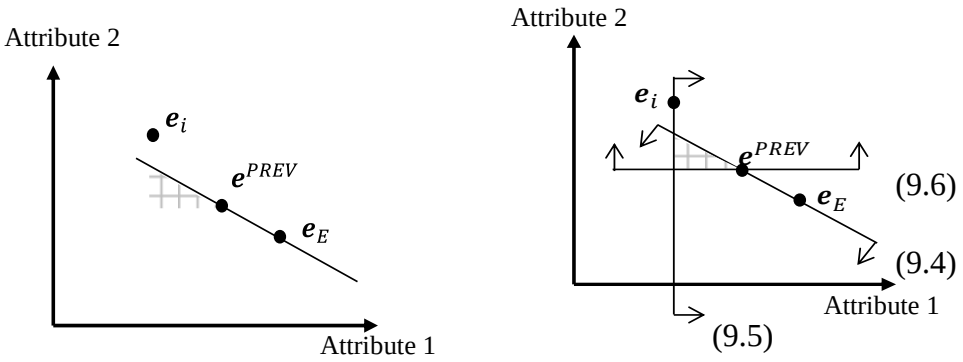


Figure 4.5 The search region in Step 3

Model (SoS_S2)

Min $w\hat{e}_1 + (1-w)\hat{e}_2$ (9.1)

s.to

$\sum_{i=1}^{I}\sum_{t=1}^{T_i}\pi_{itm}y_{it} \geq 1 \qquad \forall m$ (9.2)

$\hat{e}_j = \sum_{i=1}^{I}\sum_{t=1}^{T_i}a_{itj}y_{it} \qquad \forall j$ (9.3)

$w\hat{e}_1 + (1-w)\hat{e}_2 \leq we_1^{PREV} + (1-w)e_2^{PREV}$ (9.4)

$\hat{e}_1 \geq e_{i1} + \rho$ (9.5)

$\hat{e}_2 \geq e_2^{PREV} + \rho$ (9.6)

$y_{it} \in \{0,1\}$ (9.7)

where ρ is a small positive constant.

In SoS_S2, we aim at finding a feasible solution in the search region. Therefore any objective function can be used in the model. First two constraints are the same as the first two constraints of (ALT). (9.4), (9.5) and (9.6) bound the search region. As stated before, since $\boldsymbol{e}_i$ is a supported solution, (9.5) is redundant. If there is a feasible solution, $\hat{\boldsymbol{e}}$, we conclude that $\boldsymbol{e}^{PREV}$ is unsupported or nonextreme supported; otherwise, we say that $\boldsymbol{e}^{PREV}$ is extreme supported. The *SoS* procedure ends with this step.

In Step 2 of *SoS*, we choose the east adjacent alternative of $\boldsymbol{e}^{PREV}$ to search as we assume that $\boldsymbol{e}_i$ is in Region 1. Finding west adjacent alternative of $\boldsymbol{e}^{PREV}$ will give similar results. If $\boldsymbol{e}^{PREV}$ has no east or west adjacent alternative in the search region, we simply conclude that $\boldsymbol{e}^{PREV}$ is extreme supported. If $\boldsymbol{e}^{PREV}$has an adjacent alternative, we can determine whether $\boldsymbol{e}^{PREV}$ is extreme supported or not with an additional step. We use the adjacent alternative to enlarge the search region as much as possible for finding an $\hat{\boldsymbol{e}}$ such that $\hat{\boldsymbol{e}}$ and $\boldsymbol{e}_E$ convex dominate $\boldsymbol{e}^{PREV}$. For the sake of completeness, we show this result with the following theorem considering the east adjacent alternative.

Theorem 3: Consider Regions 4 and 5 in Figure 4.6. If there exist $\hat{\boldsymbol{e}}$ in Region 4 and $\boldsymbol{A} \neq \boldsymbol{e}_E$ in Region 5 such that $\hat{\boldsymbol{e}}$ and $\boldsymbol{A}$ convex dominate $\boldsymbol{e}^{PREV}$, then $\hat{\boldsymbol{e}}$ and $\boldsymbol{e}_E$ also convex dominate $\boldsymbol{e}^{PREV}$**.** The reverse is not necessarily true.

Proof: If $\hat{\boldsymbol{e}}$ and $\boldsymbol{A}$ convex dominate $\boldsymbol{e}^{PREV}$, then $\exists\theta \geq 0 \ni \boldsymbol{e}^{PREV} + \theta(\boldsymbol{e}^{PREV} - \boldsymbol{A}) \geq \hat{\boldsymbol{e}}$. Since $\boldsymbol{e}_E$ is the east adjacent alternative to $\boldsymbol{e}^{PREV}$, $\frac{(e_2^{PREV} - A_2)}{(A_1 - e_1^{PREV})} < \frac{(e_2^{PREV} - e_{E,2})}{(e_{E,1} - e_1^{PREV})}$. Therefore the following two inequalities hold for $\theta' = \theta \frac{(A_1 - e_1^{PREV})}{(e_{E,1} - e_1^{PREV})}$.

$e_1^{PREV} + \theta'\left(e_1^{PREV} - e_{E,1}\right) \geq \hat{e}_1 \qquad (1)$

$e_2^{PREV} + \theta'\left(e_2^{PREV} - e_{E,2}\right) > \hat{e}_2 \qquad (2)$

(1) is equivalent to $e_1^{PREV} + \theta(e_1^{PREV} - A_1) \geq \hat{e}_1$ and since $\frac{(e_2^{PREV} - A_2)}{(A_1 - e_1^{PREV})} < \frac{(e_2^{PREV} - e_{E,2})}{(e_{E,1} - e_1^{PREV})}$, (2) holds. Using (1) and (2) we conclude that $\hat{\boldsymbol{e}}$ and $\boldsymbol{e}_E$ also convex dominate $\boldsymbol{e}^{PREV}$.

We show that the reverse is not true with a counter example. Consider the alternatives in Figure 5. Although $\hat{\boldsymbol{e}}$ and $\boldsymbol{e}_E$ convex dominate $\boldsymbol{e}^{PREV}$, $\hat{\boldsymbol{e}}$ and $\boldsymbol{A}$ cannot convex dominate $\boldsymbol{e}^{PREV}$. □

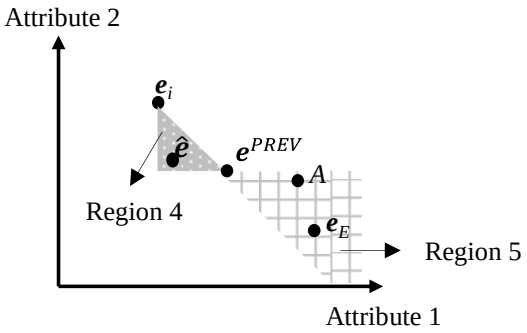


Figure 4.6 Counter example for Theorem 3

In the multi-round case, we add the following constraint in (ALT) to avoid inferior solutions:

$$w\hat{e}_1 + (1-w)\hat{e}_2 \leq we_1^* + (1-w)e_2^*$$

The constraint indicates that the estimated preference function value of the new solution should be at most as big as that of the incumbent.

To test the performance of *LIN-u*, we use 10 different weight values for the price attribute to generate different problems considering an underlying linear preference function for the buyer. The percent deviations for the linear case are reported in Table 4.1.

Table 4.1 Percentage deviations between the results of the *LIN-u* and the decentralized optimal solution

λ=0.05	λ=0.15	λ=0.25	λ=0.35	λ=0.45	λ=0.55	λ=0.65	λ=0.75	λ=0.85	λ=0.95
0.0084	0.0064	0.0000	0.0000	0.0040	0.0073	0.0000	0.0000	0.0050	0.0061

In all problems for the linear case, the winning bidders found with the algorithm and Decentralized are the same, i.e. allocative efficiency is satisfied. The percentage deviations in Table 4.1 are very small, i.e., for all problems the buyer's preference function found with the algorithm is very close to that found

by Decentralized. Moreover, the average number of pairwise comparisons the buyer is required to make is 14.8 over the 10 problems with different weights.

4.4 A Heuristic for Underlying Nonlinear Functions

In this section, we assume that the buyer has an underlying decreasing quasiconvex preference function. As in Section 3.10, we locally approximate the buyer's preference function with a linear function. We use the (IR) model to estimate the weight values.

At the end of Step 7 of the *LIN-u* algorithm, we continue searching to find some unsupported solutions to present the DM. To do this, after reducing the search space we solve weighted Tchebycheff programs to find some unsupported alternatives. Lemma 1 gives the theory and Figure 4.7 demonstrates how we reduce the search space. For the sake of simplicity, we first assume that there is a single current best alternative at the end of Step 7 of *LIN-u*, i.e. $IN = \emptyset$.
Lemma 1 in Korhonen et al. (1984) reduces the objective space based on the preferences of the DM under the assumption that the DM has a quasiconcave preference function to be maximized. The result directly applies to the case of minimizing a quasiconvex function and we present the lemma in the latter context.

Lemma 1: Let $u: R^J \rightarrow R^1$ be a decreasing quasiconvex preference function and $\boldsymbol{x}_i \in R^J, i = 1,2, \dots, n$. Let $u(\boldsymbol{x}_i) < u(\boldsymbol{x}_k),\ i \neq k$. Let $\boldsymbol{Z} = \{\boldsymbol{z}: \boldsymbol{z} = \boldsymbol{x}_k + \sum_{i=1, i\neq k}^{n} \mu_i(\boldsymbol{x}_k - \boldsymbol{x}_i), \mu_i \geq 0\}$. Then, $u(\boldsymbol{z}) \geq u(\boldsymbol{x}_k)$.

$\boldsymbol{Z}$ is referred as the cone of inferior solutions. We demonstrate Lemma 1 in Figure 4.7. Consider three supported nondominated solutions $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$, where $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{B}$, $\boldsymbol{C}$ are adjacent efficient alternatives. Assume the DM prefers $\boldsymbol{B}$ to both $\boldsymbol{A}$ and $\boldsymbol{C}$.

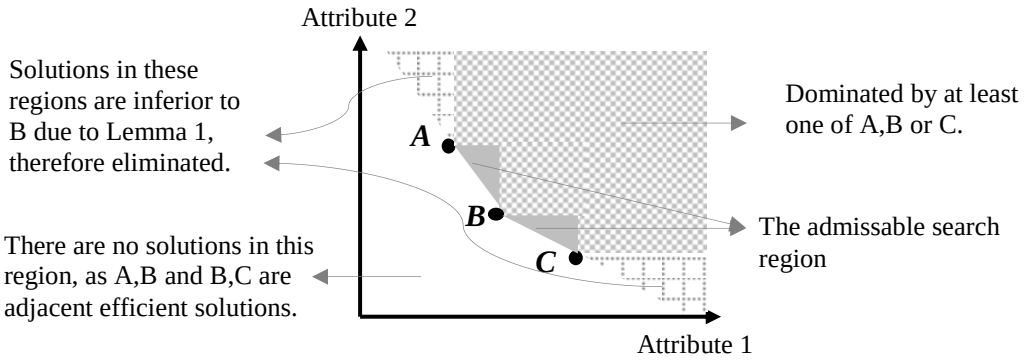


Figure 4.7 Reduced search space

We next search for an unsupported nondominated solution using the Tchebycheff program in one of the triangles in Figure 4.7. If there is a solution that is preferred to ***B***, the new cone of inferior solutions made up of the new solution and ***B*** contains the remaining triangle and we do not need to search it based on Lemma 1. However, if there is no solution in the triangle or if ***B*** is preferred to the new solution, we search for a new unsupported nondominated solution in the other triangle. This procedure requires at most two additional comparisons if there is a single best alternative at the end of Step 7 of *LIN-u*. Otherwise, besides the triangles of the extremes of set *IN*, we also search the triangle in the middle of the region.

To find a solution in one of the triangles, we first find the local ideal and nadir points of the two solutions, say ***B*** and ***C***, as in Figure 4.8 and solve the (TCH) problem to find an unsupported nondominated solution in the triangle.

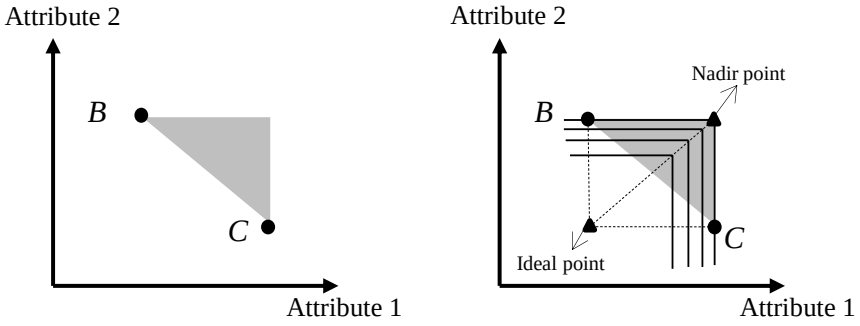


Figure 4.8 Ideal and nadir points

Parameters:

ρ: a small positive constant

a_{itj}: level of attribute *j* offered by seller *i* in bid *t*

π_{itm}: 1 if bid *t* of seller *i* includes item *m*; 0 otherwise

T_i: the number of bids offered by seller *i*

I_j^*:level of attribute *j* of the ideal point

N_j :level of attribute *j* of the nadir point

β: calculated weight of attribute 1 where $\beta = (N_2 - I_2^*)/(N_1 - I_1^* + N_2 - I_2^*)$

Decision Variables:

z : the Tchebycheff distance value of the solution from the ideal point

$\hat{e}_j$: level of attribute *j* of the optimal alternative

y_{it}: 1 if bid *t* of seller *i* is selected to be in the efficient combination; 0 otherwise

Model (TCH)

Min $z + \rho(\hat{e}_1 + \hat{e}_1)$ (10.1)

s.to

$\sum_{i=1}^{I} \sum_{t=1}^{T_i} \pi_{itm} y_{it} \geq 1 \quad \forall m$ (10.2)

$\hat{e}_j = \sum_{i=1}^{I} \sum_{t=1}^{T_i} a_{itj} y_{it} \quad j = 1,2$ (10.3)

$z \geq \beta(\hat{e}_1 - I_1^*)$ (10.4)

$z \geq (1 - \beta)(\hat{e}_2 - I_2^*)$ (10.5)

$\hat{e}_1 \leq N_1 - \rho$ (10.6)

$\hat{e}_2 \leq N_2 - \rho$ (10.7)

$y_{it} \in \{0,1\}$ (10.8)

In (TCH), we try to minimize the weighted Tchebycheff distance from the ideal point. To avoid weakly nondominated but dominated solutions, we use the augmented part in objective function. Constraints (10.6) and (10.7) bound the search region.

Although there may be several alternatives in the triangle, we find only one of them to limit the number of questions asked to the DM.

In the linear case, while finding an adjacent efficient alternative to an incumbent, we only consider those candidate incumbents that could be preferred based on the available weight space. Although this is a valid procedure when the underlying preference function is linear, it does not apply for underlying nonlinear preference functions. Therefore, we do not eliminate such alternatives in this case. Furthermore, in the linear multi-round case, we apply the *SoS* procedure and if $\boldsymbol{e}^{PREV}$ is not an extreme supported solution, we eliminate it. However, in the nonlinear case we keep that solution until the end of the corresponding round. After solving the Tchebycheff program, we check whether $\boldsymbol{e}^{PREV}$ is in a cone-inferior region or not. If $\boldsymbol{e}^{PREV}$ is not in a cone-inferior region, we also ask the DM to compare the incumbent and $\boldsymbol{e}^{PREV}$.

We next provide the results of *LIN-u* when the buyer has an underlying nonlinear preference function. In the nonlinear case, we test the performance of the algorithm by simulating the preferences of the buyer using weighted L_α preference functions; specifically we use the weighted Euclidean $(\alpha = 2)$ and the weighted Tchebycheff $(\alpha = \infty)$ functions. The percent deviations for the nonlinear case are reported in Tables 4.2 and 4.3.

Table 4.2 Percentage deviations between the results of the algorithm and decentralized optimal solution under weighted Euclidean preference function

λ=0.05	λ=0.15	λ=0.25	λ=0.35	λ=0.45	λ=0.55	λ=0.65	λ=0.75	λ=0.85	λ=0.95
0.0408	0.0090	-0.0796	-0.0303	-0.2306	-0.2971	0.1356	0.2492	0.0208	0.0000

Table 4.3 Percentage deviations between the results of the algorithm and decentralized optimal solution under weighted Tchebycheff preference function

λ=0.05	λ=0.15	λ=0.25	λ=0.35	λ=0.45	λ=0.55	λ=0.65	λ=0.75	λ=0.85	λ=0.95
0.0316	0.6988	1.0351	0.2601	4.8190	-0.6618	-4.4448	6.1772	0.0000	0.0000

When we look at the results in Tables 4.2 and 4.3, we see that in some problems our algorithm performed even better than Decentralized. As discussed in Chapter 3, this is due to the nonlinear nature of the preference function and round off errors. Although Decentralized finds slightly better preference function values for each bid separately, when a combination is constructed, the preference function of a combination for the Decentralized case may be worse than that of ours for the considered nonlinear preference functions. The average percent deviations are -0.0182% and 0.7915% for $\alpha = 2$ and $\alpha = \infty$, respectively. Furthermore, the buyer compares 45.7 and 45.1 pairs on the average of 10 problems with different weights for $\alpha = 2$ and $\alpha = \infty$, respectively. Applying the interactive method decreases the number of possible efficient combinations to be evaluated by the buyer substantially.

The experiments show that in all test problems the percent deviations are very small, i.e. the buyer's preference function value corresponding to the solution found with the algorithm is close to that of Decentralized. Moreover, the number of questions asked to the buyer with the interactive method is small. These imply that the estimation and guidance mechanisms of our approach worked well in all the test problems.

CHAPTER 5

AN INTERACTIVE APPROACH FOR BI-ATTRIBUTE MULTI-ITEM AUCTIONS UNDER QUASICONVEX PREFERENCE FUNCTIONS

In this chapter, we develop an interactive method to find the most preferred bid combination of a buyer having a quasiconvex preference function. We first explain the *QCX-u* algorithm and its versions. We then discuss the L_α-*u* algorithm which estimates both alpha and weight values of the underlying preference function. In each algorithm, we provide the results of the performance tests.

5.1 The Interactive Algorithm (*QCX-u*)

We develop an interactive algorithm, *QCX-u*, to find the most preferred combination of a buyer having a quasiconvex preference function for the 2-attribute case. In this case we assume that the buyer can distinguish between bids, i.e. $\Delta = 0$. We modify and extend the *LIN-u* algorithm.

Similar to the heuristic approach in Section 4.4, at the end of each round we continue searching in the reduced search space using Lemma 1 to find the most preferred solution of the DM in the current round. As discussed in Section 4.4, the reduced space consists of two triangles.

We consider different versions of the algorithm and test their performances against the Decentralized case. The versions are *each round*, *last round*, *band*, and *limited number of questions*. In the *each round* case, we search the whole triangle(s) in each round to find the most preferred alternative. In the *last round* case, we apply *LIN-u* algorithm for the multi-round case and search the whole triangle(s) only in the last round. In the *band* case, in each round, instead of the whole triangle(s) we search some portion of the triangle. In the *limited number of questions* case, while searching the triangle(s) in each round we limit the number of unsupported solutions visited.

In *QCX-u* algorithm, based on the preferences of the DM we reduce the search space and conduct our search in the reduced region. To do this we use Lemma 1 (in Section 4.4) and construct cones with two alternatives based on the pairwise comparisons of the DM. We use the notation $\boldsymbol{A} \rhd \boldsymbol{B}$ to depict two-point cones where $\boldsymbol{A}$ is preferred to $\boldsymbol{B}$.

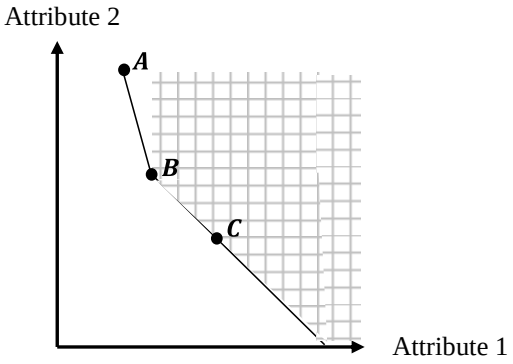


Figure 5.1 An example for two-point cones

To illustrate, consider the three alternatives in Figure 5.1 and assume that the DM prefers $\boldsymbol{A}$ to $\boldsymbol{B}$ and $\boldsymbol{B}$ to $\boldsymbol{C}$. We generate two cones, $\boldsymbol{A} \rhd \boldsymbol{B}$ and $\boldsymbol{B} \rhd \boldsymbol{C}$. We detect and eliminate redundant cones. For instance, we eliminate $\boldsymbol{B} \rhd \boldsymbol{C}$ since the inferior region implied by this cone is a subset of that of cone $\boldsymbol{A} \rhd \boldsymbol{B}$.

For the example above $(\boldsymbol{A} \rhd \boldsymbol{B})$, to reduce the search space we write the following cone constraints:

$$\hat{e}_1 \leq B_1 - \rho + Mz \quad \text{(Cons1)}$$

$$w_{AB}\hat{e}_1 + (1 - w_{AB})\hat{e}_2 \leq w_{AB}B_1 + (1 - w_{AB})B_2 - \rho + M(1 - z) \quad \text{(Cons2)}$$

where ρ is a small positive constant, M is a big positive number, z is a binary variable and $w_{AB} = \frac{A_2 - B_2}{B_1 - B_2 - A_1 + A_2}$ (i.e. w_{AB} and $(1 - w_{AB})$ are the weights of the linear function passing through alternatives $\boldsymbol{A}$ and $\boldsymbol{B}$.)

In the constraints above we use a binary variable z to enforce that if the first attribute value of the candidate solution is at least as big as B_1, then (Cons2) becomes active indicating that the candidate solution should be below the line passing through alternatives $\boldsymbol{A}$ and $\boldsymbol{B}$. With these two constraints we aim to reduce the search region and avoid the inferior solutions. Although we exemplify the cone constraints for the preferred alternative having smaller attribute 1 value, the constraints are similar for the case where the preferred alternative has a smaller attribute 2 value. In our models, for each valid cone we write such constraint pairs by defining a binary variable. To reduce the number of binary variables, we eliminate redundant cones.

Elimination of redundant cones

Consider alternatives $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$ and $\boldsymbol{D}$. Assume that the DM prefers $\boldsymbol{A}$ to $\boldsymbol{B}$ and $\boldsymbol{C}$ to $\boldsymbol{D}$. If we want to check whether $\boldsymbol{C} \rhd \boldsymbol{D}$ is redundant relative to $\boldsymbol{A} \rhd \boldsymbol{B}$, we check whether each point in $\boldsymbol{C} \rhd \boldsymbol{D}$ is dominated by a point in $\boldsymbol{A} \rhd \boldsymbol{B}$. That is, if there exists $\beta \geq 0$ satisfying $\boldsymbol{B} + \beta(\boldsymbol{B} - \boldsymbol{A}) \leq \boldsymbol{D} + \beta'(\boldsymbol{D} - \boldsymbol{C})$ for each $\beta' \geq 0$, then $\boldsymbol{C} \rhd \boldsymbol{D}$ is redundant. Instead of solving a mathematical model, we use a simple procedure to determine whether a cone is redundant or not.

After taking the preferences of the DM and generating a new cone, we apply the following procedure to detect the redundant cones with respect to the new cone. For each existing cone, we make pairwise *cone validity check* with the new cone. For the sake of simplicity we use the cones $\boldsymbol{A} \rhd \boldsymbol{B}$ and $\boldsymbol{C} \rhd \boldsymbol{D}$ to explain the

procedure. Since we consider nonnegative attribute values in our test problems, suppose all four alternatives have positive attribute values.

Cone Validity Check

Step 1: If $A_1 < B_1$ and $C_1 < D_1$ go to *Step 2*; else if $A_1 > B_1$ and $C_1 > D_1$, go to *Step 3*. Otherwise go to *Step 6*.

Step 2: If $B_1 \leq D_1$ and $\frac{B_2-D_2}{A_2-B_2} \leq \frac{D_1-B_1}{B_1-A_1}$, set $\beta = \frac{-B_2}{B_2-A_2}$, $\beta' = \frac{-D_2}{D_2-C_2}$ and go to *Step 4*. Otherwise go to *Step 6*.

Step 3: If $B_2 \leq D_2$ and $\frac{D_1-B_1}{B_1-A_1} \leq \frac{D_2-B_2}{B_2-A_2}$, set $\beta = \frac{-B_1}{B_1-A_1}$, $\beta' = \frac{-D_1}{D_1-C_1}$ and go to *Step 4*. Otherwise go to *Step 6*.

Step 4: If $\boldsymbol{B} + \beta(\boldsymbol{B} - \boldsymbol{A}) \leq \boldsymbol{D} + \beta'(\boldsymbol{D} - \boldsymbol{C})$, go to Step 5; otherwise go to *Step 6*.

Step 5: $\boldsymbol{C} \rhd \boldsymbol{D}$ is redundant, go to *Step 6*.

Step 6: Stop.

In Step 1, we check the directions of the cones (i.e. compare the first attribute values of preferred and nonpreferred alternatives in each cone). If cones have different directions, by using Theorem 4 (below), $\boldsymbol{C} \rhd \boldsymbol{D}$ cannot be redundant with respect to the new cone, $\boldsymbol{A} \rhd \boldsymbol{B}$, and the procedure stops. If both cones have the same directions, we also check attribute 1 or 2 values (based on the direction) of the nonpreferred alternatives in each cone. Suppose that $A_1 < B_1$ and $C_1 < D_1$, then we check whether $B_1 \leq D_1$ or not. If $B_1 > D_1$, due to Theorem 5 (below) we conclude that $\boldsymbol{C} \rhd \boldsymbol{D}$ cannot be redundant. On the other hand, if $B_1 \leq D_1$, we check some conditions (stated in Theorem 6 below) and if these conditions are satisfied we conclude that $\boldsymbol{C} \rhd \boldsymbol{D}$ is redundant. Otherwise, we conclude that $\boldsymbol{C} \rhd \boldsymbol{D}$ cannot be redundant relative to $\boldsymbol{A} \rhd \boldsymbol{B}$ and the procedure stops.

Theorem 4: Consider two cones $\boldsymbol{A} \rhd \boldsymbol{B}$ and $\boldsymbol{C} \rhd \boldsymbol{D}$ where the cones have different directions. Then $\boldsymbol{C} \rhd \boldsymbol{D}$ cannot be redundant.

Proof: If the cones have different directions there can be 2 cases:

Case 1: $A_1 < B_1$ and $C_1 > D_1$

Case 2: $A_1 > B_1$ and $C_1 < D_1$

Case 1

Since $B_1 - A_1 > 0$ and $B_1 > 0$, for $\beta' = \frac{-D1}{D1-C1}$, there does not exist any $\beta \geq 0$ satisfying $\boldsymbol{B} + \beta(\boldsymbol{B} - \boldsymbol{A}) \leq \boldsymbol{D} + \beta'(\boldsymbol{D} - \boldsymbol{C})$.

Case 2

Since $B_2 - A_2 > 0$ and $B_2 > 0$, for $\beta' = \frac{-D2}{D2-C2}$, there does not exist any $\beta \geq 0$ satisfying $\boldsymbol{B} + \beta(\boldsymbol{B} - \boldsymbol{A}) \leq \boldsymbol{D} + \beta'(\boldsymbol{D} - \boldsymbol{C})$. □

Theorem 5: Consider the cones in Theorem 4. Suppose that $A_1 < B_1$ and $C_1 < D_1$. If $B_1 > D_1$, then $\boldsymbol{C} \rhd \boldsymbol{D}$ cannot be redundant.

Proof:

As stated above, $\boldsymbol{C} \rhd \boldsymbol{D}$ is redundant if and only if each point in $\boldsymbol{C} \rhd \boldsymbol{D}$ is dominated by a point in $\boldsymbol{A} \rhd \boldsymbol{B}$. That is, if $\exists\, \beta \geq 0\, \epsilon\, \boldsymbol{B} + \beta(\boldsymbol{B} - \boldsymbol{A}) \leq \boldsymbol{D} + \beta'(\boldsymbol{D} - \boldsymbol{C})$ for each $\beta' \geq 0$, then $\boldsymbol{C} \rhd \boldsymbol{D}$ is redundant. Since $B_1 - A_1 > 0$ and $B_1 - D_1 > 0$, for $\beta' = 0$, there does not exist any $\beta \geq 0$ satisfying $\boldsymbol{B} + \beta(\boldsymbol{B} - \boldsymbol{A}) \leq \boldsymbol{D} + \beta'(\boldsymbol{D} - \boldsymbol{C})$. □

Theorem 6: Consider the cones in Theorem 5. If $B_1 \leq D_1$, $\frac{B_2-D_2}{A_2-B_2} \leq \frac{D_1-B_1}{B_1-A_1}$ and $\boldsymbol{B} + \frac{-B_2}{B_2-A_2}(\boldsymbol{B} - \boldsymbol{A}) \leq \boldsymbol{D} + \frac{-D_2}{D_2-C_2}(\boldsymbol{D} - \boldsymbol{C})$, then $\boldsymbol{C} \rhd \boldsymbol{D}$ is redundant.

Proof:

$\boldsymbol{C} \rhd \boldsymbol{D}$ is redundant if and only if both starting and ending points of $\boldsymbol{C} \rhd \boldsymbol{D}$ are dominated by the points in $\boldsymbol{A} \rhd \boldsymbol{B}$ as we consider linear functions. Therefore, we check whether both starting and ending points of $\boldsymbol{C} \rhd \boldsymbol{D}$ are dominated or not.

$\boldsymbol{B} + \beta(\boldsymbol{B} - \boldsymbol{A}) \leq \boldsymbol{D} + \beta'(\boldsymbol{D} - \boldsymbol{C})$ can be rewritten as follows:

$B_1 + \beta(B_1 - A_1) \leq D_1 + \beta'(D_1 - C_1)$

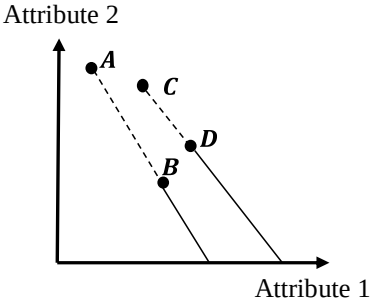
$B_2 + \beta(B_2 - A_2) \leq D_2 + \beta'(D_2 - C_2)$

The starting point of $\boldsymbol{C} \rhd \boldsymbol{D}$ is (D_1, D_2), that is set $\beta' = 0$. For $\beta' = 0$, we check whether there exists $\beta \geq 0$ such that $\boldsymbol{B} + \beta(\boldsymbol{B} - \boldsymbol{A}) \leq \boldsymbol{D}$. If there exists feasible β values, i.e. if $\frac{B_2 - D_2}{A_2 - B_2} \leq \frac{D_1 - B_1}{B_1 - A_1}$, then the requirement for the starting points is satisfied. On the other hand, if no such β exists, no need to check the ending point.

Without loss of generality, consider the minimum attainable attribute 2 value as zero. Then, set $\beta' = \frac{-D_2}{D_2 - C_2}$ and the ending point of $\boldsymbol{C} \rhd \boldsymbol{D}$ is $\left(D_1 + \frac{-D_2}{D_2 - C_2}(D_1 - C_1), 0\right)$. We compare it with the ending point of $\boldsymbol{A} \rhd \boldsymbol{B}$, $\left(B_1 + \frac{-B_2}{B_2 - A_2}(B_1 - A_1), 0\right)$. If $B_1 + \frac{-B_2}{B_2 - A_2}(B_1 - A_1) \leq D_1 + \frac{-D_2}{D_2 - C_2}(D_1 - C_1)$, then the ending point of $\boldsymbol{C} \rhd \boldsymbol{D}$ is dominated by the ending point of $\boldsymbol{A} \rhd \boldsymbol{B}$.

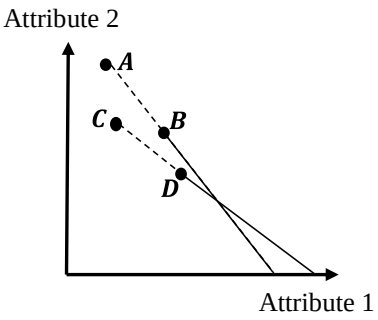
If both starting and ending points of $\boldsymbol{C} \rhd \boldsymbol{D}$ are dominated by the points in $\boldsymbol{A} \rhd \boldsymbol{B}$, then , $\boldsymbol{C} \rhd \boldsymbol{D}$ is redundant. Otherwise, it cannot be redundant. □

We demonstrate some possible cases in Figure 5.2.



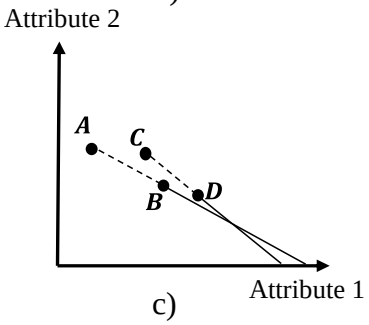
a)

Since both starting and ending points are dominated, $\boldsymbol{C} \rhd \boldsymbol{D}$ is redundant.



b)

Since starting point is not dominated, $\boldsymbol{C} \rhd \boldsymbol{D}$ is not redundant.



c)

Since ending point is not dominated, $\boldsymbol{C} \rhd \boldsymbol{D}$ is not redundant.

Figure 5.2 Some examples for Theorem 6

The steps of QCX-u

Recall that $\boldsymbol{e}_E$ and $\boldsymbol{e}_W$ are the east and west adjacent efficient alternatives to the incumbent, respectively. Select an arbitrary direction (*DIR*) to search an adjacent efficient alternative to the incumbent where *DIR=east* corresponds to searching for $\boldsymbol{e}_E$ and *DIR=west* corresponds to searching for $\boldsymbol{e}_W$. Set iteration counter $i = 1$ and without loss of generality set DIR=*east*. Recall that $\boldsymbol{e}^{PREV}$ is the best alternative up to the current round, e_{k1} and e_{k2} are the values of attributes 1 and 2 of the corresponding solution, $\boldsymbol{e}_k$, respectively, $\boldsymbol{w} = (w, 1-w)$ where w is the estimated weight of attribute 1.

For the sake of simplicity, we first provide the steps to find the best supported solution in a single round. We then discuss the variations of the algorithm.

Step 1: Select an arbitrary set of weights, find an incumbent, $\boldsymbol{e}^*$, and go to *Step 3*.

Step 2: Set $i \leftarrow i + 1$. Find a solution, $\boldsymbol{e}_i$, using $\boldsymbol{w}$. If it is the first round or there does not exist a new $\boldsymbol{e}_i$, go to *Step 3*. Otherwise, if a new round is started, check the status of $\boldsymbol{e}^{PREV}$ and if $\boldsymbol{e}^{PREV}$ is not extreme supported, set $\boldsymbol{e}^* = \boldsymbol{e}_i$ and go to

Step 3. Otherwise, if $\boldsymbol{e}_{i1} < \boldsymbol{e}_1^*$, set *DIR=west*; otherwise set *DIR=east*. Ask the DM $\boldsymbol{e}^*$ versus $\boldsymbol{e}_i$. If

- $\boldsymbol{e}_i$ is preferred to $\boldsymbol{e}^*$, add a constraint $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_i) \geq \rho + \mu$, add $\boldsymbol{e}_i \rhd \boldsymbol{e}^*$, check the validity of the existing cones and write the relevant cone constraints. Set $\boldsymbol{e}^* = \boldsymbol{e}_i$ and go to *Step3*.
- $\boldsymbol{e}^*$ is preferred to $\boldsymbol{e}_i$, add a constraint $\boldsymbol{w}(\boldsymbol{e}_i - \boldsymbol{e}^*) \geq \rho + \mu$, add $\boldsymbol{e}^* \rhd \boldsymbol{e}_i$, check the validity of the existing cones and write the relevant cone constraints. Switch the value of DIR (i.e. set *DIR=east* if it is equal to *west* and set *DIR=west* if it is equal to *east*) and go to *Step 3*.

Step 3: If *DIR=east*, find $\boldsymbol{e}_E$ and set $\boldsymbol{e}_{adj} = \boldsymbol{e}_E$; otherwise find $\boldsymbol{e}_W$ and set $\boldsymbol{e}_{adj} = \boldsymbol{e}_W$. If there exists $\boldsymbol{e}_{adj}$, go to *Step 4*. Otherwise, if both $\boldsymbol{e}_E$ and $\boldsymbol{e}_W$ have been evaluated before, go to *Step 6*; otherwise, switch the value of *DIR* and go to *Step 3*.

Step 4: Ask the DM $\boldsymbol{e}^*$ versus $\boldsymbol{e}_{adj}$. If

- $\boldsymbol{e}_{adj}$ is preferred to $\boldsymbol{e}^*$, add a constraint $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_{adj}) \geq \rho + \mu$, add $\boldsymbol{e}_{adj} \rhd \boldsymbol{e}^*$, check the validity of the existing cones and write the relevant cone constraints. Set $\boldsymbol{e}^* = \boldsymbol{e}_{adj}$ and go to *Step 5*.
- $\boldsymbol{e}^*$ is preferred to $\boldsymbol{e}_{adj}$, add a constraint $\boldsymbol{w}(\boldsymbol{e}_{adj} - \boldsymbol{e}^*) \geq \rho + \mu$, add $\boldsymbol{e}^* \rhd \boldsymbol{e}_{adj}$, check the validity of the existing cones and write the relevant cone constraints. If both $\boldsymbol{e}_E$ and $\boldsymbol{e}_W$ have been evaluated, go to *Step 6*. Otherwise switch the value of *DIR* and go to *Step 3*.

Step 5: Find a feasible set of weights satisfying all constraints corresponding to past preferences of the DM by solving the (IR) and go to *Step 2*.

Step 6: Depending on the version of the algorithm find some candidate solutions, ask the DM and write necessary constraints based on the preferences of the DM. If it is not the first round, while finding candidate solutions if $\boldsymbol{e}^{PREV}$ is not in a cone-inferior region, also consider $\boldsymbol{e}^{PREV}$. The details of this procedure will be discussed later. Set $\boldsymbol{e}^{PREV} = \boldsymbol{e}^*$ and go to *Step 7*.

Step 7: If it is the final round, go to *Step 8*; otherwise inform the sellers about estimations, get the updated bids and go to *Step 2*.

Step 8: Stop.

In Steps 1 and 2, we add cone constraints to the (ALT) model in Section 4.2 to find a new alternative with the estimated weights. For the sake of completeness, we provide the updated model (ALT') below:

Parameters:

M: a big number

ρ: a small positive constant

a_{itj}: level of attribute *j* offered by seller *i* in bid *t*

π_{itm}: 1 if bid *t* of seller *i* includes item *m*; 0 otherwise

T_i: the number of bids offered by seller *i*

w: estimated weight of attribute 1

$w_{\boldsymbol{A}\triangleright\boldsymbol{B}}$: calculated weight of attribute 1 where $w_{\boldsymbol{A}\triangleright\boldsymbol{B}} = \frac{(A_2 - B_2)}{(B_1 - B_2 - A_1 + A_2)}$

e_{kj}: level of attribute *j* in efficient combination *k*

Decision Variables:

y_{it}: 1 if bid *t* of seller *i* is selected to be in the efficient combination; 0 otherwise

$\hat{e}_j$: level of attribute *j* of the optimal alternative

$z_{\boldsymbol{e}_p\triangleright\boldsymbol{e}_n}$: 1 if constraint (11.6) is active; 0 otherwise

$z_{\boldsymbol{e}_l\triangleright\boldsymbol{e}_r}$:1 if constraint (11.8) is active; 0 otherwise

Model (ALT')

Min $w\hat{e}_1 + (1-w)\hat{e}_2$ (11.1)

s.to

$\sum_{i=1}^{I}\sum_{t=1}^{T_i} \pi_{itm} y_{it} \geq 1 \qquad \forall m$ (11.2)

$\hat{e}_j = \sum_{i=1}^{I}\sum_{t=1}^{T_i} a_{itj} y_{it} \qquad j = 1,2$ (11.3)

$w\hat{e}_1 + (1-w)\hat{e}_2 \leq we_1^* + (1-w)e_2^*$ (11.4)

$\hat{e}_1 \leq e_{n1} - \rho + Mz_{\boldsymbol{e}_p \triangleright \boldsymbol{e}_n}$ for each $\boldsymbol{e}_p \triangleright \boldsymbol{e}_n, e_{p1} > e_{n1}$ (11.5)

$w_{\boldsymbol{e}_p \triangleright \boldsymbol{e}_n}\hat{e}_1 + (1 - w_{\boldsymbol{e}_p \triangleright \boldsymbol{e}_n})\hat{e}_2 \leq w_{\boldsymbol{e}_p \triangleright \boldsymbol{e}_n} e_{n1} + (1 - w_{\boldsymbol{e}_p \triangleright \boldsymbol{e}_n})e_{n2} - \rho + M(1 - z_{\boldsymbol{e}_p \triangleright \boldsymbol{e}_n})$ for each $\boldsymbol{e}_p \triangleright \boldsymbol{e}_n, e_{p1} > e_{n1}$ (11.6)

$\hat{e}_2 \leq e_{r2} - \rho + Mz_{\boldsymbol{e}_l \triangleright \boldsymbol{e}_r}$ for each $\boldsymbol{e}_l \triangleright \boldsymbol{e}_r, e_{l1} < e_{r1}$ (11.7)

$w_{\boldsymbol{e}_l \triangleright \boldsymbol{e}_r}\hat{e}_1 + (1 - w_{\boldsymbol{e}_l \triangleright \boldsymbol{e}_r})\hat{e}_2 \leq w_{\boldsymbol{e}_l \triangleright \boldsymbol{e}_r} e_{r1} + (1 - w_{\boldsymbol{e}_l \triangleright \boldsymbol{e}_r})e_{r2} - \rho + M(1 - z_{\boldsymbol{e}_l \triangleright \boldsymbol{e}_r})$ for each $\boldsymbol{e}_l \triangleright \boldsymbol{e}_r, e_{l1} < e_{r1}$ (11.8)

$y_{it} \in \{0,1\}$ (11.9)

In (ALT'), the objective function and the first two constraints are the same with the objective function and the first two constraints of (ALT). (11.4) indicates that the estimated preference function value of the new solution should be at most as big as that of the incumbent. Remaining constraints are the cone constraints and are used to avoid inferior solutions.

In Step 2 of *QCX-u*, at the beginning of each round except for the first round, we apply the *SoS* procedure (explained in Section 4.3.2) to determine whether the best alternative up to the current round, $\boldsymbol{e}^{PREV}$, is an extreme supported solution or not. If it is extreme supported, we ask the DM to compare the incumbent and $\boldsymbol{e}^{PREV}$. If it is dominated, we eliminate $\boldsymbol{e}^{PREV}$; whereas if it is unsupported or nonextreme supported, we keep that solution until the end of the corresponding round. While finding some alternatives to ask the DM based on the version of the algorithm, we check whether $\boldsymbol{e}^{PREV}$ is in a cone-inferior region or not. If $\boldsymbol{e}^{PREV}$ is not in a cone-inferior region, we also ask the DM to compare the incumbent with $\boldsymbol{e}^{PREV}$ in Step 6.

In Step 3, to find adjacent alternatives of an incumbent we apply a similar procedure explained in Section 4.2. The only difference is that we reduce the search region using the cone constraints to avoid nonpreferable alternatives. Therefore in *QCX-u* we do not find an adjacent alternative that is previously asked to the DM, as we eliminate inferior solutions.

In Step 6 of *QCX-u*, we apply different procedures for different versions. We note that in this step we deal with unsupported solutions and therefore we use "neighbor" instead of "adjacent". We explain the procedures as follows:

Each Round Version

In the algorithm, we first deal with the supported solutions and at the beginning of Step 6 of *QCX-u*, we have the most preferred supported solution. During the algorithm, we reduce the search region using the preferences of the DM and continue our search in the reduced region. With this, we eliminate inferior alternatives and find unsupported solutions in the admissible search region.

In the single-round case, as shown in Section 4.4, the reduced space consists of two triangles. However, in multi-round case due to the cone constraints in the previous rounds, the reduced region may be smaller.

In the *each round* version, we conduct our search in the whole reduced region. We apply a similar procedure to that in Step 3. The only difference is that, in Step 6 we exclude the eastmost or westmost alternative and find an adjacent alternative to the incumbent in the reduced search region (i.e., in the triangle), and refer to this alternative as the "*neighbor*". After finding a neighbor to the incumbent in the search region, we ask the DM to compare them as usual and write the relevant cone constraints. We progress our search until there is no neighbor to be compared. We note that during our search, if the best alternative up to the current round, $\boldsymbol{e}^{PREV}$, is not found in a cone-inferior region, we also consider $\boldsymbol{e}^{PREV}$.

In this version, at the end of each round we find the most preferred alternative of the corresponding round since we consider the whole reduced region.

Last Round Version

In this version of the algorithm, until the final round of the auction, we only check whether $\boldsymbol{e}^{PREV}$ is in a cone-inferior region or not in Step 6. If $\boldsymbol{e}^{PREV}$ is not in a cone-inferior region, we also ask the DM to compare the incumbent and $\boldsymbol{e}^{PREV}$. With this we aim to hold on to the best alternative during rounds. In the final round, as in the *each round* version, we search the whole reduced region.

In the *each round* version, we find the most preferred alternative in each round by searching the whole reduced region. In the *last round* version, we aim to ask a smaller number of questions and therefore we do not search the reduced region in Step 6. However, in the final round we want to find the most preferred alternative and search the whole reduced region.

Band Version

In the *band* version, we search some portion of the reduced region in Step 6. By searching some part of the reduced region, we aim to find good alternatives. However, we also want to keep the number of questions asked to the DM low. Therefore, in each round instead of searching the whole reduced region, we systematically search some portion of the reduced region.

As stated before, in the multi-round case due to the cone constraints in the previous rounds, the reduced region may be smaller than the triangle(s). In this version, in early rounds we search small portions of the corresponding triangle(s) and in succeeding rounds we increase the searched portions of the triangles. We explain the procedure to define the east search region in each round.

We use the similarity of triangles to define the portion of the triangle to be searched. Suppose at the beginning of Step 6, we have the following search region in Figure 5.3 a.

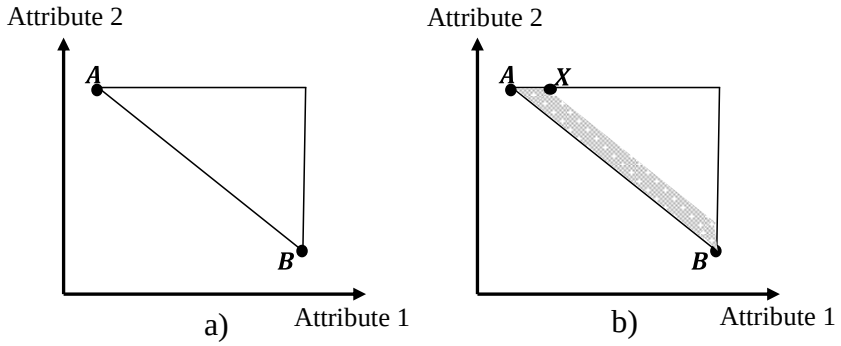


Figure 5.3 An example reduced region in band version

Let *Th* be the predetermined round number when the whole triangle will be searched and assume that we increase the search region the same amount in each iteration. Suppose that we are in round *h*, then we expect that the area of the dashed region to be h/Th of the total area of the triangle. We calculate point $\boldsymbol{X}$ to bound the search region where $X_2 = A_2$ and using similarity $X_1 = B_1 - (B_1 - A_1)\sqrt{h/Th}$. To find neighbor alternatives in Step 6, we apply a similar procedure as in *each round* version by adding the following constraint to bound the search region.

$w'\hat{e}_1 + (1 - w')\hat{e}_2 \leq w'X_1 + (1 - w')X_2$ where $w' = \frac{A_2 - B_2}{B_1 - B_2 - A_1 + A_2}$.

As can be seen from Figure 5.3.b, the reduced region is in the form of a band and hence it name *band*.

If $h \geq Th$ or if it is the final round of the auction, we search the whole triangle as in *each round* version. We note that while finding neighbor alternatives, we also consider the cone constraints and during our search, if the best alternative up to the current round, $\boldsymbol{e}^{PREV}$, is not found in a cone-inferior region, we also consider $\boldsymbol{e}^{PREV}$.

Limited Number of Questions Version

This version is similar to the *band* version. In the *band* version, if the solutions are crowded in a region, the number of solutions found in the determined portion of the triangle(s) may be large. Therefore, we develop *limited number of questions* version in which we limit the number of alternatives found in each round.

Similar to *band* version, we increase the number of alternatives found in the reduced region in each round. Let *LNQ* be the predetermined number used to limit the number of alternatives found in the triangle(s). In round *h*, we find at most *h* LNQ* alternatives (including the best alternative up to the current round) in the triangles.

In Step 7 of *QCX-u*, if it is not the final round, we solve (IR) to estimate the weight values and inform sellers with this information as well as the estimated preference function value of each item separately (for details please refer to Section 3.3). Then, the sellers update their bids and the algorithm continues with Step 2 of *QCX-u*.

5.2 Experimental Results for *QCX-u*

To test the performance of Q*CX-u*, we use 10 different weight values for the price attribute to generate different problems. We simulate the preferences of the buyer using weighted linear, weighted Euclidean and weighted Tchebycheff functions. Based on our preliminary experiments in *band* version we set *Th*, the predetermined round number that whole triangle will be searched, to 10 and in *limited number of questions* version *LNQ*, predetermined number used to limit the number of alternatives found in the triangle(s), to 1 and 2. We use *LNQ=1* and *LNQ=2* to refer to *limited number of questions* version with *LNQ* equal to1 and 2, respectively.

The average percent deviations and the average number of comparisons are reported in Tables 5.1 and 5.2, respectively.

Table 5.1 Average percentage deviations between the results of different versions of *QCX-u* and decentralized optimal solution*

	Underlying Utility Function		
Version	Weighted Linear	Weighted Euclidean	Weighted Tchebycheff
Each round	0.0490	-0.0532	-0.8943
Last round	0.0120	-0.0962	0.7273
Band	0.0171	-0.0251	-0.5145
LNQ=1	0.0250	-0.0723	-0.9277
LNQ=2	0.0258	-0.0532	-0.9039

* Based on 10 instances with different weight values

Table 5.2 Average number of comparisons w.r.t. different versions of *QCX-u**

	Underlying Utility Function		
Version	Weighted Linear	Weighted Euclidean	Weighted Tchebycheff
Each round	31.6	46.7	43.2
Last round	19.3	26.5	32.3
Band	39.3	70.1	64.3
LNQ=1	29.9	38.0	37.0
LNQ=2	32.5	43.1	40.9

* Based on 10 instances with different weight values

The results in Table 5.1 show that in all problems the percent deviations are very small in each version. Moreover, for underlying nonlinear preference functions, our algorithm performed better than the Decentralized case except for the *last round* version for underlying Tchebycheff function which is 0.7273%. The number of questions asked to the DM is smallest in *last round* and highest in *band* versions. In the *band* version we ask a relatively large number of questions, indicating that the unsupported alternatives in the defined band are dense. To decrease the number of questions asked, we can use the idea of Karahan and Köksalan (2010) and define territories around alternatives to get diverse pairs. As

expected, *LNQ=1* requires fewer questions than *LNQ=2* on the average. *Each round* requires more questions than *last round*, but the performances are not so different in terms of percent deviations. Although the percent deviations are very small in each version, considering both the percent deviations and the number of questions asked to the DM, we can say that *last round* and *LNQ=1* performs slightly better than other versions.

5.3 Modified Algorithm (L_α-u)

We modify *QCX-u* and develop L_α-*u* which estimates both alpha and weight values of the underlying preference function. In L_α-*u*, the buyer's preference function is estimated with an L_α function as stated in Section 2.5. Let α be the estimated parameter value of the underlying preference function of the DM. The algorithm starts with the linear case where $\alpha = 1$ and increases α as necessary.

Step 1: Set $\alpha = 1$, select an arbitrary set of weights and find an incumbent, $\boldsymbol{e}^*$, solving the following (ALT-α) model.

Parameters:

a_{itj}: level of attribute *j* offered by seller *i* in bid *t*

π_{itm}: 1 if bid *t* of seller *i* includes item *m*; 0 otherwise

T_i: the number of bids offered by seller *i*

w: estimated weight of attribute 1

α: estimated parameter value of the L_α function

Decision Variables:

y_{it}: 1 if bid *t* of seller *i* is selected to be in the efficient combination; 0 otherwise

$\hat{e}_j$: level of attribute *j* of the optimal alternative

Model (ALT-α)

$$\text{Min } (w\hat{e}_1)^\alpha + ((1-w)\hat{e}_2)^\alpha \qquad (12.1)$$

s.to

$$\sum_{i=1}^{I}\sum_{t=1}^{|T_i|}\pi_{itm}y_{it} \geq 1 \qquad \forall m \qquad (12.2)$$

$$\hat{e}_j = \sum_{i=1}^{I}\sum_{t=1}^{|T_i|}a_{itj}y_{it} \qquad \forall j \qquad (12.3)$$

$$(w\hat{e}_1)^\alpha + ((1-w)\hat{e}_2)^\alpha \leq (we_1^*)^\alpha + ((1-w)e_2^*)^\alpha \qquad (12.4)$$

$$y_{it} \in \{0,1\} \qquad (12.5)$$

In (ALT-α), if $\alpha = 1$, we solve a mixed-integer programming problem; otherwise we solve a mixed-integer nonlinear programming problem. We include constraint (12.4) in the model for the sake of completeness, but during the first time we search for an incumbent, we do not enforce (12.4). After finding the optimal solution of the problem, we set the solution to $\boldsymbol{e}^*$ and go to *Step 3*.

We note that although in (ALT') we use cone constraints, we do not use them in (ALT-α) due to Theorem 7.

Theorem 7: Let the underlying preference function be quasiconvex and let $\boldsymbol{e}^*$ and $\hat{\boldsymbol{e}}$ be the current best and optimal alternative of (ALT-α), respectively, estimate with a quasiconvex preference function *u*. Let $\boldsymbol{K}$ be the set of cone inferior solutions. Then $\hat{\boldsymbol{e}}$ cannot be cone inferior solution, i.e. $\hat{\boldsymbol{e}} \notin \boldsymbol{K}$.

Proof:

Suppose $\hat{\boldsymbol{e}} \in \boldsymbol{K}$, then by definition in Lemma 1 $u(\boldsymbol{e}^*) < u(\hat{\boldsymbol{e}})$ which contradicts constraint (12.4) of (ALT-α). □

We note that this situation may not be valid for (ALT'). The reason is that in *QCX-u* although the linearity assumption is violated we continue to estimate the preference function as linear. Therefore, the estimated preference function may not fit the preferences of the DM.

Step 2: Set $i \leftarrow i+1$. Find a solution, $\boldsymbol{e}_i$, solving (ALT-α). If it is the first round or there does not exist a solution, go to *Step 3*. Otherwise set the solution to $\boldsymbol{e}_i$. If a new round is started, check the status of $\boldsymbol{e}^*$ and if $\boldsymbol{e}^*$ is not extreme supported, set $\boldsymbol{e}^* = \boldsymbol{e}_i$ and go to *Step 3*. Otherwise, if $\boldsymbol{e}_{i1} < \boldsymbol{e}_1^*$, set *DIR=west*; otherwise set *DIR=east*. Ask the DM $\boldsymbol{e}^*$ versus $\boldsymbol{e}_i$. If

- $\boldsymbol{e}_i$ is preferred to $\boldsymbol{e}^*$, add a constraint $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_i) \geq \rho + \mu$, add $\boldsymbol{e}_i \rhd \boldsymbol{e}^*$, check the validity of the existing cones and write the relevant cone constraints. Set $\boldsymbol{e}^* = \boldsymbol{e}_i$ and go to *Step 3*.
- $\boldsymbol{e}^*$ is preferred to $\boldsymbol{e}_i$, add a constraint $\boldsymbol{w}(\boldsymbol{e}_i - \boldsymbol{e}^*) \geq \rho + \mu$, add $\boldsymbol{e}^* \rhd \boldsymbol{e}_i$, check the validity of the existing cones and write the relevant cone constraints. Switch the value of DIR (i.e. set *DIR=east* if it is equal to *west* and set *DIR=west* if it is equal to *east*) and go to *Step 3*.

Step 3: If *DIR=east*, find $\boldsymbol{e}_E$ and set $\boldsymbol{e}_{adj} = \boldsymbol{e}_E$; otherwise find $\boldsymbol{e}_W$ and set $\boldsymbol{e}_{adj} = \boldsymbol{e}_W$. If there exists $\boldsymbol{e}_{adj}$, go to *Step 4*. Otherwise, if $\alpha = 1$ go to *Step 3.1*; otherwise go to *Step 3.2*.

Step 3.1: If both $\boldsymbol{e}_E$ and $\boldsymbol{e}_W$ have been evaluated before, go to *Step 6*; otherwise, switch the value of *DIR* and go to *Step 3*.

Step 3.2: If incumbent has no neighbor alternative, solve (Walpha) and go to *Step 8*; otherwise, switch the value of *DIR* and go to *Step 3*.

Step 4: Ask the DM $\boldsymbol{e}^*$ versus $\boldsymbol{e}_{adj}$. If

- $\boldsymbol{e}_{adj}$ is preferred to $\boldsymbol{e}^*$, add a constraint $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_{adj}) \geq \rho + \mu$, add $\boldsymbol{e}_{adj} \rhd \boldsymbol{e}^*$, check the validity of the existing cones and write the relevant cone constraints. Set $\boldsymbol{e}^* = \boldsymbol{e}_{adj}$, if $\alpha = 1$ go to *Step 5*; otherwise switch the value of *DIR* and go to *Step 3*.
- $\boldsymbol{e}^*$ is preferred to $\boldsymbol{e}_{adj}$, add a constraint $\boldsymbol{w}(\boldsymbol{e}_{adj} - \boldsymbol{e}^*) \geq \rho + \mu$, add $\boldsymbol{e}^* \rhd \boldsymbol{e}_{adj}$, check the validity of the existing cones and write the relevant cone constraints. If $\alpha = 1$ and both $\boldsymbol{e}_E$ and $\boldsymbol{e}_W$ have been evaluated, go to *Step 6*. Otherwise switch the value of *DIR* and go to *Step 3*.

Step 5: Find a feasible set of weights satisfying all constraints corresponding to past preferences of the DM for the corresponding feasible α by solving the following (Walpha) model.

Parameters:

ρ: a small positive constant

e_{kj}: level of attribute *j* in efficient combination *k*

α: estimated parameter value of the L_α function

Decision Variables:

μ : an auxiliary variable (to measure the estimated value difference between alternatives and bound the weights)

$\hat{w}_j$: estimated weight of attribute *j*

Model (Walpha)

Max μ (13.1)

s.to

$\mu \leq \hat{w} \leq 1 - \mu$ (13.2)

$u(\boldsymbol{e}_k) = ((\hat{w}e_{k1})^\alpha + ((1-\hat{w})e_{k2})^\alpha)^{1/\alpha}$ (13.3)

$u(\boldsymbol{e}_n) \geq u(\boldsymbol{e}_p) + \rho + \mu$ for each $\boldsymbol{e}_p \succ \boldsymbol{e}_n$ (13.4)

$\mu \geq 0$ (13.5)

The (Walpha) model is similar to (Wt) model except that in (Wt) we set $\alpha = 1$ and only estimate the weight values. On the other hand, in (Walpha) we estimate both α and the corresponding weight values. For given α values we solve (Walpha) model. As in Karakaya and Köksalan (2011), we take the smallest α value to fit a function satisfying all constraints but having the least curvature. At the beginning we set α to 1 (i.e. we start with a weighted linear preference function) and increase α by 1 if necessary.

After finding the estimated parameters α and $\boldsymbol{w}$, if $\alpha = 1$ go to *Step 2*; otherwise switch the value of DIR and go to *Step 3*.

Step 6: If $\boldsymbol{e}^{PREV}$ is in a cone-inferior region, go to *Step 7*. Otherwise, if $e_1^{PREV} < \boldsymbol{e}_1^*$, set *DIR=west*; otherwise set *DIR=east*. Ask the DM $\boldsymbol{e}^*$ versus $\boldsymbol{e}^{PREV}$. If

- $\boldsymbol{e}^{PREV}$ is preferred to $\boldsymbol{e}^*$, add a constraint $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}^{PREV}) \geq \rho + \mu$, add $\boldsymbol{e}^{PREV} \rhd \boldsymbol{e}^*$, check the validity of the existing cones and write the relevant cone constraints. Set $\boldsymbol{e}^* = \boldsymbol{e}^{PREV}$ and go to *Step 7*.
- $\boldsymbol{e}^*$ is preferred to $\boldsymbol{e}^{PREV}$, add a constraint $\boldsymbol{w}(\boldsymbol{e}^{PREV} - \boldsymbol{e}^*) \geq \rho + \mu$, add $\boldsymbol{e}^* \rhd \boldsymbol{e}^{PREV}$, check the validity of the existing cones and write the relevant cone constraints. Switch the value of DIR and go to *Step 7*.

Step 7: Solve (Walpha). If $\alpha > 1$, go to *Step 3*; otherwise go to *Step 8*.

Step 8: If it is the final round, go to *Step 9*; otherwise solve (IR) to inform the sellers, get the updated bids and go to *Step 2*.

Step 9: Stop.

In Step 2 of L_α-*u*, if a new round is started while checking the status of the best alternative up to the current round, $\boldsymbol{e}^{PREV}$, if $\alpha = 1$, we follow the procedure in *QCX-u*. Else if $\alpha > 1$, we check the dominance of $\boldsymbol{e}^{PREV}$, by solving the following (DOM) model.

Parameters:

a_{itj}: level of attribute *j* offered by seller *i* in bid *t*

π_{itm}: 1 if bid *t* of seller *i* includes item *m*; 0 otherwise

T_i: the number of bids offered by seller *i*

Decision Variables:

y_{it}: 1 if bid *t* of seller *i* is selected to be in the efficient combination; 0 otherwise

$\hat{e}_j$: level of attribute *j* of the alternative

μ_j: the difference between $\hat{\boldsymbol{e}}$ and $\boldsymbol{e}^{PREV}$ in attribute *j*

Model (DOM)

Max $\mu_1 + \mu_2$ (14.1)

s.to

$\sum_{i=1}^{I} \sum_{t=1}^{|T_i|} \pi_{itm} y_{it} \geq 1$ $\forall m$ (14.2)

$\hat{e}_j = \sum_{i=1}^{I} \sum_{t=1}^{|T_i|} a_{itj} y_{it}$ $\forall j$ (14.3)

$\hat{e}_1 \leq e_1^{PREV} - \mu_1$ (14.4)

$\hat{e}_2 \leq e_2^{PREV} - \mu_2$ (14.5)

$\mu_1 \geq 0$ (14.6)

$\mu_2 \geq 0$ (14.7)

$y_{it} \in \{0,1\}$ (14.8)

In (DOM), we search for an alternative that dominates $\boldsymbol{e}^{PREV}$. Constraint set (14.2) guarantees to satisfy the demand for each item. (14.4) and (14.5) are used to force the resulting solution to dominate $\boldsymbol{e}^{PREV}$. If the problem is optimal with a positive objective function value, we conclude that $\boldsymbol{e}^{PREV}$ is dominated and we eliminate $\boldsymbol{e}^{PREV}$. Otherwise, we say that there is no solution dominating $\boldsymbol{e}^{PREV}$, and we ask the DM to compare the incumbent and $\boldsymbol{e}^{PREV}$ in Step 2.

In Step 3, if $\alpha = 1$ we apply the procedure in Step 3 of *QCX-u*; whereas if $\alpha > 1$ we apply the procedure in Step 6 of *QCX-u* with the following additional constraints:

If *DIR=east*

$\hat{e}_2 \leq e_2^* - \rho$

If *DIR=west*

$\hat{e}_1 \leq e_1^* - \rho$

Based on the direction of the search we add a new constraint. The reason is that in L_α*-u*, the incumbent neighbor of which is searched can be an unsupported solution. If we do not consider this additional constraint, although the direction is

east(west), we may end up a dominated solution or a solution in the west(east) of the incumbent.

In the case that $\alpha > 1$, there are different ways of finding neighbor alternatives. For instance, we can find the alternatives those are closest to the incumbent in each attribute as neighbor alternatives. Keeping the other steps as they are, we apply this procedure in Step 3 if $\alpha > 1$ and call this version as L_α-*NN* to indicate that the alternatives found are the nearest neighbors in each attribute.

We note that during our neighbor alternative search, if the best alternative up to the current round, $\boldsymbol{e}^{PREV}$, is not found in a cone-inferior region, we also consider $\boldsymbol{e}^{PREV}$.

In the algorithm, as in *QCX-u* after solving (IR) we inform sellers about the estimated weight values of the linear function as well as the estimated preference function value of each item separately in Step 8.

Modifications for underlying Tchebycheff Preference Functions

The L_α-*u* algorithm is a general algorithm for underlying quasiconvex preference functions. If we know that the DM has a Tchebycheff preference function at the beginning or at any step of the algorithm, we use the properties of the Tchebycheff functions and make some modifications to L_α-*u* .

Modifications in (ALT_α)

We solve the following model to find a new solution based on the estimated Tchebycheff function.

Parameters:

ρ: a small positive constant

a_{itj}: level of attribute *j* offered by seller *i* in bid *t*

π_{itm}: 1 if bid *t* of seller *i* includes item *m*; 0 otherwise

T_i: the number of bids offered by seller *i*

w: estimated weight of attribute 1

$u(\boldsymbol{e}^*)$: estimated Tchebycheff function value of $\boldsymbol{e}^*$ where $u(\boldsymbol{e}^*) = max\{we_1^*, (1-w)e_2^*\}$

Decision Variables:

z : the Tchebycheff distance value of the solution

y_{it}: 1 if bid *t* of seller *i* is selected to be in the efficient combination; 0 otherwise

$\hat{e}_j$: level of attribute *j* of the optimal alternative

Model (ALT_∞)

$$\text{Min } z + \rho(\hat{e}_1 + \hat{e}_2) \qquad (15.1)$$

s.to

$$\sum_{i=1}^{I}\sum_{t=1}^{|T_i|}\pi_{itm}y_{it} \geq 1 \qquad \forall m \qquad (15.2)$$

$$\hat{e}_j = \sum_{i=1}^{I}\sum_{t=1}^{|T_i|}a_{itj}y_{it} \qquad j = 1,2 \qquad (15.3)$$

$$z \geq w\hat{e}_1 \qquad (15.4)$$

$$z \geq (1-w)\hat{e}_2 \qquad (15.5)$$

$$z \leq u(\boldsymbol{e}^*) \qquad (15.6)$$

$$y_{it} \in \{0,1\} \qquad (15.7)$$

In (ALT_∞), we try to minimize the weighted Tchebycheff distance. To avoid weakly nondominated but dominated solutions, we use the augmented part in objective function. A suitable ρ value should be selected to make sure that the second term in the objective function does not cause any trade-offs with the first term, and only has an effect of breaking ties. Constraint (15.6) indicates that the estimated preference function value of the new solution should not be worse than that of the incumbent.

Instead of using an augmented part in the objective function, we solve the problem in a lexicographic manner. First, we minimize z. By fixing the value of

the attribute whose value multiplied by the corresponding weight is equal to z, we then solve the problem again to minimize the other attribute.

Modifications in Cone Constraints

In L_α-u we write the general cone constraints. However, for Tchebycheff preference function case we modify the cone constraints and reduce the search space accordingly.

Consider two alternatives $\boldsymbol{A}$ and $\boldsymbol{B}$ where the DM prefers $\boldsymbol{A}$ to $\boldsymbol{B}$. We write the cone, $\boldsymbol{A} \rhd \boldsymbol{B}$, and the corresponding cone constraint as follows:

$\hat{e}_1 \leq B_1 - \rho$

With this constraint we eliminate all the alternatives having attribute 1 value of B_1 or more. Compared to the general cone constraints, here we eliminate a greater region and we do not need to use binary variables.

Modifications in Finding Neighbor Alternatives

As stated above, we try L_α-*NN* version where we find the alternatives those are closest to the incumbent in each attribute as neighbor alternatives when α is estimated as greater than 1. This method is beneficial when the underlying preference function is Tchebycheff.

To illustrate consider the alternatives in Figure 5.4. Suppose $\boldsymbol{B}$ is found as incumbent by solving (ALT_∞) and then we look for east neighbor alternative of $\boldsymbol{B}$. If we apply the procedure in Step 6 of *QCX-u*, we will find $\boldsymbol{D}$; whereas if we find the nearest neighbor in attribute 1, we will find $\boldsymbol{C}$. If $\boldsymbol{B}$ is preferred to its east neighbor, with selecting $\boldsymbol{C}$ as east neighbor more search region will be eliminated than selecting $\boldsymbol{D}$. On the other hand, if east neighbor is preferred to $\boldsymbol{B}$, the reduced search region will be the same. Since $\boldsymbol{B}$ is found as incumbent based on the estimations, $\boldsymbol{B}$ is likely to be preferred to its east neighbor. Therefore, we say that

finding the nearest neighbor of incumbent in each attribute is more suitable when the underlying preference function is Tchebycheff.

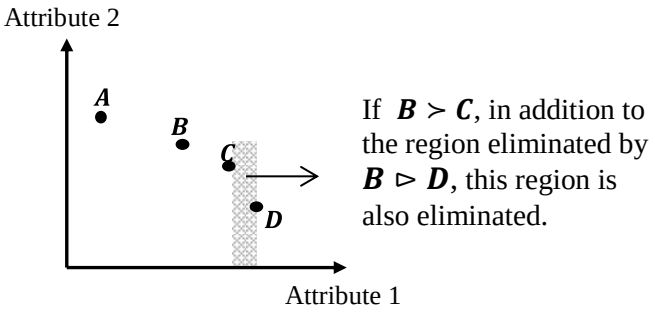


Figure 5.4 Search space reduction with estimated Tchebycheff functions

Modifications in (Walpha)

Since we assume that the underlying preference function is Tchebycheff, we only estimate the weight values by solving the following (Wt_∞) model:

Parameters:

ρ: a small positive constant

e_{kj}: level of attribute *j* in efficient combination *k*

Decision Variables:

μ : an auxiliary variable (to measure the estimated value difference between alternatives and bound the weights)

$\hat{w}_j$: estimated weight of attribute *j*

Model (Wt_∞)

Max μ (16.1)

s.to

$\mu \leq \widehat{w} \leq 1 - \mu$ (16.2)

$(1 - \widehat{w})e_{n2} \geq \widehat{w}e_{p1} + \rho + \mu$ for each $\boldsymbol{e}_p \succ \boldsymbol{e}_n$ and $e_{p1} > e_{n1}$ (16.3)

$(1 - \widehat{w})e_{n2} \geq (1 - \widehat{w})e_{p2} + \rho + \mu$ for each $\boldsymbol{e}_p \succ \boldsymbol{e}_n$ and $e_{p1} > e_{n1}$ (16.4)

$\widehat{w}e_{r1} \geq \widehat{w}e_{l1} + \rho + \mu$ for each $\boldsymbol{e}_l \succ \boldsymbol{e}_r$ and $e_{l1} < e_{r1}$ (16.5)

$\widehat{w}e_{r1} \geq (1 - \widehat{w})e_{l2} + \rho + \mu$ for each $\boldsymbol{e}_l \succ \boldsymbol{e}_r$ and $e_{l1} < e_{r1}$ (16.6)

$\mu \geq 0$ (16.7)

The objective (16.1) and the first constraint of (Wt_∞) are the same as the objective function and the first constraint of (Walpha). We only modify the preference constraints (16.3-16.6) considering the Tchebycheff function. In these constraints we reduce the feasible weight space using the preferences of the buyer as well as the attribute values of the alternatives in each pairwise comparison (see Bozkurt et al. 2010 for a general coverage of weight space reduction for Tchebycheff functions).

We use the weights found by (Wt_∞), to find a new incumbent in Step 2. We apply the L_α-u algorithm by starting a linear estimated preference functions. As the rounds progresses, if the estimated parameter of the underlying preference function, α, is greater than a predetermined threshold value, $T\alpha$, we assume that the underlying preference function is Tchebycheff. The reason is that as the α of the L_α function increases, it converges to Tchebycheff function. To utilize the properties of the Tchebycheff functions, if α is greater than $T\alpha$, we assume the that the underlying preference function is Tchebycheff and we apply modifications stated above.

If the underlying preference function of the DM is linear, L_α-u algorithm deals with the supported efficient solutions only. On the other hand if the exact parameter of the underlying preference function is greater than 1, we expect to

capture this with the preferences of the DM and utilize properties of the L_α-u algorithm. To understand the curvature of the underlying preference function of the DM (i.e., if the underlying preference function is not linear, we aim to detect it as soon as possible) at the early rounds in addition to the supported solutions we also consider unsupported solutions. We set a threshold number $Tquestion$ above which we stop searching for unsupported solutions if the estimated parameter of the underlying preference function, α, is 1. That is, although we estimate a linear function, we continue searching as in Step 6 of *QCX-u* (searching the triangles) to understand whether the underlying preference is linear or not. By asking about unsupported solutions we aim to rule out the linearity assumption if the underlying preference function is not linear. While doing this, to avoid high number of comparisons, we limit the number of alternatives found at the end of each round by searching the triangles. If the total number of unsupported solutions visited is $Tquestion$ and the linearity assumption still holds, we stop searching the triangles.

Understanding the curvature of the underlying preference function is important as the algorithm can be more beneficial with this information. To achieve this, we consider unsupported solutions in addition to the supported solutions. Different methods can be tried to understand the form of the underlying preference function (see for example Köksalan and Sagala 1995).

5.4 Results

We test the performance of the L_α-u algorithm using 10 problems generated with different weight values for the price attribute. We simulate the preferences of the buyer using weighted linear, weighted Euclidean and weighted Tchebycheff functions. We consider two versions of the algorithm: L_α-uQ where the neighbor alternative search is as in Step 6 of *QCX-u* when $\alpha > 1$ and L_α-*NN* where the alternatives closest to the incumbent in each attribute are found as neighbor alternatives when $\alpha > 1$. We set the predetermined threshold value for α , above which we assume that the underlying preference function is Tchebycheff to $T\alpha$=4

and the number of questions above which we stop searching for unsupported solutions if $\alpha = 1$ to $Tquestion$=10 based on our preliminary experiments.

The average percent deviations and the average number of comparisons are provided in Tables 5.3 and 5.4, respectively.

Table 5.3 Average percentage deviations between the results of different versions of L_α-u and decentralized optimal solution*

	Underlying Preference Function		
Version	Weighted Linear	Weighted Euclidean	Weighted Tchebycheff
L_α-uQ	0.0248	-0.0196	-0.6122
L_α-NN	0.0248	-0.0196	-0.2858

* Based on 10 instances with different weight values

Table 5.4 Average number of comparisons w.r.t. different versions of L_α-u*

	Underlying Preference Function		
Version	Weighted Linear	Weighted Euclidean	Weighted Tchebycheff
L_α-uQ	25.8	38.8	31.3
L_α-NN	25.8	38.8	26.0

* Based on 10 instances with different weight values

In all problems, the percent deviations are very small in both versions of L_α-u. Indeed, when the underlying preference function is weighted linear, both versions of the algorithm are exactly the same. The algorithms only differ in Step 3 when the estimated parameter value, α, is greater than one. The algorithms are identical for underlying weighted linear functions as α is 1 in this case. Although the number of questions asked to the DM varies slightly in some problems, they are the same on the average for underlying weighted Euclidean preference functions. As expected when the underlying function is weighted Tchebycheff, the L_α-NN version requires fewer questions.

For the sake of completeness in the following tables we provide the results of all versions of *QCX-u* and L_α*-u* together.

Table 5.5 Average percentage deviations between the results of algorithms and decentralized optimal solution*

	Underlying Preference Function		
Version	Weighted Linear	Weighted Euclidean	Weighted Tchebycheff
Each round	0.0490	-0.0532	-0.8943
Last round	0.0120	-0.0962	0.7273
Band	0.0171	-0.0251	-0.5145
LNQ=1	0.0250	-0.0723	-0.9277
LNQ=2	0.0258	-0.0532	-0.9039
L_α*-uQ*	0.0248	-0.0196	-0.6122
L_α*-NN*	0.0248	-0.0196	-0.2858

* Based on 10 instances with different weight values

Table 5.6 Average number of comparisons in different versions of the algorithms*

	Underlying Preference Function		
Version	Weighted Linear	Weighted Euclidean	Weighted Tchebycheff
Each round	31.6	46.7	43.2
Last round	19.3	26.5	32.3
Band	39.3	70.1	64.3
LNQ=1	29.9	38.0	37.0
LNQ=2	32.5	43.1	40.9
L_α*-uQ*	25.8	38.8	31.3
L_α*-NN*	25.8	38.8	26.0

* Based on 10 instances with different weight values

The results show that in each version of each algorithm the percent deviations are very small, i.e. our algorithms perform well. As stated before *last round* and *LNQ=1* performs slightly better than other versions in *QCX-u*. L_α*-NN* which is a general algorithm for underlying quasiconvex preference functions also performs

well among others. In L_α-*NN* version, even if we estimate the α value to be greater than 1, while informing sellers we treat it as 1 as it is not straightforward to figure out the contributions of different sellers of a combination of bids to the overall value of the bid combination when α>1. There can be room for improvement by informing the sellers about the estimated α value, rather than treating as if α =1. How this information can be utilized awaits future research.

CHAPTER 6

AN INTERACTIVE APPROACH FOR COORDINATED BIDDING

In the previous chapters, we assume that we do not know the cost functions of the sellers. In this chapter, we assume that all sellers disclose their cost functions to the auction decision support system. By using their cost functions, we create good combinations to present the buyer. We refer to this case as "*Coordinated Bidding*" and we develop an interactive algorithm for this case in Section 6.1. We discuss the algorithm considering a discretized search space in Section 6.2 and we provide experimental results in Section 6.3.

6.1 The Interactive Algorithm (*CO-u*)

We develop an interactive algorithm, *CO-u* that finds good combinations knowing the cost functions of the sellers, when the buyer's preference function is quasiconvex and there are two attributes. As in Chapter 5, we assume that the buyer can distinguish between bids even when their preference function values are close. Similar to L_α-*u*, we estimate both alpha and weight values of the underlying preference function.

We assume that the sellers give their initial bids at the beginning of the auction. After finding the most preferred supported bid combination using these bids, we then compose bid combinations using the sellers' cost functions. Our algorithm

continues until there is no alternative bid combination with predetermined mark-up percentages.

The steps of CO-u

In addition to the notation used in previous chapters, we define $\boldsymbol{e}_{ch}$ and w_{ch} as a challenger alternative to the incumbent and the estimated weight of attribute 1 used to find $\boldsymbol{e}_{ch}$, respectively.

We provide the steps of the algorithm as follows:

Step 1: Apply *Steps 1-5* of *QCX-u* to find the most preferred supported alternative, $\boldsymbol{e}^*$.

Step 2: Find a feasible set of weights satisfying all constraints corresponding to past preferences of the DM for the corresponding feasible α by solving (Walpha) model. Set $\boldsymbol{e}^{PREV} = \boldsymbol{e}^*$ and go to *Step 3*.

Step 3: Find a solution by solving the (Min_u) model below.

Parameters:

π_{itm}: 1 if bid *t* of seller *i* includes item *m*; 0 otherwise

T_i: the number of bids offered by seller *i*

w: estimated weight of attribute 1

α: estimated parameter value of the L_α function

$f_{it}(d_{it})$: cost function of seller *i* for bid *t* for given d_{it}

v_i^{perc}: minimum mark-up percentage for seller *i*; if it is 0, then seller *i* may bid with zero profit. For the sake of simplicity let $v_i = v_i^{perc}/100$.

Decision Variables:

y_{it}: 1 if bid *t* of seller *i* is selected to be in the efficient combination; 0 otherwise

$\hat{d}_{it}$: level of defect rate suggested to seller *i* for bid *t*

$\hat{e}_j$: level of attribute *j* of the optimal alternative

Model (Min_u)

$$\text{Min } ((w\hat{e}_1)^{\alpha} + ((1-w)\hat{e}_2)^{\alpha})^{1/\alpha} \quad (17.1)$$

s.to

$$\sum_{i=1}^{I}\sum_{t=1}^{T_i} \pi_{itm} y_{it} \geq 1 \qquad \forall m \quad (17.2)$$

$$\hat{e}_1 = \sum_{i=1}^{I}\sum_{t=1}^{T_i} f_{it}(\hat{d}_{it})(1+v_i)y_{it} \quad (17.3)$$

$$\hat{e}_2 = \sum_{i=1}^{I}\sum_{t=1}^{T_i} \hat{d}_{it} y_{it} \quad (17.4)$$

$$y_{it} \in \{0,1\} \quad (17.5)$$

By solving (Min_u) we find the combination with minimum estimated preference function value. (Min_u) is always feasible when there are bids to satisfy the demand constraint. Set the optimal solution of the problem to $\boldsymbol{e}^*$ and go to *Step 4*.

Step 4: If *DIR=east*, set $w_{ch} = \frac{w+WL}{2}$; otherwise set $w_{ch} = \frac{w+WU}{2}$. Solve the following (Challenger) model to find an alternative, $\boldsymbol{e}_{ch}$.

Parameters:

M: a big number

ρ: small positive constant

π_{itm}: 1 if bid *t* of seller *i* includes item *m*; 0 otherwise

T_i: the number of bids offered by seller *i*

w_{ch}: estimated weight of attribute 1 to find a challenger alternative

$w_{\boldsymbol{A}\triangleright\boldsymbol{B}}$: calculated weight of attribute 1 where $w_{\boldsymbol{A}\triangleright\boldsymbol{B}} = \frac{(A_2-B_2)}{(B_1-B_2-A_1+A_2)}$

α: estimated parameter value of the L_α function

$f_{it}(d_{it})$: cost function of seller *i* for bid *t* for given d_{it}

v_i^{perc}: minimum mark-up percentage for seller *i*; if it is 0, then seller *i* may bid with zero profit. For the sake of simplicity let $v_i = v_i^{perc}/100$.

Decision Variables:

y_{it}: 1 if bid *t* of seller *i* is selected to be in the efficient combination; 0 otherwise

$\hat{d}_{it}$: level of defect rate suggested to seller *i* for bid *t*

$\hat{e}_j$: level of attribute *j* of the optimal alternative

$z_{\boldsymbol{e}_p \triangleright \boldsymbol{e}_n}$: 1 if constraint (18.6) is active; 0 otherwise

$z_{\boldsymbol{e}_l \triangleright \boldsymbol{e}_r}$:1 if constraint (18.8) is active; 0 otherwise

Model (Challenger)

Min $((w_{ch}\hat{e}_1)^{\alpha} + ((1 - w_{ch})\hat{e}_2)^{\alpha})^{1/\alpha}$ (18.1)

s.to

$\sum_{i=1}^{I}\sum_{t=1}^{T_i} \pi_{itm} y_{it} \geq 1$ $\forall m$ (18.2)

$\hat{e}_1 = \sum_{i=1}^{I}\sum_{t=1}^{T_i} f_{it}(\hat{d}_{it})(1 + v_i) y_{it}$ (19.3)

$\hat{e}_2 = \sum_{i=1}^{I}\sum_{t=1}^{T_i} \hat{d}_{it} y_{it}$ (18.4)

$\hat{e}_1 \leq e_{n1} - \rho + M z_{\boldsymbol{e}_p \triangleright \boldsymbol{e}_n}$ for each $\boldsymbol{e}_p \triangleright \boldsymbol{e}_n$, $e_{p1} > e_{n1}$ (18.6)

$w_{\boldsymbol{e}_p \triangleright \boldsymbol{e}_n}\hat{e}_1 + (1 - w_{\boldsymbol{e}_p \triangleright \boldsymbol{e}_n})\hat{e}_2 \leq w_{\boldsymbol{e}_p \triangleright \boldsymbol{e}_n} e_{n1} + (1 - w_{\boldsymbol{e}_p \triangleright \boldsymbol{e}_n}) e_{n2} - \rho + M(1 -$

$z_{\boldsymbol{e}_p \triangleright \boldsymbol{e}_n})$ for each $\boldsymbol{e}_p \triangleright \boldsymbol{e}_n$, $e_{p1} > e_{n1}$ (18.7)

$\hat{e}_2 \leq e_{r2} - \rho + M z_{\boldsymbol{e}_l \triangleright \boldsymbol{e}_r}$ for each $\boldsymbol{e}_l \triangleright \boldsymbol{e}_r$, $e_{l1} < e_{r1}$ (18.8)

$w_{\boldsymbol{e}_l \triangleright \boldsymbol{e}_r}\hat{e}_1 + (1 - w_{\boldsymbol{e}_l \triangleright \boldsymbol{e}_r})\hat{e}_2 \leq w_{\boldsymbol{e}_l \triangleright \boldsymbol{e}_r} e_{r1} + (1 - w_{\boldsymbol{e}_l \triangleright \boldsymbol{e}_r}) e_{r2} - \rho + M(1 - z_{\boldsymbol{e}_l \triangleright \boldsymbol{e}_r})$

for each $\boldsymbol{e}_l \triangleright \boldsymbol{e}_r$, $e_{l1} < e_{r1}$ (18.9)

$y_{it} \in \{0,1\}$ (18.10)

The objective is to minimize the estimated preference function using the updated weight values. Constraints (18.6) - (18.9) are the cone constraints and are used to avoid inferior solutions. Moreover, to restrict the search region based on the direction, we add the following constraints:

If *DIR=east*

$\hat{e}_1 \geq e_1^*$

$\hat{e}_2 \leq e_2^* - \rho$

If *DIR=west*

$\hat{e}_1 \leq e_1^* - \rho$

$\hat{e}_2 \geq e_2^*$

We eliminate $\boldsymbol{e}^*$ with these constraints. Therefore, if there exists an optimal solution set $\boldsymbol{e}_{ch}$ to that solution and go to *Step 5*. Otherwise, if both directions have been evaluated before, go to *Step 6*; otherwise switch the value of *DIR* and go to *Step 4*.

Step 5: Ask the DM $\boldsymbol{e}^*$ versus $\boldsymbol{e}_{ch}$. If

- $\boldsymbol{e}_{ch}$ is preferred to $\boldsymbol{e}^*$, add a constraint $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}_{ch}) \geq \rho + \mu$, add $\boldsymbol{e}_{ch} \rhd \boldsymbol{e}^*$, check the validity of the existing cones and write the relevant cone constraints. Set $\boldsymbol{e}^* = \boldsymbol{e}_{ch}$ and go to *Step 6*.
- $\boldsymbol{e}^*$ is preferred to $\boldsymbol{e}_{ch}$, add a constraint $\boldsymbol{w}(\boldsymbol{e}_{ch} - \boldsymbol{e}^*) \geq \rho + \mu$, add $\boldsymbol{e}^* \rhd \boldsymbol{e}_{ch}$, check the validity of the existing cones and write the relevant cone constraints. If both directions have been evaluated before, go to *Step 6*; otherwise switch the value of *DIR* and go to *Step 4*.

Step 6: If $\boldsymbol{e}^{PREV}$ is in a cone-inferior region, go to *Step 7*. Otherwise, ask the DM $\boldsymbol{e}^*$ versus $\boldsymbol{e}^{PREV}$ and if

- $\boldsymbol{e}^{PREV}$ is preferred to $\boldsymbol{e}^*$, add a constraint $\boldsymbol{w}(\boldsymbol{e}^* - \boldsymbol{e}^{PREV}) \geq \rho + \mu$, add $\boldsymbol{e}^{PREV} \rhd \boldsymbol{e}^*$, check the validity of the existing cones and write the relevant cone constraints. Set $\boldsymbol{e}^* = \boldsymbol{e}^{PREV}$ and go to *Step 2*.
- $\boldsymbol{e}^*$ is preferred to $\boldsymbol{e}^{PREV}$, add a constraint $\boldsymbol{w}(\boldsymbol{e}^{PREV} - \boldsymbol{e}^*) \geq \rho + \mu$, add $\boldsymbol{e}^* \rhd \boldsymbol{e}^{PREV}$ check the validity of the existing cones and write the relevant cone constraints. Go to *Step 2*.

Step 7: If incumbent has no challenger alternative, stop $\boldsymbol{e}^*$ is the most preferred alternative. Otherwise go to *Step 2*.

Similar to the previous versions, at the beginning of the auction, sellers first give their bids. In Step 1 we find the most preferred supported alternative. Based on the preference of the buyer, (Walpha) model is solved. Here we note that since we deal with supported solutions in Step, (Wt) model can also be solved.

After estimating the parameters of the preference function of the buyer, in Step 3 considering the sellers' mark-up percentages we find a combination in the reduced search space that minimized the estimated preference function value. The

algorithm continues with Step 4 where we search for challenger alternatives in different directions with different weight values.
We note that since the problems are nonlinear, instead of using binary variables we can use nonnegative y_{it} values with the following constraint:

$$y_{it}(1 - y_{it}) = 0$$

As in L_α-u, to utilize the properties of the Tchebycheff functions, if α is greater than the predetermined threshold value, $T\alpha$, we assume that the underlying preference function is Tchebycheff and modify our models.

In the algorithm, except from Step 1, while finding a new bid combination we use the cost functions and the mark-up percentages of the sellers as we assume that they explicitly give them to us. At the end of the auction, each winner will get the profit with his/her predetermined mark-up percentage.

We try to solve these models using GAMS 22.8 and the global optimization solver, BARON. However, due to nonlinearity in a continuous space, the run time of the algorithm turns out to be long. Moreover, for some problems BARON could not find solutions. Therefore we discretize the bid space and apply the *CO-u* algorithm in Section 6.2.

6.2 *CO-u* in Discretized Space

In Section 6.1 we provide the models considering a continuous search space. Here, we assume a discretized space which is reasonable in real life examples. For each bid of each seller, we divide the possible defect rate range into *K* equal intervals. Therefore, we consider *K+1* possible defect rate-price value combination for each seller in all his/her bids. In our updated models we consider only these values. Here, we provide the modified parameters and decision variables only. For the sake of completeness, we provide the updated (Min_u) model. The changes are similar for (Challenger) model.

Modifications in (Min_u)

Parameters:

$\boldsymbol{f_{it}(d_{itk})}$**:** cost function of seller *i* for bid *t* for given d_{itk}

$\boldsymbol{d_{itk}}$: defect rate value corresponding to the k^{th} point in bid *t* of seller *i*

Decision Variables:

$\boldsymbol{y_{itk}}$: 1 if bid *t* of seller *i* with defect rate value of point *k* is selected to be in the efficient combination; 0 otherwise

Model (Min_u')

$$\text{Min } (w\hat{e}_1)^{\alpha} + ((1-w)\hat{e}_2)^{\alpha} \qquad (17'.1)$$

s.to

$$\sum_{k=1}^{K}\sum_{i=1}^{I}\sum_{t=1}^{T_i} \pi_{itm} y_{itk} \geq 1 \qquad \forall m \qquad (17'.2)$$

$$\hat{e}_1 = \sum_{k=1}^{K}\sum_{i=1}^{I}\sum_{t=1}^{T_i} f_{it}(d_{itk}) y_{itk} \qquad (17'.3)$$

$$\hat{e}_2 = \sum_{k=1}^{K}\sum_{i=1}^{I}\sum_{t=1}^{T_i} d_{itk} y_{itk} \qquad (17'.4)$$

$$y_{itk} \in \{0,1\} \qquad (17'.5)$$

We use GAMS 22.8 and the global optimization solver, BARON to solve the updated model. We next provide the experiment results.

6.3 Results

To test the performance of the *CO-u* algorithm in discretized space, we solve (Min_u) with exact parameter values and report its results as "Centralized." This corresponds to the best possible solutions that can be obtained under full information. Similar to the previous chapters we compare the preference function values of the buyer for the winning combination found with the algorithm against that of Centralized. To evaluate the performance of the algorithm for these values we use % deviations:

$$\%\text{ deviation} = \frac{u(final\_combination\_algorithm) - u(Centralized)}{u(Centralized)} 100$$

In our experiments, without loss of generality we set the mark-up percentages to zero. We set K=10, i.e. we divide the possible defect rate range into 10 equal intervals and there are 11 possible defect rate-price value combination for all sellers in all their bids. We set $T\alpha$, the predetermined threshold value to 4, above which we treat the underlying preference function as a Tchebycheff function.

We use 10 different weight values for the price attribute of the underlying preference function to consider different problems. We simulate the preferences of the buyer using weighted linear, weighted Euclidean and weighted Tchebycheff functions.

In all problems, the winning seller-bid pairs and the corresponding attribute values found with *CO-u* and Centralized are the same, i.e. allocative efficiency is satisfied and the percent deviations are zero. Therefore, we report only the average number of questions asked to the buyer in Table 6.1.

Table 6.1 Average number of comparisons*

	Underlying Preference Function		
	Weighted Linear	Weighted Euclidean	Weighted Tchebycheff
Step 1	1.9	2.1	2.2
Total	6.4	8.1	9.7

* Based on 10 instances with different weight values

As can be seen from Table 6.1, the average number of pairwise comparisons the buyer is required to make is 6.4, 8.1 and 9.7 for underlying weighted linear, weighted Euclidean and weighted Tchebycheff functions, respectively. We also report the number of questions asked to the buyer to find the most preferred supported alternative using the initial bids of the sellers in Step 1 of *CO-u*. We observe that our algorithm performs well, as it finds the optimal winning bids requiring a small number of questions.

We also find the optimal bids when the sellers bid independently of each other (Decentralized) in the discretized space and compare the results with Centralized. Centralized is guaranteed to have at least as good results as that of Decentralized, since in the former case we consider a centralized approach suitably matching the sellers to create the best combination. As expected, when the buyer has an underlying weighted linear preference function both Decentralized and Centralized are equivalent. In the nonlinear case, we test the performance of the algorithm by simulating the preferences of the buyer using weighted Euclidean and the weighted Tchebycheff preference functions. The percent deviations of Decentralized from Centralized are reported in Tables 6.2 and 6.3.

Table 6.2 Percentage deviations of decentralized from centralized optimal solutions under weighted Euclidean preference function

λ=0.05	λ=0.15	λ=0.25	λ=0.35	λ=0.45	λ=0.55	λ=0.65	λ=0.75	λ=0.85	λ=0.95
0.0000	0.0000	0.0000	0.0000	0.5978	1.7231	0.3873	0.0000	0.0000	0.0000

Table 6.3 Percentage deviations of decentralized from centralized optimal solutions under weighted Tchebycheff preference function

λ=0.05	λ=0.15	λ=0.25	λ=0.35	λ=0.45	λ=0.55	λ=0.65	λ=0.75	λ=0.85	λ=0.95
0.0000	0.0000	3.6321	4.2553	4.9567	2.5316	5.3631	1.8655	0.0000	0.0000

As can be seen from the tables, Centralized finds better solutions in many cases. In the Coordinated Bidding case where the sellers disclose their cost functions to ADSS, the buyer will benefit as his/her preference function value for the resulting bid combination is at least as good as that of Decentralized. Sellers will also benefit in the sense that the more competitive sellers will be matched through coordination. Therefore, both parties (buyer and the sellers) will benefit more when the sellers share their cost functions with ADSS.

CHAPTER 7

EXTENSIONS TO PREVIOUS WORK

In this chapter we discuss modifications we made to improve the Evolutionary Algorithm (EA) developed in Karakaya (2009) for MAMI auctions in order to overcome the computational difficulties.

The application of EAs in multi-objective optimization is beneficial as the EAs maintain a population of solutions in a single run and there are examples in the literature (see for example Deb et al. 2002, Zitzler et al. 2001, Soylu and Köksalan 2010, Karahan and Köksalan 2010).

Karakaya (2009) adapted the Non-Dominated Sorting Genetic Algorithm NSGA-II (Deb et al. 2002) to solve a MAMI reverse auction problem. The developed EA is used to approximate the Pareto front. Karakaya (2009) considers a MAMI auction environment with two attributes: price and defect rate. She assumes that all units of an item should be supplied by a single seller and each seller gives bids for each item. She considers two variations of the problem. The *base case* corresponds to a simpler version in terms of the prices, whereas the *discounted case* introduces price discounts in the bids that supply several items. She ran different versions of the algorithm considering different procedures to seed several initial solutions in the initial population and tested their performances against the true Pareto frontier for both *original* and *discounted* cases. She uses different problems with different combinations of the number of items and sellers:

(10,20), (30,30), and (30,100) where in the parentheses the former and latter values refer to the number of items and the number of sellers, respectively.

In the modified version, we made 10 replications for each instance by randomly generating problems. We try to synchronize the random numbers we use for the same purpose in each version of the algorithm to reduce the variation due to randomness. We consider an additional version, in which the initial population is seeded with all supported efficient solutions of the original case. We apply the procedure of Aneja and Nair (1979) to find all supported efficient solutions for the original case. As stated in Chapter 4, this procedure minimizes a weighted linear objective that combines the two objectives. Systematically changing the weights at each iteration, it guarantees obtaining all supported efficient solutions. Once we get the weights from this procedure at each iteration, we find the solution that minimizes the weighted objective function employing an efficient sorting procedure. We first calculate the value of each seller for each item by multiplying the attribute values of the item of the seller with the corresponding weights and summing them up. Then, for each item we choose the seller having the minimum value as the winner. We try this new version in both original and discounted cases. As expected this version of the algorithm performs well as the algorithm starts with good solutions. However, its performance is not good in the discounted case as the problem structures are different. Karakaya (2009) considers two performance measures to test the algorithm. We apply *paired-t test* to statistically compare different versions of the algorithm in terms of these two performance measures. The results of the experiments can be seen in Appendix B.

In the manuscript, although we demonstrated our algorithm for a single round, it is directly applicable in a multi-round setting as well. After the bidders update their bids based on the feedback mechanism of the auction, our algorithm can be employed to find the new approximate efficient front in the next round. We intend to incorporate our algorithm into such multi-round settings as future research.

For single as well as multiple round auctions it would be useful to develop preference-based EAs that explore only the parts of the Pareto front that are of interest to the decision maker.

CHAPTER 8

CONCLUSIONS

In this study, we address multi-attribute multi-item auction problems. We develop auction decision support systems, ADSSs, that provide aid to the buyer in single-round auctions whereas it provides aid to both parties in multi-round auctions. We first develop an approach that finds a set of efficient bid combinations to present to the buyer. The buyer determines the preferred and nonpreferred combinations. Based on the preferences of the buyer, ADSS estimates the parameter values of the underlying preference function of the buyer as well as the estimated preference function value for each item separately, and inform sellers about these estimations. In the succeeding rounds sellers update their bids and the auction continues until a termination condition is met. We generate a number of test problems and test our algorithm for both two and three attribute problems for an underlying linear preference function. Our algorithm finds the same winning sellers that are found using exact parameter values, i.e. allocative efficiency is reached. Also the buyer's preference function is closely approximated. We also use the algorithm as a heuristic for nonlinear preference functions. The results also indicate that our algorithm performs well.

We then develop an interactive algorithm, *LIN-u*, to support the buyer to find the most preferred bid combination for underlying linear preference functions. The results show that with *LIN-u* we guide the sellers well and both the buyer and the sellers can benefit. We also modify the algorithm and develop a heuristic method

for underlying quasiconvex preference functions. The experiments show that our algorithm performs well.

For underlying quasiconvex preference functions we develop two algorithms *QCX-u* and L_α*-u*, with different version of each algorithm. Based on our experiments, we conclude that our guidance mechanism works well for underlying quasiconvex preference functions.

In the algorithms above, we assume that we do not know the cost functions of the sellers. We also develop an interactive approach assuming that all sellers disclose their cost functions explicitly to us. We refer to this case as "*Coordinated Bidding*" and develop the interactive *CO-u* algorithm. Our algorithm finds the optimal winning sellers requiring only a small number of preference comparisons from the buyer.

In Karakaya (2009), an Evolutionary Algorithm (EA) was developed for the case of multi-attribute, multi-item reverse auctions in order to overcome the computational difficulties. We made some modifications and improved the algorithm. We approximately generate the whole Pareto front using the EA. We also develop heuristic procedures to find several good initial solutions and insert those in the initial population of the EA. We test the EA on a number of randomly generated problems and report our findings.

When the number of possible bid combinations is too high to find the efficient bid combinations within a reasonable computational effort, heuristics such as EAs can be utilized. Developing a preference-based EA that finds some parts of the Pareto front based on the preferences of the buyer could be beneficial as this would avoid generating the whole Pareto front. As a future study, the interactive approaches we developed in this thesis can be utilized in the development of a preference-based EA.

When we do not know the cost functions of the sellers, we estimate the preference function value of each item in a combination of bids to inform the sellers. We intend to work on procedures to assign meaningful contribution values to the components of a bid combination.

In Chapter 5, we develop algorithms for underlying quasiconvex preference functions. Understanding the form of the underlying preference function is important to utilize these algorithms more beneficially. We aim to work on different methods to identify the form of the underlying preference function; specifically we intend to apply the procedure in Köksalan and Sagala (1995) as a future study.

The experiments show that our guidance mechanism works well and both the sellers and the buyer can benefit from using ADSSs. The implementation of this decision support system in a web-based platform and implementing it in practice are important future challenges.

REFERENCES

Aneja, Y.P., and Nair, K.P.K., 1979, Bicriteria Transportation Problem, *Management Science*, 25(1), 73–78.

Bapna, R., Wolfgang, J. and Shmueli, G., 2008, Price formation and its dynamics in online auctions, *Decision Support Systems*, 44 (3), 641-656.

Baykal, Ş., Combinatorial Auction Problems, Master's Thesis, Department of Industrial Engineering, Middle East Technical University, 2007.

Bellosta, M.J., Brigui, I., Kornman, S. and Vanderpooten, D., 2004, A Multi-criteria Model for Electronic Auctions, *In Proceedings of ACM Symposium on Applied Computing*, 759-765, March 14-17, Nicosia, Cyprus.

Bichler, M. and Kalagnanam, J., 2005, Configurable offers and winner determination in multi-attribute auctions, *European Journal of Operational Research*, 160(2), 380-394.

Bozkurt, B., Fowler, J.W., Gel, E.S., Kim, B., Köksalan, M. and Wallenius, J., 2010, Quantitative comparison of approximate solution sets for multicriteria optimization problems with weighted Tchebycheff preference function, *Operations Research*, 58(3), 650-659.

Buer, T., and Pankratz, G., 2010, Solving a bi-objective winner determination problem in a transportation procurement auction, *Logistics Research*, 2, 65–78.

Catalán, J., Epstein, R., Guajardo, M., Yung, D. and Martínez, C., 2009, Solving multiple scenarios in a combinatorial auction, *Computers & Operations Research*, 36(10), 2752–2758.

Chinneck, J.W., 2008, *Feasibility and infeasibility in optimization: algorithms and computational methods*, New York: Springer.

De Vries, S. and Vohra, R., 2003, Combinatorial Auctions: A Survey, *INFORMS Journal on Computing*, 15(3), 284–309.

Deb, K., Pratap, A., Agrawal, S. and Meyarivan, T., 2002, A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation*, 6(2), 182–197.

Ervasti, V. and Leskelä, R.L., 2010, Allocative efficiency in simulated multiple-unit combinatorial auctions with quantity support, *European Journal of Operational Research*, 203(1), 251–260.

Fujishima, Y., Leyton-Brown, K. and Shoham, Y., 1999, Taming the Computational Complexity of Combinatorial Auctions: Optimal and Approximate Approaches, *International Joint Conference on Artificial Intelligence (IJCAI)*, 548–553.

Haimes, Y., Lasdon, L. and Wismer, D., 1971, On a Bicriterion Formulation of the Problems of Integrated System Identification and System Optimization, *IEEE Transactions on Systems, Man, and Cybernetics 1*, 296–297.

Hohner, G., Rich, J., Ng, E., Reid, G., Davenport, A.J., Kalagnanam, J.R., Lee, H.S. and An, C., 2003, Combinatorial and quantity–discount procurement auctions benefit, Mars, Incorporated and its suppliers, *Interfaces*, 33(1), 23–35.

Karahan, İ. and Köksalan, M., 2010, A Territory Defining Multiobjective Evolutionary Algorithms and Preference Incorporation, *IEEE Transactions on Evolutionary Computation*, 14 (4), 636–664.

Karakaya, G. and Köksalan, M., 2011, An interactive approach for multi-attribute auctions, *Decision Support Systems*, 51(2), 299–306.

Karakaya, G., 2009, Approaches for multi-attribute auctions, Master's Thesis, Department of Industrial Engineering, Middle East Technical University.

Keeney, R.L. and Raiffa, H., 1993, Decision Making with Multiple Objectives: Preferences and Value Tradeoffs, Cambridge University Press, Cambridge.

Koppius, O.R., Van Heck, E. and Wolters, M.J.J., 2004, The importance of product representation online: empirical results and implications for electronic markets, *Decision Support Systems*, 38 (2), 161-169.

Korhonen P., Wallenius J. and Zionts, S., 1984, Solving the discrete multiple criteria problem using convex cones, *Management Science*, 30(1), 1336–1345.

Korhonen, P.J. and Laakso, J., 1986, A visual interactive method for solving the multiple criteria problem, *European Journal of Operational Research*, 24(2), pp. 277-287.

Köksalan, M. and Sagala, P.N.S., 1995, An approach to and computational results on testing the form of a decision maker's utility function, *Journal of Multi-criteria Decision Analysis*, 4, 189-202.

Köksalan, M., 1999, A heuristic approach to bicriteria scheduling, *Naval Research Logistics*, 46(7), 777–789.

Köksalan, M., Leskelä, R.L., Wallenius, H. and Wallenius, J., 2009, Improving efficiency in multiple-unit combinatorial auctions: bundling bids from multiple bidders, *Decision Support Systems,* 48(1), 103–111.

Lehmann, D., Mueller, R. and Sandholm, T., 2006, The Winner Determination Problem, In: Cramton, P., Shoham, Y., Steinberg, R. (Eds.), *Combinatorial Auctions*, Chapter 12. MIT Press.

Leskelä, R.L., Teich, J.E., Wallenius, H. and Wallenius, J., 2007, Decision support for multi-unit combinatorial bundle auctions, *Decision Support Systems,* 43(2), 420–434.

Leyton-Brown, K., Pearson, M. and Shoam, Y., 2000, Towards a universal test suite for combinatorial auction algorithms, *ACM Conference on Electronic Commerce*, 66–76.

Lokman, B. and Köksalan, M., 2012, Finding All Nondominated Points of Multi-objective Integer Programs, *Journal of Global Optimization*, 57, 347–365.

McAfee, R. and McMillan, J., 1987, Auctions and Bidding, *Journal of Economic Literature*, 25(2), 699–738.

Metty, T., Harlan, R., Samelson, Q., Moore, T., Morris, T., Sorensen, R., Schneur, A., Raskina, O., Schneur, R., Kanner, J., Potts, K. and Robbins J., 2005, Reinventing the Supplier Negotiation Process at Motorola, *Interfaces,* 35(1), 7–23.

Ramesh, R., Karwan, M.H. and Zionts, S., 1990, An interactive method for bicriteria integer programming, *IEEE Transactions on Systems, Man, and Cybernetics*, 20(2), 395–403.

Rothkopf, M.H. and Park, S., 2001, An Elementary Introduction to Auctions, *Interfaces,* 31(6), 83–97.

Sandholm, T. and Suri, S., 2003, BOB: Improved winner determination in combinatorial auctions and generalizations, *Artificial Intelligence,* 145(1-2), 33–58.

Sandholm, T. and Suri, S., 2006, Side constraints and non-price attributes in markets, *Games and Economic Behavior*, 55, 321–330.

Sandholm, T., 2002, Algorithm for optimal winner determination in combinatorial auctions, *Artificial Intelligence,* 135(1-2), 1–54.

Sandholm, T., Levine, D., Concordia, M., Martyn, P., Hughes, R., Jacobs, J., & Begg, D., 2006, Changing the Game in Strategic Sourcing at Procter & Gamble: Expressive Competition Enabled by Optimization, *Interfaces*, 36(1), 55–68.

Sandholm, T., Suri, S., Gilpin, A. and Levine, D., 2002, Winner determination in combinatorial auction generalizations, *In Proceedings of the first International Conference on Autonomous and Multi-agent Systems*, 69-76, Bologna, Italy.

Sandholm, T., Suri, S., Gilpin, A. and Levine, D., 2005, CABOB: A Fast Optimal Algorithm for Winner Determination in Combinatorial Auctions, *Management Science,* 51(3), 374–390.

Sheffi, Y., 2004, Combinatorial Auctions in the Procurement of Transportation Services, *Interfaces,* 34(4), 245–252.

Soylu, B. and Köksalan, M., 2010, A Favorable Weight Based Evolutionary Algorithm for Multiple Criteria Problems, *IEEE Transactions on Evolutionary Computation*, 14 (2), 191–205.

Talluri, S., Narasimhan, R. and Viswanathan, S., 2007, Information technologies for procurement decisions: a decision support system for multi-attribute e-reverse auctions, *International Journal of Production Research*, 45(11), 2615–2628.

Teich, J.E., Wallenius, H., Wallenius, J. and Zaitsev, A., 2006, A multi-attribute e-auction mechanism for procurement: theoretical foundations, *Journal of Operational Research*, 175(1), 90–100.

Teich, J.E., Wallenius, H., Wallenius, J. and Koppius, O.R., 2004, Emerging multiple issue e-auctions, *European Journal of Operational Research*, 159, 1–16.

Van Veldhuizen, D. A. and Lamont, G. B., 2000, On measuring multiobjective evolutionary algorithm performance, *Proceedings of the 2000 Congress on Evolutionary Computation, IEEE*, 1, 204–211.

Zionts, S., 1981, A multiple criteria method for choosing among discrete alternatives, *European Journal of Operational Research*, 7(2), 143–147.

Zitzler, E. and Thiele, L., 1998, Multiobjective Optimization Using Evolutionary Algorithms - A Comparative Case Study, *Parallel Problem Solving from Nature, PPSN V*, 292–301.

Zitzler, E., Laumanns, M., and Thiele, L., 2001, SPEA2: Improving the Strength Pareto Evolutionary Algorithm, TIK-Report, 103, Swiss Federal Institute of Technology, Switzerland.

APPENDIX A

PARAMETER SETTING IN EFFCOM MODEL

In our experiments, we round the attribute values to four significant digits. Thus, the minimum increments of the objectives (attributes of a bid combination) are 10^{-4}. For the two-attribute case, by setting a suitable ρ value in the objective function and systematically changing ε_j value, we generate all efficient solutions.

We first solve the EFFCOM model to minimize only the second objective. The objective function value of the optimal solution is the is the smallest possible value for ε_j. To obtain the largest possible value for ε_j, we solve the EFFCOM model in a lexicographic manner. We minimize the first objective. By fixing attribute 1 value of the corresponding solution, we then solve the model again to minimize the second objective. Attribute 2 value of the optimal solution is the largest possible value for ε_j.

Let *mn* and *mx* be the smallest and largest possible values for ε_j. Since the minimum increment between the attribute 1 values is 10^{-4}, we set ρ such that $\rho.(mx - mn) < 10^{-4}$. With this we guarantee that the maximum increment in the second objective would not cause any trade-offs between the first and second objectives. The augment part only serves for breaking ties.

After setting ρ value, we solve EFFCOM model repeatedly by changing the ε_j value.We start with the largest possible value (*mx*) for ε_j and systematically reduce it until its smallest possible value (*mn*). We reduce the second attribute

value obtained in the most recent solution of EFFCOM by 10^{-4}, which is the minimum increment between the attribute 2 values as we round the attribute values to four significant digits. This guarantees generating all efficient solutions.

APPENDIX B

RESULTS OF THE EVOLUTIONARY ALGORITHM

Karakaya (2009) developed an Evolutionary Algorithm (EA) with different variations. In the original case, she considers two versions of the algorithm: *without seeding*, *seeding by sorting*. In the *without seeding* case, all members of the initial population are generated randomly. In the *seeding by sorting* case, the initial population is seeded with two solutions corresponding to the best solutions for each objective for the original case. In the discounted case, she considers three versions of the algorithm: *without seeding*, *optimal seeding* and *rank heuristic.* Without seeding is as in the original case. Optimal seeding is similar to seeding by sorting but in the discounted case the initial population is seeded with the best solutions for each objective for the discounted case. In the last version she applies *rank heuristic* to find an approximate nondominated solution considering the price objective in the discounted case and uses a simple sorting procedure to find a good solution in terms of the defect rate objective. She then uses these two combinations to seed the initial population. As stated in Chapter 7, in the modified version we consider an additional version, *supported seeding*, in which the initial population is seeded with all supported efficient solutions of the original case. We refer to the version that the initial population is seeded with two solutions corresponding to the best solutions for each objective in the corresponding case as *seeding two extremes* in both cases.

Karakaya (2009) compares the results of the different version of the algorithm with the true Pareto optimal front obtained by solving a series of integer programs. She considers two performance measures to test the algorithm;

Hypervolume Indicator (Zitzler and Thiele, 1998) which measures the dominated hypervolume to a given reference point and the Inverted Generational Distance Metric (Van Veldhuizen and Lamont, 2000) which measures the Euclidean distance of each solution in the true Pareto front to the closest solution in the population generated by the algorithm. The average of these distances are used and for the Inverted Generational Distance Metric (IGDM) smaller values are desirable.

Let HI_v^* to represent the ratio of the hypervolumes of the vth version of the algorithm to that of the true Pareto front (i.e., HI_v^*= hypervolume of the vth version of the algorithm/hypervolume of the true Pareto front) where v corresponds to seeding two extremes, without seeding, supported seeding, and rank heuristic. When an algorithm generates the true Pareto front exactly, HI_v^* takes its best possible value of 1, and it takes smaller values as the algorithm's performance deteriorates, with a minimum possible value of 0.

We conduct experiments and the preliminary results show that seeding two extremes performs well in both cases. We apply *paired-t test* to statistically compare this version with the other versions. We compute the sample means $\overline{HI_v^*}, \overline{IGDM_v}$ and the sample deviations $SD(HI_v^*), SD(IGDM_v)$ for both metrics. At a 99% significance level we test the following hypothesis:

$H_0 : \mu_1^m = \mu_v^m$
$H_1 : \mu_1^m \neq \mu_v^m$
v= without seeding, supported seeding for the original case
v= without seeding, supported seeding, rank heuristic for the discounted case

where *v* is the version of the algorithm as before and *m* stands for the performance metric (HI^* and IGDM). The null hypothesis states that there is no statistical difference between seeding two extremes and its contender. When we fail to

reject the null hypothesis at 99% significance level, we indicate the winner as "none" in the corresponding tables. On the other hand, if the statistical test indicates a significant difference, we report the winner in the table based on the corresponding 99% confidence interval.

For each problem set, we find the difference $\left(\Delta_v^{HI^*} = \overline{HI}_1^* - \overline{HI}_v^* \text{ and } \Delta_v^{IGDM} = \overline{IGDM}_1 - \overline{IGDM}_v\right)$ between the average performance metrics of the seeding two extremes and its contenders. The performance of the algorithm for each version in terms of HI*, IGDM, CPU time, and statistical test results for each problem set are reported in the following tables. We also give the CPU time of the ε-constraint method used to generate the true Pareto front as a benchmark. We report the results for the original and discounted cases in Tables B.1-B.6 and in Tables B.7-B.12, respectively.

Table B.1 Results for Original Case Problem Set (10,20) (300 Generations)*

Version (v)	$\overline{HI}_v^*$	$SD(HI_v^*)$	$\overline{IGDM}_v$	$SD(IGDM_v)$	Average CPU time (sec)
seeding two extremes	0.9990	0.0010	0.00164	0.00120	0.8244
without seeding	0.7087	0.1793	0.02806	0.02656	0.8361
supported seeding	0.9995	0.0007	0.00112	0.00089	0.8222
true Pareto	-				5.3415

*Based on 10 instances

Table B.2 Statistical Comparison of Seeding two Extremes with its Contenders for Original Case Problem Set (10,20)

Contender	Δ_v^{HI*}	p-value	Winner	Δ_v^{IGDM}	p-value	Winner
without seeding	0.2903	0.001	seeding two extremes	-0.02643	0.013	None
supported seeding	-0.0005	0.027	none	0.00052	0.111	None

Table B.3 Performance Measures for Original Case Problem Set (30,30) (2000 Generations)*

Version (v)	$\overline{HI}_v^*$	$SD(HI_v^*)$	$\overline{IGDM}_v$	$SD(IGDM_v)$	Average CPU time (sec)
seeding two extremes	0.9925	0.0013	0.00723	0.00081	8.1884
without seeding	0.6922	0.0694	0.01901	0.00462	8.3148
supported seeding	0.9927	0.0009	0.00708	0.00046	8.1901
true Pareto	-				39.5942

*Based on 10 instances

Table B.4 Statistical Comparison of Seeding two Extremes with its Contenders for Original Case Problem Set (30,30)

Contender	Δ_v^{HI*}	p-value	Winner	Δ_v^{IGDM}	p-value	Winner
without seeding	0.3003	0.000	seeding two extremes	-0.01179	0.000	seeding two extremes
supported seeding	-0.0002	0.496	none	0.00015	0.463	none

Table B.5 Performance Measures for Original Case Problem Set (30,100) (4000 Generations)*

Version (v)	$\overline{HI}_v^*$	$SD(HI_v^*)$	$\overline{IGDM}_v$	$SD(IGDM_v)$	Average CPU time (sec)
seeding two extremes	0.9908	0.0019	0.00999	0.00137	19.4171
without seeding	0.5325	0.0620	0.04268	0.01546	19.3516
supported seeding	0.9929	0.0015	0.00845	0.00107	20.3645
true Pareto	-				87.7548

*Based on 10 instances

Table B.6 Statistical Comparison of Seeding two Extremes with its Contenders for Original Case Problem Set (30,100)

Contender	Δ_v^{HI*}	p-value	Winner	Δ_v^{IGDM}	p-value	Winner
without seeding	0.4582	0.000	seeding two extremes	-0.03268	0.000	seeding two extremes
supported seeding	-0.0021	0.004	supported seeding	0.00155	0.004	supported seeding

The results show that for the original case both seeding two extremes and supported seeding performs well. It is an expected result for the supported seeding version to work well as many solutions of the true Pareto front is seeded in the initial population. In Table B.6, we observe that supported seeding outperforms seeding two extremes however the results of both algorithms are very close to each other in both HI* and IGDM values. Seeding two extremes requires only two extreme solutions to seed in the initial population whereas supported seeding requires all supported efficient solutions. Although finding all supported efficient solutions was easy for the original case, it may prove difficult in general and one may need to find approximations of these solutions to seed in the initial population as will be the case in our discounted problem.

Table B.7 Performance Measures for Discounted Case Problem Set (10,20) (300 Generations)*

Version (v)	$\overline{HI}_v^*$	$SD(HI_v^*)$	$\overline{IGDM}_v$	$SD(IGDM_v)$	Average CPU time (sec)
seeding two extremes	0.9952	0.0061	0.00495	0.00405	1.0221
without seeding	0.7140	0.1465	0.02770	0.01578	0.8378
rank heuristic	0.7415	0.1696	0.02611	0.01526	0.8377
supported seeding	0.7654	0.1475	0.02198	0.01363	0.8255
true Pareto	-				10.4410

*Based on 10 instances

Table B.8 Statistical Comparison of Seeding two Extremes with its Contenders for Discounted Case Problem Set (10,20)

Contender	Δ_v^{HI*}	p-value	Winner	Δ_v^{IGDM}	p-value	Winner
without seeding	0.2812	0.000	seeding two extremes	-0.02274	0.001	seeding two extremes
rank heuristic	0.2537	0.001	seeding two extremes	-0.02116	0.002	seeding two extremes
supported seeding	0.2298	0.001	seeding two extremes	-0.01703	0.002	seeding two extremes

Table B.9 Performance Measures for Discounted Case Problem Set (30,30) (4000 Generations)*

Version (v)	$\overline{HI}_v^*$	$SD(HI_v^*)$	$\overline{IGDM}_v$	$SD(IGDM_v)$	Average CPU time (sec)
seeding two extremes	0.9811	0.0081	0.01335	0.00485	17.1414
without seeding	0.7259	0.1075	0.04016	0.01808	16.5237
rank heuristic	0.7332	0.1238	0.03197	0.01493	16.5298
supported seeding	0.6965	0.1465	0.04031	0.01266	16.5874
true Pareto	-				332.8922

*Based on 10 instances

Table B.10 Statistical Comparison of Seeding two Extremes with its Contenders for Discounted Case Problem Set (30,30)

Contender	Δ_v^{HI*}	p-value	Winner	Δ_v^{IGDM}	p-value	Winner
without seeding	0.2551	0.000	seeding two extremes	-0.02680	0.000	seeding two extremes
rank heuristic	0.2478	0.000	seeding two extremes	-0.01862	0.003	seeding two extremes
supported seeding	0.2846	0.000	seeding two extremes	-0.02696	0.000	seeding two extremes

Table B.11 Performance Measures for Discounted Case Problem Set (30,100) (7000 Generations)*

Version (v)	$\overline{HI}_v^*$	$SD(HI_v^*)$	$\overline{IGDM}_v$	$SD(IGDM_v)$	Average CPU time (sec)
seeding two extremes	0.9709	0.0124	0.02244	0.01033	36.5301
without seeding	0.6478	0.1298	0.07746	0.01918	33.6851
rank heuristic	0.6656	0.0684	0.06801	0.0209	34.0037
supported seeding	0.6413	0.0670	0.07878	0.01502	34.3830
true Pareto	-				1241.0686

*Based on 10 instances

Table B.12 Statistical Comparison of Seeding two Extremes with its Contenders for Discounted Case Problem Set (30,100)

Contender	Δ_v^{HI*}	p-value	Winner	Δ_v^{IGDM}	p-value	Winner
without seeding	0.3231	0.000	seeding two extremes	-0.05502	0.000	seeding two extremes
rank heuristic	0.3053	0.000	seeding two extremes	-0.04557	0.000	seeding two extremes
supported seeding	0.3296	0.000	seeding two extremes	-0.05635	0.000	seeding two extremes

The results show that in all test problems seeding two extremes is significantly better than the other version for the discounted case. Seeding two extremes represents the true Pareto front well in a fraction of the time required to generate the true Pareto front. Moreover, for the largest problem set (30,100), seeding two extremes has HI* value of 97% and a small IGDM value. Although with rank heuristic and supported seeding the best solution for the defect objective can be found, the best solution for the price objective cannot be found.

We expect that as the number of good solutions seeded in the initial population increases, the performance of the algorithm further improves. The main tradeoff is the computational time to find the good solutions. Therefore, it may be worthwhile to develop fast heuristics that give good seed solutions.

CIRRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Karakaya, Gülşah
Nationality: Turkish (TC)
Date and Place of Birth: 5 November 1983, Akşehir
Marital Status: Married
Phone: +90 312 210 29 56
email: kgulsah@metu.edu.tr

EDUCATION

Degree	Institution	Year of Graduation
MS	METU Industrial Engineering	2009
BS	METU Industrial Engineering	2007
High School	Konya Meram Science High School	2002

WORK EXPERIENCE

Year	Place	Enrollment
2007-Present	METU Dept. of Industrial Eng.	Research Assistant
2006-2007	METU Dept. of Industrial Eng.	Undergraduate Assistant
2006 July	Bosch	Intern Engineering Student
2005 July	Vestel	Intern Engineering Student

PUBLICATIONS

Karakaya, G. and M. Köksalan, "An Interactive Approach for Bi-attribute Multi-Item Auctions," *under review* in *Annals of Operations Research.*

Köksalan, M. and G. Karakaya, "An Evolutionary Algorithm for Finding Efficient Solutions in Multi-Attribute Auctions," *under review* in *International Journal of Information Technology & Decision Making*.

Karakaya, G. and M. Köksalan, "An Interactive Approach for Multi-attribute Auctions," *Decision Support Systems*, 51, 299-306, (2011).

CONFERENCE PRESENTATIONS

Karakaya, G. and M. Köksalan, "Decision Support for Multi-attribute Multi-item Reverse Auctions," 22nd International Conference on MCDM, Málaga, Spain, June 2013.

Karakaya, G. and M. Köksalan, "An Approach for Multi-attribute Multi-item Reverse Auctions," 1st International IIE Conference, İstanbul, Turkey, June 2013.

Karakaya, G. and M. Köksalan, "Decision Support for Multi-attribute Multi-item Reverse Auctions," 21st International Symposium on Mathematical Programming (ISMP), Berlin, Germany, August 2012.

Karakaya, G. and M. Köksalan, "An Interactive Approach for Multi-attribute Multi-item Reverse Auctions," 21st International Conference on MCDM, Jyväskylä, Finland, June 2011.

Karakaya, G. and M. Köksalan, "An Approach for Multi-attribute Reverse Auctions," 20th International Conference on MCDM, Chengdu, China, June 2009.

SEMINARS

Karakaya, G., "Decision Support for Multi-attribute Reverse Auctions," Department of Industrial Engineering, Middle East Technical University, Ankara, Turkey, April 2012.

Karakaya, G., "Approaches for Multi-attribute Reverse Auctions," Department of Industrial Engineering, Middle East Technical University, Ankara, Turkey, May 2010.

Karakaya, G., "Approaches for Multi-attribute Auctions," Aalto University School of Science and Technology, Helsinki, Finland, October 2010.

AWARDS

METU Graduate Courses Performance Award 2011

Ph.D. Scholarship, The Scientific and Technological Research Council of Turkey (TÜBİTAK) 2009 - 2013

M.S. Scholarship, The Scientific and Technological Research Council of Turkey (TÜBİTAK) 2007 - 2009

Bosch Engineering Scholarship 2005 - 2007

FOREIGN LANGUAGES

Advanced English, Intermediate German