INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.
Multiobjective decision-making: An interactive integrated optimization approach

Al-Alwani, Jumah Eid, Ph.D.
Case Western Reserve University, 1991
MULTIOBJECTIVE DECISION MAKING: AN INTERACTIVE INTEGRATED OPTIMIZATION APPROACH

by

JUMAH EID AL-ALWANI

Submitted in partial fulfillment of the requirements for the Degree of Doctor of Philosophy

Thesis Advisors: Dr. Benjamin F. Hobbs and Dr. Behnam Malakooti

Department of Systems Engineering
CASE WESTERN RESERVE UNIVERSITY
January, 1991
We hereby approve the thesis of

Jumah Eid Al-Alwani

candidate for the Ph.D.

degree.*

Signed: [Signatures]

(Chairman)

Date December 5, 1990

*We also certify that written approval has been obtained for any proprietary material contained therein.
MULTIOBJECTIVE DECISION MAKING: AN INTERACTIVE INTEGRATED OPTIMIZATION APPROACH

Abstract

by

JUMAH EID AL-ALWANI

In this thesis, a new interactive integrated approach for solving multiobjective decision making (optimization) problems is presented. The approach is very flexible and general in that it handles two broad classes of implicit utility functions: quasiconcave and quasiconvex. The first step is to use preference comparison-based tests to determine the class of utility function that is consistent with the decision maker's underlying preferences with respect to a sample of nondominated alternatives. Then an algorithm appropriate for the selected function is used. For the case of quasiconcave utility, an interactive modified Geoffrion-Dyer-Feinberg algorithm is developed by projecting the gradient-based improvement direction on the nondominated frontier as well as providing an interactive termination criterion. However, for the case of quasiconvex utility, the structure of model (i.e. linear vs nonlinear) plays a vital role in selecting the algorithm. In the MOLP case, an interactive branch and bound-based algorithm is presented. This algorithm takes advantage of the fact that the most-preferred solution occurs at an efficient
extreme point. In the MONLP case, an interactive gradient-based heuristic algorithm is outlined. This algorithm also takes advantage of some derived properties for the most-preferred solution. The demands upon the DM are kept to a minimum and only preference comparisons information are elicited in all of the above algorithms. Example applications are presented for each algorithm.
This dissertation is dedicated to the following,

my brother: Mohammed Eid Al-alwani,

my wife: Samyra Mofeed Osta,

and

my two daughters: Deemh and Danh Jumah Al-alwani
Acknowledgement

I would like to thank both of my thesis advisers Dr. Benjamin F. Hobbs and Dr. Behnam Malakooti for their guidance, encouragement and suggestions through this research. I am also in debt to Dr. Vira Chankong for his valuable suggestions. I truly appreciate Dr. Michael J. Ginzberg for serving as a committee member.

My graduate study is supported by a scholarship from King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia through the Saudi Arabian Cultural Mission to the USA, Washington, D.C. I would like to express my thanks and appreciation to them.

I would like also to Acknowledge the encouragement and support of my brother Mohammad Eid Al-alwani through all of my school years. His invaluable support made a difference.

At last, but not the least, I would like to offer my deepest gratitude to my wife Samyra Mofeed Osta and my two daughters Deemh and Danh for their patience, love and understanding through the years of my graduate school.
# Table of Contents

Chapter 1: Introduction  
1.1 Purpose ..............................................................................................................1  
1.2 The Research Problem .....................................................................................2  
1.2.1 Motivation .....................................................................................................2  
1.2.2 The Integrated Multiobjective Approach .....................................................6  
1.2.3 Methodological Contribution ....................................................................12  
1.3 Scope ..................................................................................................................14

Chapter 2: Multiobjective Optimization: A Literature Review .............15  
2.1 Introduction .........................................................................................................15  
2.2 The MCDM Problem: Definition and Terminology .................................16  
2.3 Multiobjective Optimization ..........................................................................19  
2.3.1 Single Solution-based Approaches ............................................................21  
2.3.2 Representative Generation-based Approaches .........................................23  
2.3.3 Progressive Generation-based Approaches ...............................................26  
2.3.3.1 Value Judgment Information ..................................................................28  
2.3.3.2 Classes of Interactive Methods .................................................................30

Chapter 3: The Integrated Multiobjective Approach .............................36  
3.1 Introduction .........................................................................................................36  
3.2 Properties of Nondominated Alternatives ......................................................37  
3.2.1 Properness ...................................................................................................38  
3.2.2 Supportedness .............................................................................................38  
3.3 The Tchebycheff-based Sampling Scheme ....................................................39  
3.4 Classes of Utility functions ..............................................................................45  
3.4.1 Analysis of Standard Utility/value Models ...............................................46  
3.4.2 Utility Class Testing .....................................................................................55  
3.5 The Integrated Multiobjective Optimization Approach ..........................63  
3.5.1 Assumptions .................................................................................................63
Chapter 4: The Quasiconcave Utility-based Multiobjective Approach

4.1 Introduction ..........................................................................................................86
4.2 Frank-wolf’s Algorithm And its GDF Version ...................................................87
4.3 The Interactive Gradient Projection-based Algorithm ........................................91
   4.3.1 Steps of The algorithm ................................................................................91
4.4 Convergence .........................................................................................................93
4.5 Implementation Issues .......................................................................................94
   4.5.1 Gradient Assessment ...................................................................................96
   4.5.1.1 MRS-based Assessment .........................................................................96
   4.5.1.2 Paired Comparison-based Assessment ..................................................100
4.5.2 Projection .......................................................................................................103
   4.5.2.1 Tchebycheff-based Projection ................................................................105
   4.5.2.2 Deviation-based Projection .....................................................................106
4.5.3 One-dimensional Curve Search .....................................................................110
4.5.4 The Interactive Termination Criterion .........................................................110
4.6 Examples .............................................................................................................126
   4.6.1 Example 4.1 ................................................................................................126
   4.6.2 Example 4.2 ................................................................................................136
4.7 Computational Experiments ..............................................................................137
4.8 Conclusions .......................................................................................................143

Chapter 5: The Quasiconvex Utility-based Multiobjective Approach

5.1 Introduction .......................................................................................................144
5.2 Basic Definition and Theories ............................................................................146
5.3 Multiobjective Linear Programming ...................................................................148
   5.3.1 Basic Theorems ..........................................................................................148
   5.3.2 The interactive Branch and Bound Algorithm ............................................151
      5.3.2.1 Overview ............................................................................................151
5.3.2.2 Steps of the algorithm and its convergence ........154
5.3.2.3 Implementation Issues .....................................156
5.3.2.4 Example 5.1 ....................................................159
5.3.2.5 Computational Experiments ..............................168
5.4 Multiobjective Nonlinear Programming ..................172
  5.4.1 Basic Theorems ..................................................173
  5.4.2 An Interactive Heuristic Algorithm ....................175
    5.4.2.1 Overview ...................................................175
    5.4.2.2 Steps of the algorithm ................................176
    5.4.2.3 Implementation Issues .................................177
    5.4.2.4 Examples ..................................................178
5.5 Conclusions .......................................................180

Chapter 6: Conclusions ..............................................182

Chapter 7: References ..............................................185

Appendix I: Proof of Theorem 4.6 .............................192

Appendix II: The GENERATE_EVALUE Routine ..............194

Appendix III: Test Problems for the Quasiconcave Utility-based
  Multiobjective Algorithm ...........................................199

Appendix IV: The QXMOLP Program for the Quasiconvex
  Utility-based MOLP Algorithm ...................................205

Appendix V: Test Problems for the Quasiconvex
  Utility-based MOLP Algorithm ...................................222
1. INTRODUCTION

1.1 Purpose

In this research, we develop a general interactive integrated multiobjective optimization methodology. The developed approach is designed to handle the general form of multiobjective mathematical programming problems with continuous decision variables. The approach exhibits the new feature of accommodating different preference structures that decision makers might have. The approach is not only equipped with an algorithm that handles quasiconcave utility functions, but also algorithms for handling the case of the quasiconvex ones.

The methodology starts by identifying the class of utility functions that decision maker might behave according to with respect to a representative set of competitive outcomes. After deciding on the class of utility function, the approach selects the appropriate multiobjective method from a set of developed algorithms. Each algorithm guides the decision maker to search for his most-preferred alternative and thereby the best decision through a systematic exploration of the space of candidate alternatives for the problem under consideration.

The introduction of the human component as an essential part during the solution process is facilitated by the developed methodology through an appropriate question-answering scheme in order to have a comfortable decision-making session.
1.2 The Research Problem

1.2.1 Motivation

Multiple Criteria Decision Making (MCDM) has been the emphasis of several research subjects for the last ten to fifteen years, where various approaches were developed to enable the decision maker to proceed through the decision-making process with ease and comfort (e.g., Chankong and Haines 1983a; Chankong et al. 1984; Zions 1982). While some of these methods handle a finite set of explicit alternatives (Malakooti 1988a), others tend to handle a set of infinite number of alternatives represented in a mathematical programming form such as MOLP or MONLP (e.g., Chankong and Haines 1978).

Due to the necessity of human involvement in any preference-driven approach, the decision maker’s underlying preference structure (assumed to be monotonic) is usually represented as an implicit real-valued monotonic function. While some approaches did not restrict the utility function shape to any form (e.g., Chankong and Haines 1978; Steuer and Choo 1983), other researchers, motivated by the economic theory through the principle of diminishing marginal rate of utility, restricted it to the class of generalized concave functions such as concave (Gofferson, Dyer and Feinberg 1972), pseudoconcave (Korhonen and Laakso 1986) and quasiconcave (Malakooti 1988b). Another motivation for using this class is that the problem of maximizing a generalized concave
function is relatively well developed in the literature (Bazaraa and Shetty 1979).

Although, the concept of restricting the utility function to a certain class is theoretically appealing in the sense that useful theoretical results could be derived, using only the class of generalized concave functions might not be consistent with the DM's actual preference structure and consequently the DM might be forced to accept the final alternative recommended by the algorithm that he/she was not satisfied with.

Interestingly enough, Malakooti (1990) pointed to the possibility of using a quasiconvex utility function. The possibility of a generalized convex (convex, pseudoconvex or quasiconvex) utility function is supported by multiattribute utility/value theory (Keeney and Raiffa 1976). In particular, multiattribute utility/value functions can be quasiconvex if the single utility functions possess certain properties. Assuming a risk-prone decision maker with respect to each objective, the single utility functions $u_i(f_i)$ will be convex. Consequently, if the additive utility model is used to represent the DM's preference behavior, its multiattribute utility function would be convex. Furthermore, the multiattribute utility function of the above model is found to be strictly quasiconvex if each of its single utility function is a strictly convex and increasing. As far as the multiplicative utility model is concerned, it is shown in Chapter 3, that if each of the $u_i(f_i)$'s is a strictly convex function and increasing, its multiattribute utility function is strictly quasiconvex with $-1 < w <$
0. Furthermore, under same conditions, it can behave as a strictly quasiconvex function if each of the $u_i(f_i)$'s is a strictly convex function and increasing with $w > 0$. These results are also applicable to Dyer and Sarin's (1979) measurable value functions. Therefore, in order to improve the quality of the final alternative as well as DM's satisfaction, the notion of generalized convex-based preference will be exploited for the first time in this research. Algorithms to handle this class as well as the class of quasiconcave utilities are developed.

As far as the existing approaches are concerned, a study of a sample consisting of seventy-four existing computerized multiple criteria decision making (MCDM) approaches in the United States and Europe, most of which were developed between 1976 and 1983, revealed that more than forty methods of the sample relied on sophisticated mathematical concepts such as mathematical programming (Bui 1984). However, only six methods are based on quadratic or nonlinear programming, while thirty-five methods are derived from linear programming. In other words, for every six linear programming-based approaches, only one nonlinear programming-based approach was developed.

The motivation for our research in multiobjective optimization is due to the following facts:

1. Restricting the utility function to the class of generalized concave functions, without testing whether it is consistent with the underlying preferences or not, may leave the decision maker unsatisfied with the final recommended alternative. Accommodating
generalized convex-based preferences explicitly in addition to the quasiconcave class is clearly needed. Our research will consider this fact and offer a new flexible integrated approach.

(2) None of the multiobjective methods selected for a certain problem can claim superiority. Furthermore, different approaches may yield different solutions. This is due to the fact that selecting a multiobjective method is in itself a decision with multiple conflicting attributes such as cognitive load, ease of use, generality and validity. Our developed approach contributes effectively with respect to all of these attributes. Preference elicitation will be done through pairwise comparisons or simple tradeoff questions, as opposed to the value estimates required in some of the existing algorithms such as marginal rate of substitution (MRS) or reference-point questions.

(3) The number of general approaches are very few compared to the number of specific (LP-based) ones as shown in the sample studied by Bui (1984). Devising a general approach that handles linear as well as nonlinear multiobjective programming problems is highly desirable. Our approach is designed to handle general multiobjective problems with continuous decision variables.

It is our belief that the above facts represent an invitation to enrich multiple criteria decision making (MCDM) literature with additional methodologies. These new approaches should be able to help accommodating different users with different decision-making styles for different decision problems. Therefore, the overall thrust
of our research is to devise a general interactive and integrated multiobjective optimization methodology. The general framework, a summary of the imbedded algorithms as well as the methodological contribution of the thesis will be presented next.

1.2.2 The Integrated Multiobjective Approach

The developed methodology will be able to handle the general multiobjective programming problem while preserving simplicity and ease of use. The approach will include algorithms that would interact with respect the decision maker, who will be guided through the objective space of alternatives, exploring its promising portions, as well as learning more about the different features of possible outcomes, such as their levels and tradeoffs.

The underlying concept of our integrated approach is to assess the utility function class first that would be consistent with the decision maker underlying preference structure. Two classes will be considered in this research. These are the generalized-concave and generalized-convex classes. The most relaxed form of these two classes will be assumed to represent the DM's preference structure. Therefore, his/her implicit utility function will be expected to be either quasiconcave or quasiconvex. Mathematical programming-based tests are developed to accomplish this task. The test for each class will be based on pairwise preference comparisons of a representative sample of the nondominated set, which can generated by a weighted Tchebycheff-based scheme. Having then decided on the appropriate form of the utility function, the associated
multiobjective algorithm is invoked. A quasiconcave as well as two quasiconvex utility-based algorithms are developed in this research. They are briefly discussed next.

The developed quasiconcave utility-based algorithm will draw upon the work of Frank and Wolf in nonlinear programming (Frank and Wolf 1956). Their algorithm had been adopted by Geoffrion et al. (1972) to solve the multiobjective convex programming problem interactively. Although their interactive algorithm had been well received by the multiobjective research community in the sense that it was one of the first attempts to extend single nonlinear programming methods to the multiobjective case, it exhibit serious drawbacks. These drawbacks include posing hard preference questions to the decision maker, providing him/her with dominated alternatives during the solution process, as well as the potential for concluding prematurely with a dominated final alternative. Furthermore, the termination is automated in the sense that the decision maker does not have the control over the conclusion of the process. The developed algorithm will not only be able to overcome these setbacks, but can also be used to handle the nonconvex feasible region case easily.

The two fundamental differences between our approach and that of Geoffrion et al. are the following. First, our developed algorithm will be able to trace a nondominated curve, rather than a line as in the GDF approach, avoiding the possibility of generating the undesirable (dominated) alternatives. To accomplish this, we use a
sophisticated alternative projection scheme. This will contribute effectively to the possible situation that, whenever the decision maker decides to quit the solution process, the algorithm will guarantee that he/she has a nondominated solution at hand. Secondly, our algorithm provides an interactive termination criterion based on a simple tradeoff questions. This is in contrast to the GDF algorithm, where termination is automated. Consequently, the decision maker will have the ultimate control over the final conclusion of the algorithm. This feature will enhance greatly the credibility of algorithm in the sense that the decision maker has the final say over the final alternative. An outline of the developed algorithm is as follows,

**Step 0**

Generate an initial nondominated alternative.

**Step 1**

(i) Assess the gradient of the DM's utility function at the current solution.

(ii) Solve the direction-finding problem and identify the direction of improving utility.

**Step 2**

Generate a sample of nondominated alternatives along a nondominated curve associated with the gradient improvement direction obtained in step 1 via a weighted Tchebycheff or a Deviation-based projection scheme.
Step 3

Interact with the DM to identify the most-preferred alternative for the above sample.

Step 4

Check for stopping; if the alternative identified in step 3 is the same as the current solution, invoke the interactive termination criterion and stop, if its conditions are satisfied; otherwise update the current solution and go back to step 1.

In contrast to the above quasiconcave utility-based algorithm, the quasiconvex utility-based algorithms we propose here depend on the structure of constraint set of the model under consideration. It turns out that in the case of multiobjective linear programming, the most-preferred solution occurs at an extreme point of the feasible set. The developed interactive MOLP algorithm builds on the work of Falk and Hoffman (1986). The algorithm is initialized by identifying the center of the feasible set $X$ through formulating and solving an optimization problem. This problem's feasible region $C$ is one dimension higher than the original feasible set $X$ and has it as one of its faces. Through a branch and bound framework, the algorithm partially generates vertices of the set $C$. This process will extend the tree of solutions at each iteration. Each vertex generated is projected on the hyperplane that contains the set $X$. Consequently, at the $i$-th iteration, the terminal nodes of the $i$-th tree contain points whose
convex hull defines an enclosing polytope of the problem feasible set X. Those points which might feasible or infeasible are examined by the decision maker, through their associated alternatives, to decide on the one that is most-preferred. Having identified the most-preferred alternative, its associated terminal node is declared for the next expansion. Unless it satisfies the termination criterion, the algorithm iterates by executing the expansion process. A summary of its steps follows:

**Step 0**

Formulate and solve the center-finding problem for the feasible set X. Identify the tree $T^0$ and its node for expansion, denote it $v^*$. Set $i = 0$.

**Step 1**

(i) Expand node $v^*$ to identify its neighboring vertices. generate the ones that are eligible for generation for $T^{i+1}$.

(ii) Project those generated vertices on the hyperplane containing the feasible set X and append those points to their nodes.

(iii) Set $i = i + 1$.

**Step 2**

(i) Screen the alternatives in the terminal nodes for efficiency.

(ii) Present those efficient alternatives to the DM to select the best one $x^S$. Denote its associated node as $v^*$. 


(iii) If the point $x^s$ is feasible, then stop; $x^s$ is the final solution. Otherwise go to step 1.

In the case of multiobjective nonlinear programming, it turns out that, under mild conditions such as the utility function being strict quasiconvex or having positive gradient everywhere, the most-preferred solution is a supported or proper alternative, respectively. Using these results, a heuristic algorithm is designed. It uses the fact that the simple scalarization mechanism of the weighted objectives can be used in this case. Interestingly enough, the newly generated alternative through this scheme is guaranteed to have higher utility values than the current one unless it returns the current alternative again. The needed weights can be provided by assessing the gradient of the utility function at the current point. This algorithm includes the following main steps:

Step 0

Generate an initial nondominated alternative.

Step 1

Assess the utility function gradient at the current alternative, consider it as the current weighting vector.

Step 2

Solve the weighted-objective problem using the above weights.
Step 3
If the current alternative solves the above problem, conclude the algorithm with this alternative and EXIT; otherwise go to the next step.

Step 4
Select any alternative from the weighted-objective problem's solution set. Make it the current alternative and return to Step 1.

Due to the heuristic nature of the above algorithm, a good strategy is to execute it with multiple starting points, where the final alternative from each cycle is accumulated in a list called "FINAL". Whenever the decision maker decides to quit the session, this list is presented and he/she then selects the final alternative.

1.2.3 Methodological Contribution

The developed methodological research will enrich the multiobjective literature with an approach that exhibits the following appealing features:

(1) Flexible, since two general classes of utility functions will be handled. The least restrictive form of generalized concavity or convexity (i.e., quasiconcave or quasiconvex) is assumed to represent the DM's preference. Furthermore, if the DM's preference happens to be inconsistent with both of these classes, the quasiconcave utility-based algorithm can be used as an exploratory search method with
multiple starting points. This feature has not been provided by any previous research. We believe it would be an effective concept for enhancing the overall DM's satisfaction of the final recommended solution.

(2) **General**, since it will be able to handle the very general multiobjective optimization problems. Furthermore, it could be extended to handle multiobjective integer programming as well as discrete multiple criteria decision making problems.

(3) **Easy to Use**, since every algorithm of the approach consists of easy steps. This feature is necessary in order for users (especially naive ones) to understand the underlying problem-solving process. Consequently, they are likely to be more satisfied with it.

(4) **Places Low Cognitive Load**, on users since the DM will be faced only with pairwise preference questions. These type of questions are considered to be easy relative to the other types, such as those based on marginal rate of substitution and reference values. Consequently, the cognitive burden would not be burdensome.

(5) **Credible**. In the case of the quasiconcave utility-based algorithm, all of alternatives presented to the DM during the process are nondominated and consequently the final solution must also be nondominated. Furthermore, the decision maker has control over the conclusion of the process through the interactive termination criterion. In the case of the quasiconvex utility-based algorithms, final alternative recommended by both algorithms (MOLP and MONLP) is guaranteed to be nondominated.
1.3 Scope

The scope of this thesis is as follows. The next chapter consists of a literature review of multiobjective optimization. In Chapter 3, the interactive integrated approach for multiobjective optimization will be presented. Chapter 4 presents the multiobjective algorithm for the quasiconcave utility function. In Chapter 5, multiobjective algorithms for quasiconvex utility function will be outlined. Conclusions and future research will be discussed in Chapter 6. References are in Chapter 7. Appendix I contains the proof of theorem 4.6. Appendix II contains the GENERATE_EVALUATE routine of the quasiconcave utility-based algorithm. Appendix III contains the test problems for the quasiconcave utility-based algorithm. Appendix IV contains the QXMOLP program that implements the quasiconvex utility-based multiple objective linear programming algorithm. Appendix V contains the test problems for the quasiconvex utility-based MOLP algorithm.
2. MULTIOBJECTIVE OPTIMIZATION: A LITERATURE REVIEW

2.1 Introduction

In this chapter, the process of decision making, for a situation considering more than one performance index (also called criterion, objective or attribute) is introduced. This branch of systems analysis is referred to as Multiple Criteria Decision Making (MCDM). One of its subbranches referred to as Multiobjective Optimization (MOP), where our proposed approach belongs, is briefly reviewed. Classification schemes for its solution strategies are presented.

In general, MCDM is the field that is concerned with the process (i.e. methods and procedures) of selecting a possible course of action (i.e. decision alternative) from a finite or infinite set of available alternatives in a complex decision situation. The complexity of the decision situation (problem) is due to the presence of multiple decision criteria rather than a single one which are non-commensurable and often conflicting. Another characteristic of these decision problems is the necessity of the human element involvement as decision maker(s). This will certainly have an impact on the final solution of the problem since his/her (their) decision-making style(s) should be captured as a part of the solution process.

Hence analysis methods and procedures, through using the available technologies such as mathematical models and computers, are derived to clarify the decision situation in order to have a
successful overall decision-making session in the sense that the
DM(s) has a satisfactory conclusion of the analysis process.

In the next section, the problem definition and related
terminology are presented, also a classification of the multicriteria
decision problems under certainty is included. In section 2.3, the
category of multiobjective optimization (mathematical programming)
is introduced, and its solution strategies will be classified.
Representative solution approaches for each class are included.

2.2 The MCDM Problem: Definition and Terminology

Multiple criteria decision making problem formulation consists
of the following two mathematical entities:

(A) a set of feasible decisions \( X \subset \mathbb{R}^n \).

(B) a set of scalar-valued criterion functions \( f_1, f_2, \ldots, f_k \)
defined on \( X \), without loss of generality, all to be maximized.

A general formulation might be:

"Select a feasible decision \( x^p \), such that its associated
outcome \( f^p \) is the best outcome (alternative) from the
decision maker's point of view".

Notice that the above formulation is highly subjective since in
order to decide whether outcome \( f(x) \) is better than outcome \( f(y) \), or
whether outcome, \( f(x) \) is the best among \( \{f^1(x), \ldots, f^r(x)\} \) outcomes
is the ultimate responsibility of the decision maker (DM).
Monotonicity Assumption

For a rational DM, his preference is assumed to be a monotone function of the value of each objective (criterion).

This assumption implies that the decision maker's satisfaction will never decrease as fi increases (all other objectives held fixed). In other words, the more of each objective, the better.

Definition (Ideal Outcome)

The point (outcome) $f^* = (f_1^*, f_2^*, \ldots, f_k^*)$ in the objective space $F$ is called the ideal point (outcome) where $f_i^*$ is the maximum of $f_i(x)$ s.t. $x \in X$ for all $i = 1, \ldots, k$.

Since the above objectives are assumed to be conflicting, the outcome $f^*$ is not feasible, since these indices cannot be maximized at the same time. Therefore the final solution to the problem should be a compromise-based solution from the DM perspective. Hence the best outcome selected to be the final solution will be denoted as the most-preferred or the best-compromise solution.

Definition (Dominance)

Given $x, y \in X$, alternative $f(y)$ is dominated (strongly dominated) by alternative $f(x)$ if and only if $f(x) \succeq f(y)$ ($f(x) > f(y)$).

\* $x \succeq y$ denotes $x_i \geq y_i$ and $x \neq y$. 
Definition (Efficiency)

A solution $x \in X$ is an efficient (weakly efficient) solution if and only if there does not exist a solution $y \in X$ such that alternative $f(y)$ dominates (strongly dominates) alternative $f(x)$.

The latter definition implies that for the weakly efficient solutions, there exist another solution which improves some criteria while other criteria remain unchanged.

Definition

The nondominated frontier "N" is the set of all nondominated outcomes in the objective space.

Deterministic multicriteria decision problems can be classified into two broad categories with respect to the way that alternatives are represented. This classification is similar to the one given in (Zionts 1982).

I. Multiattribute Decision (MAD) problems

Refer to the problems where decision alternatives are explicitly represented. These alternatives might be listed in a tabular form, called the payoff matrix, where each one is represented as a row in the matrix. These decision problems might have small number of alternatives such as buying decisions (e.g., car, personal computer or house) or siting large and complex facilities (Hobbs 1979).
II. Multiobjective Optimization (MOP) problems

Refer to the problems where decision alternatives are implicitly represented. The feasible decision set and the objectives are implicitly defined through explicit functional forms of the decision variables. These problems commonly have a large number of feasible alternatives, often infinite, where it is not practical to enumerate and/or process them at the same time. This class might be viewed as an extension of the well-known single objective optimization in the sense that efficient algorithms are needed for reducing the large set of the candidate alternatives to a small, possibly singleton, set of recommended solutions.

The latter category, which is of interest to us, has been one of the fastest growing areas of the systems analysis research during the last fifteen years. This is due to the wide recognition that most of decision problems, whether previously formulated with respect to a single objective or not, are multidimensional in nature and should be addressed in a multiple objective framework in order to arrive at a satisfactory final solution. Since our proposed approach intends to handle such decision problems, multiobjective optimization approaches will be reviewed next.

2.3 Multiobjective Optimization (MOP)

In this class, the problem is expressed in the following form:
Maximize \( f = \begin{vmatrix} f_1(x) \\ f_2(x) \\ \vdots \\ f_k(x) \end{vmatrix} \)

subject to:
\( x \in X = \{ x \mid g_i(x) \leq 0, \ i = 1, \ldots, m \}. \)

Several classification schemes have been presented in the literature for the solution methodologies of this particular class (e.g., Chankong and Haines (1983), Cohon and Marks (1975), Evans (1984), Hwang et al. (1980)). However, these classifications differ from each other, since they are based either on a key element of the solution process such as the timing of DM's preference information elicitation, a key element of the decision problem such as the type of decision variables (i.e. continuous or discrete) or a combination of both.

Solution approaches for multiobjective optimization can be broadly classified into three categories according to the underlying concepts of the overall solution strategies as:

(1) Single Solution-based Approaches.

(2) Representative Generation-based Approaches.

(3) Progressive Generation-based Approaches.

These classifications will be briefly discussed next, and representative procedures for each category will be reviewed.
23.1 Single Solution-based Approaches

This class of solution procedures refers to the ones that do not generate any alternatives to be examined by the decision maker before a final solution is recommended. Hence the multiobjective problem is transformed into a single objective optimization problem which could be solved using appropriate procedures for single objective optimization models. The problem transformation is done by one of two ways,

(A) Methods without preference articulation: these refer to the type of solution approaches that do not use any preference information elicited from the decision maker, rather it assumes that he/she is able to accept the final solution obtained. The advantage of this approach is that it does not place any type of information burden on the DM, but it has the disadvantage that it relies on the analyst to make assumptions related to the DM's preference which might not be consistent with the true underlying preference structure.

An example of this category is the method of the global criterion (Hwang et al. 1980). In this approach, the multiobjective problem is transformed to the following single optimization problem:

\[
\text{Minimize } d_p = \Sigma_{i=1}^{k} ((f_i^*(x) - f_i(x))/f_i^*(x))^p \\
\text{subject to } x \in X, \\
\text{where, } f_i^*(x) \text{ is maximum of objective function } f_i(x) \text{ and } p \geq 1 \\
\text{to be chosen arbitrary by the analyst.} 
\]
(B) Methods with preference articulation: these refer to the type of solution approaches that assume that the DM is able to supply useful information related to his underlying preference structure. Explicit utility function approach (Keeney and Raiffa 1976) is one of the pioneering representative methods of this subclass. In this approach, a scalar-valued function, called utility or value function "U", defined on the objective space, is constructed. Using this function, the multiobjective model is transformed to a single problem which can solved by an appropriate optimization method. Hence, the single objective formulation will be as follows:

Maximize \( U(f_1(x), f_2(x), \ldots, f_k(x)) \)

subject to \( x \in X \).

The overall utility \( U(f(x)) \) can be of many forms. Based on the assumptions of preferential and utility independence among the objectives (Keeney and Raiffa 1976), the following additive and multiplicative forms are commonly used, assuming that a single utility for each objective function has been identified,

for the additive form:

\[
U(f(x)) = \sum_{i=1}^{k} w_i u_i(f_i(x)),
\]

where \( w_i \geq 0 \) and \( \sum_{i=1}^{k} w_i = 1 \),

for the multiplicative form:

\[
1 + wU(f(x)) = \prod_{i=1}^{k} [1 + ww_i u_i(f_i(x))],
\]

where \( 1 + w = \prod_{i=1}^{k} [1 + w w_i] \), \( w > -1 \) and \( w \neq 0 \).
Although this approach makes current optimization concepts and techniques immediately usable, utility function assessment involves a very delicate process. Eventhough a particular functional form can be specified to represent a particular DM's underlying preference structure, it might be well suited locally but not globally.

2.3.2 Representative Generation-based Approaches

In this class of solution procedures, a representative nondominated subset is constructed using an appropriate generating technique such as a simplex-based method or the ε-constraint method. A final solution alternative is chosen either arbitrary (i.e. without any type of DM's value judgment articulation such as by presenting the available set to him/her to select one), or by using a value judgment assessment form such as the worth function (Haines et al. 1974).

There are two main concerns related to the generated representative set that can affect the final outcome of this type of solution strategy which are:

(1) whether the size of the generated subset is enough to capture the ranges of the objectives over the nondominated frontier or not.

(2) whether the generated subset is considered to be a nonbiased representation of the nondominated frontier or not.

This class of solution procedures might be categorized into two groups with respect to value judgment articulation as:

(A) Methods without preference articulation.
(B) Methods with preference articulation.

which will be briefly discussed next together with an example approach for each category.

(A) Methods without Preference Articulation: these refer to the approaches that involve only generating a sample of nondominated alternatives. The final solution is chosen arbitrarily by presenting the sample to the decision maker. A major drawback of this type of methods is that it is very difficult for the DM to handle a large number of alternatives at the same time if the objectives are greater than three. This will subsequently affect the final solution. An example method of this category is a simplex-based method presented next.

The Three Phase Method of Evan and Steuer (1973)

Evan and Steuer (1973) introduced a three phase scheme to generate all of the extreme nondominated vertices associated with the nondominated frontier of the MOLP problems as a representative set. In their approach, Phase I and II, which are based on the linear programming theory, are used to find an initial extreme point and an initial efficient extreme point respectively. Phase III, however, uses an adjacent efficient extreme point method to locate all of the remaining efficient extreme points of the MOLP problem.

(B) Methods with Preference articulation: these refer to the solution approaches that not only generate a representative set but also use an information assessment form to help articulating DM's
value judgment so that a final solution can be recommended. A pioneering approach developed by Haimes, Hall, and Freedman (1974) is briefly reviewed next as a representative procedure of this category.

The Surrogate Worth Tradeoff (SWT) Method of Haimes et al. (1974)

Haimes, Hall and Freedman (1974) developed an approach for solving the multiobjective nonlinear programming problem based on the e-constraint generating scheme. The approach consists of three main steps. First, a representative subsets of nondominated alternatives are generated with respect to every two objective functions keeping the rest of the objectives fixed at predefined levels. Then, a form of subjective tradeoff assessment, called the surrogate worth function, is assessed with respect to every two objectives through interaction with the DM using the objective generated tradeoffs. These tradeoffs correspond to the values of the strictly positive Kuhn-Tucker multipliers resulted from the first step. The worth function $W_{rij}$ represents DM's degree of preference toward trading $l_{rij}$ units of the reference objective $f_r$ for one unit of objective $f_j$, where the DM responds on a scale between +10 and -10 indicting his degree of acceptance or rejection. Using a curve fitting technique, a functional form relating the worth values $W_{rij}$ to the levels of the objective function $f_j$ is constructed. Finally, a system of the equations (often nonlinear) $W_{ij} = 0$ for all $i, j = 1, \ldots, k$, $i \neq j$, is solved in order to determine the acceptable levels of objectives which are used to recommend the final solution. Although the
methodology involves a substantial computational effort starting with the generation process then the curve fitting then the solution of a system of nonlinear equations, it enjoys the desirable feature that its preference questions can be handled easily by the decision maker.

2.3.3 Progressive Generation-based Approaches

This class of solution procedures refers to the methods that progressively generate an alternative or group of alternatives during the solution process. The idea behind them is to involve the decision maker during the successive exploration of the objective space (only the nondominated portion in most methods) along with the progressive articulation of his/her underlying preference structure.

This group of methods generally referred to as Interactive methods are very popular in the multiobjective literature due to the following facts:

(1) The nondominated frontier may be too large to generate completely.

(2) Their recognition of the importance of the involvement of the decision maker during the solution process so that he learns about the possible outcomes of the problem in order to arrive at a satisfactory final solution.

(3) Progressive learning as well as articulation of the DM's underlying preference structure are done through a relatively simple question-answering scheme, since the human ability of processing a set of alternatives simultaneously is limited
(4) They are relatively easy to implement since most of them are based on the well-known mathematical programming techniques. Several man-machine interactive approaches have been proposed in the literature (e.g., Chankong and Haimes 1978, Geoffrion et al. 1972, Ho 1979, Malakooti and Ravindran 1985/86). Although different types of these methods exist, they commonly consist of two stages which alternate at each iteration of the solution process. These are decision-making and computation stages.

At the decision-making stage, the DM is provided with an alternative or group of alternatives to be examined in order for him/her to supply value judgement information based on the tradeoffs between objectives, preference comparison between alternatives or a combination of both. However, at the computation stage, a new alternative or group of alternatives are generated based on the value judgment information elicited from the DM during the decision-making stage hoping for better one(s).

In order to systematically review the literature of the interactive approaches, the solution methods will be classified with respect to the subjective information demanded from the decision maker at the decision-making stage. From a particular decision maker point of view, the type of DM's value judgment information assessed at this stage is very crucial. It determines whether a certain approach can be viewed as a better decision-aid or not with respect to another that could be used to solve the same problem. However, before presenting these classes, the different types of value
judgment information that have been suggested throughout the literature are discussed next.

2.3.3.1 Value Judgment Information

At the decision-making stage, the emphasis is on the human factor rather than the mathematical structure of the decision problem as in the computation stage. Since the involvement of the human factor is due to the assumptions on the ability of the DM to articulate his/her value judgment, assuming that he/she has an implicit utility function to represent his/her value judgment (preference) structure, several types of value judgment information have been used at the decision-making stage of each iteration of the interactive solution process. They can be clustered into two types of information: Tradeoff-based information and Preference Comparison-based information. These are discussed next.

(i) Tradeoff-based information

This type of subjective information is an assessment of marginal information of the assumed implicit utility function. The following forms of marginal information were proposed in the literature (Geoffrion et al. 1972; Haines et al. 1974).

(a) Marginal Rate of Substitution (MRS): also called the tradeoff rate, MRS is defined as the amount of one objective (say $f_r$) that the decision maker is willing to sacrifice in order to gain an extra unit of another (say $f_j$) given the levels of the other objectives fixed. Mathematically, the MRS denoted as $M_{rj}(f^*)$ is represented by the
negative slope of the indifference curve at point \( f^* \) in the objective space as :

\[
M_{rj}(f^*) = \frac{\partial U(f^*)/\partial f_j}{\partial U(f^*)/\partial f_r} = \\
= - \left( \frac{\partial f_r}{\partial f_j} \right)_i \, dU = 0, \, df_i = 0, \, i \neq r, j.
\]

where each indifference curve is defined as the locus of points in the objective space for which the DM is indifferent (i.e. \( U(f^*) \) is constant).

(b) **Worth Assessment Information:** the concept of worth value assessment for generated tradeoffs between any two objective functions, keeping the rest of the objectives fixed at certain levels, was introduced by Haines et al. (1974). Given a feasible tradeoff \( T_{rj}(f^*) \) at a current solution \( f^* \) between the reference objective \( "f_r" \) and any other objective \( "f_j" \) (\( j \neq r \)), the decision maker may express his/her subjective worth value \( "W_{rj}" \) of trading \( T_{rj}(f') \) units of objective \( "f_r" \) for one unit of objective \( "f_j" \) by assigning an integer value between -10 and +10 to \( "W_{rj}" \). A high positive value of \( "W_{rj}" \) implies that the DM has a strong desire to approve the trade, and a high negative value of \( "W_{rj}" \) implies that the DM has a strong desire to disapprove it, whereas assigning \( "W_{rj}" \) a value of zero, the DM is indifferent to the trade.

(c) **Explicit Tradeoff Evaluation:** in this form, a vector of explicit tradeoffs involving all of the objectives at a current solution is presented to the DM for evaluation where he/she indicates his/her acceptance, rejection or indifference (Zionts and Wallenius 1976, 1983).
(d) **Aspiration Level Adjustment:** in this case, the DM is asked to indicate which objective he/she is willing to relax in order to improve another, and the amount of relaxation he/she is willing to accept based on the current solution levels (Benayoun et al. 1971).

(ii) **Preference Comparison-based Information**

In this type of subjective information, the decision maker may indicate his/her preference judgment with respect to a set of two or more alternatives presented to him/her at the decision-making stage of each iteration. The following different situations for assessment were used (Malakooti and Ravindran 1985/86; Steuer and Choo 1983).

(a) **Pairwise Comparison:** the DM is provided with two alternatives and asked to indicate which he prefers, if any.

(b) **Pairwise Comparison with Strength of Preference:** the DM is provided with two alternatives and not only asked to indicate whether he prefers one to the other but also asked to express his/her strength of preference as strong, weak, or medium.

(c) **Best (Worst) Alternative Selection:** the DM is provided with a set of more than two alternatives and asked to select the best (or worst) among them. These preferences imply constraints on acceptable tradeoffs which can be used to generate search directions

### 2.3.3.2 Classes of Interactive Methods

According to the two stated types of value judgment information above, the interactive approaches developed to solve the
multiobjective mathematical programming problems with continuous decision variables can be classified into three categories as follows:

(1) Tradeoff-based Approaches.
(2) Preference Comparison-based Approaches.
(3) Combined Tradeoff and Preference Comparison-based Approaches.

Representative approaches for each category will be briefly discussed.

(1) **Tradeoff-based Methods**

Refer to methods that use any of the above information forms based on tradeoffs.

(i) **The Interactive Surrogate Worth Tradeoff (ISWT) Method** 
(Chankong and Haines 1978)

Chankong and Haines (1978) presented an interactive and computationally more efficient version of the surrogate worth tradeoff method pioneered by Haines et al. (1974). At each iteration of the method, the n-constraint generating scheme is used to generate a nondominated alternative, where its associated tradeoffs correspond to the values of the strictly positive Kuhn-Tucker multipliers of the constrained objectives. At the interaction step, the worth values of the objective generated tradeoffs are assessed to determine a feasible direction in which the implicit utility function increases in the objective space. This interactive version reduces the computational requirement of the originally developed approach (i.e. SWT) substantially, since neither curve fitting nor solving a system
of nonlinear equations is required. Also, unlike the original SWT, this algorithm yields the globally optimal solution under certain mild conditions.


Zionts and Wallenius (1976, 1983) developed an approach for handling problems with concave objective functions and a convex constraint set. The 1976 version of the approach was designed to handle the linearized version of the model and assumed a linear additive utility function. However, the 1983 version handles only the MOLP case and assumes an implicit concave utility function which is approximated locally by a linear additive utility function. At each iteration, the DM is provided with the current solution together with the efficient tradeoffs associated with the adjacent nondominated extreme points, which he/she is required to evaluate. Based on the DM's responses, constraints on the optimal set of weights $w_1^*, w_2^*, \ldots, w_k^*$ are formulated. Using the set of constraints formulated at the current and the previous iterations, a new set of consistent weights is determined for the next iteration. The procedure stops whenever no improvement can be achieved beyond the current efficient extreme point. The main drawback is that the nondominated alternative with the highest utility value often not be an extreme point due to the nonlinearity of the utility function. The method will not be able to generate such points.
(iii) The Step Method (STEM) of Benayoun, De Montgolfier, Tergny and Laritchev (1971)

Benayoun et al. (1971) developed the step method (STEM) which was one of the first techniques to employ a progressive generation of alternatives to solve the Multiple Objective Linear Programming (MOLP) problem. At each iteration, a nondominated alternative based on a single optimization problem which minimizes the maximum weighted distance from the ideal point is generated. Based on this solution, levels of certain objectives might be relaxed by the DM in order to improve the others. The procedure is concluded whenever the DM is satisfied with the attainable levels of objectives. Due to the utilization of the weighted Tchebycheff scheme for generating alternatives, weakly nondominated alternatives might be generated.

(2) Preference Comparison-based Methods

Refer to methods that use any of the information forms based on the preference comparison mentioned above.

(i) The Method of Malakooti and Ravindran (1985/86)

Malakooti and Ravindran (1985/86) developed an approach for solving MOLP problems similar to the Zionts and Wallenius (1976) approach. The concepts of utility efficiency and the assessment through the strength of preference based on an assumed linear additive utility function were introduced to eliminate utility-inefficient adjacent extreme points at each iteration. This reduces the
burden on the DM and speeds up the convergence of the algorithm since more subjectively useful information, using the concept of strength of preference, can be easily extracted from the decision maker.

(3) Combined Tradeoff and Preference Comparison-based Methods

Refer to solution methods that use a combination of tradeoffs and preference Comparisons.

The Method of Geoffrion, Dyer and Feinberg (1972)

Geoffrion, Dyer and Feinberg (GDF) (1972) presented an approach to handle multiobjective nonlinear programming problems with a convex constraint set. Their approach is based on a single objective nonlinear programming algorithm developed by Frank and Wolf (1956). Geoffrion et al.'s approach is the first method that used the concept of marginal rate of substitution in order to estimate the direction of the gradient for the assumed implicit concave utility function. Using the direction vector of the gradient, the direction finding problem is formulated in order to obtain a feasible direction for which the utility function will rapidly increase. The obtained feasible direction is used to search for better alternatives in a one-dimensional scheme, where a set of feasible alternatives generated along this direction are presented to the decision maker for preference comparison. Since our proposed quasiconcave-based approach is similar to this one in the sense that they are both based on the Frank and Wolf (1956) method of NLP, the G-D-F approach
will be presented in detail in the next chapter, where its pros and cons will be discussed.
3. THE INTEGRATED MULTIOBJECTIVE APPROACH

3.1 Introduction

In this chapter, a general man-machine interactive integrated approach for solving general multiobjective optimization problems is developed. The approach has the unique feature of not only accommodating decision makers with quasiconcave utility functions but also those that may exhibit quasiconvex utility preference structures. This important feature has not been provided by any previous multiobjective methodology. The approach’s imbedded interactive algorithms will be able to handle the general nonlinear case and take into consideration certain characteristics of the nondominated alternatives, which may arise due to nonlinearity, that could not be handled by most of the interactive methods reviewed in Chapter 2. Furthermore, the imbedded algorithms will take into account the decision maker's desire to lessen the burden of the preference information demanded at each iteration of the solution process.

This chapter consists of seven sections. In the next section, certain properties for nondominated alternatives will be presented. This is to justify the suggestion of using the Tchebycheff method in the process of sampling the nondominated frontier as well as in accomplishing the projection process within the quasiconcave utility-based algorithm. In Section 3.3, the Tchebycheff generation scheme and its implementation will be discussed. This scheme is needed to
generate a representative sample of the nondominated frontier for use in utility class testing. Classes of utility function, an analysis of the standard utility/value models, and a testing theory will be outlined in Section 3.4. The analysis indicates conditions under which the multiattribute utility/value function behaves as quasiconcave or quasiconvex. However, testing theory indicates conditions under which a sample of alternative is inconsistent with a quasiconcave or quasiconvex utility function. The steps of the integrated approach that can handle both quasiconcave and quasiconvex utility functions will be presented in Section 3.5. Implementation issues are discussed in Section 3.6. Illustrative example for utility class testing and conclusions are presented in Sections 3.7 and 3.8 respectively.

3.2 Properties of Nondominated Alternatives

Two relevant properties of nondominated alternatives have been introduced in the literature. They are properness and supportedness. These two properties are important because the nondominated frontier that is needed to be sampled may contain all kinds of alternatives with respect to these properties and therefore the use of the Tchebycheff generating method is justified. Also, they help characterize the most-preferred solution set in the case of quasiconvex utility functions. The two properties are discussed next.
3.2.1 Properness

Definition (Geoferion 1968)

A nondominated alternative \( f(y) \) is properly nondominated if there exists a real number \( M > 0 \) such that for each \( i \), we have 
\[
[(f_i(x)-f_i(y)) / (f_j(y)-f_j(x))] \leq M \text{ for some } j \text{ such that } f_j(x) < f_j(y)
\]
whenever \( x \in X \) and \( f_i(x) > f_i(y) \).

This definition implies that the ratio of the marginal gain of the \( i \)-th objective to the marginal loss in the \( j \)-th objective should be bounded for the alternative to be properly nondominated; otherwise it is improperly nondominated. Inverse images of properly and improperly nondominated alternatives (vectors in the objective space) are properly and improperly efficient solutions in the decision space respectively. They might occur in nonlinear multiobjective programming problems with a nonconvex objective feasible region.

3.2.2 Supportedness

The concept of supportedness, also called convexity domination, was introduced by Zionts (1980).

Definition

A nondominated alternative is unsupported if it is dominated by some convex combination of other nondominated alternatives; otherwise it is supported.

Inverse images of supported and unsupported nondominated outcomes (in the objective space) are denoted as supported and
unsupported efficient solutions (in the decision space) respectively. Those types of alternatives might occur in multiobjective nonlinear and integer programming problems.

**Definition**

A supported nondominated alternative is a supported-extreme if it is an extreme point of the objective space "F", otherwise it is supported-nonextreme alternative.

Supported-extreme and nonextreme alternatives occur in multiobjective linear programming (MOLP) problems as extreme points and facets of the feasible region in the objective space, respectively.

**3.3 The Tchebycheff-based Sampling Scheme**

This method is reviewed here because it is an integral part of the initialization phase of our integrated approach. Several different schemes have been developed for generating nondominated alternatives such as the weighting method, the ϵ-constraint, the proper equality constraint, the hybrid method, and the weighted norm method (Chankong and Haimes 1983; Cohon 1978). In this subsection, various weighted Tchebycheff generating schemes and their implementation will be discussed. Before presenting these techniques, the notion of the Lp-distance metric between two points is presented.

The distance between the two points $z^1$ and $z^2 \in \mathbb{R}^k$ according to the Lp-metric is defined as [Steuer 1986]:
\[ d_p = \|z^1 - z^2\|_p = [\Sigma_{i=1}^{k} |z_{i1} - z_{i2}|^p]^{1/p}, \quad 1 \leq p \leq \infty \] (1)

Therefore, the distance between the two points changes as the value of the parameter "p" changes, where the values of "d_p" using \( p = 1 \) and \( p = \infty \) represent the upper and the lower bounds on the distance between the two points respectively, (i.e. \( d_\infty < d_p < d_1 \) for \( 1 \leq p \leq \infty \)) (Steuer 1986). As the value of the parameter "p" increases, more weight is assigned to large component differences, since all of the components \( (z_{i1} - z_{i2}) \) will be raised to increasing values of "p".

The lower bound metric that corresponds to \( p = \infty \) is known in the literature as the Tchebycheff metric, where an infinite weight is assigned to the largest component difference. Therefore, the distance "d_\infty" according to the Tchebycheff metric between the two points \( z^1 \) and \( z^2 \) is determined as follows:

\[ d_\infty = \|z^1 - z^2\|_\infty = \max_i \{ |z_{i1} - z_{i2}|, \quad i = 1, \ldots, k \}. \]

So far, the distance according to the this metric is determined assuming the component differences are equally weighted. A more general weighted version of the above metric, called the weighted Tchebycheff metric, can be defined as follows:

\[ d_\infty^w = \|z^1 - z^2\| = \max_i \{ w_i |z_{i1} - z_{i2}|, \quad i = 1, \ldots, k \} \]

where,

\[ w \in W = \{ w \in \mathbb{R}^k | w_i > 0, \Sigma_{i=1}^{k} w_i = 1 \}. \]

Using the above weighted Tchebycheff metric, nondominated alternatives can be generated with respect to a predefined reference point outside the objective feasible region using the weight "w" as a control parameter (Bowman 1975). The concept of the Tchebycheff-
based generating technique is to determine the nondominated alternative closest to the predefined reference point according to the weighted Tchebycheff distance metric.

What makes this metric powerful is that it has an L-shaped contour with a vertex that can prop any nondominated frontier regardless of its shape by just changing the weights. Therefore it enjoys the following appealing characteristics:

(1) It can generate supported-nonextreme nondominated alternatives uniquely, such as alternatives belonging to the nondominated facets of the MOLP problems.

(2) It can generate unsupported nondominated alternatives uniquely, such as those that might occur in multiobjective nonlinear and integer programming problems.

(3) It can generate improper nondominated alternatives that might occur in multiobjective nonlinear programming problems.

Using the Utopian point "f**" as a reference point, which is obtained by moving the Ideal point "f*" further from the nondominated frontier by a vector ε of sufficiently small positive scalars, the weighted Tchebycheff-based generating problem is formulated as follows:

Problem 3.1: minimize \[ d^{W^*} = \|f** - f(x)\|^{W^*} \]
\[
\begin{align*}
  & x \\
  \text{s.t.} \\
  & x \in X.
\end{align*}
\]

where,
\|f^{**} - f(x)\| = \max \{w_1|f_1^{**} - f_1(x)|, \ldots, w_k|f_k^{**} - f_k(x)|\},
\quad i = 1, \ldots, k
\]

\[f^{**} = f^* + \varepsilon; \quad \varepsilon > 0\]

\[w \in W = \{w \in \mathbb{R}^k \mid w_i > 0, \Sigma_{i=1}^k w_i = 1\},\]

In order to implement the above weighted Tchebycheff generating problem 3.1, the following equivalent mathematical program is used,

Program 3.1: minimize \{ \alpha \}
\[s.t. \]
\[w_i(f_i^{**} - f_i(x)) \leq \alpha \quad 1 \leq i \leq k\]
\[x \in X.\]

The solution to the above weighted minimax formulation of program 3.1 is a vector of the form \((x', \alpha') \in \mathbb{R}^{n+k+1}\), where \(f'(x')\) is the closest alternative to the reference point "f**" in the objective space and \(x\) is its inverse image in the decision space. Unfortunately, the weighted Tchebycheff-based generating program 3.1 guaranties only weakly efficient solutions (Choo and Atkins 1982). However it might have an alternative optima, for some \(w > 0\), where at least one of the solutions is efficient (Steuer and Choo 1983).

To avoid this major setback, Steuer and Choo (1983) developed a modified version of the above weighted Tchebycheff metric. It is called the "augmented weighted Tchebycheff" metric and has the following form:

\[d_{W_{\infty}} = \|f^{**} - f(x)\|_{W_{\infty}} = \|f^{**} - f(x)\|_{W_{\infty}} + \rho \epsilon \|f^{**} - f(x)\|,\]

where \(\rho\) is a sufficiently small positive scalar and \(\epsilon^t\) is a vector of ones. Its associated Tchebycheff-based generating program follows:
Program 3.2: \[ \text{minimize } \{ \alpha + \rho e^{t}(f^{**} - f(x)) \} \]
\[ \text{x} \]
\[ \text{s.t.} \]
\[ w_{i}(f_{i}^{**} - f_{i}(x)) \leq \alpha \quad 1 \leq i \leq k \]
\[ x \in X \]

Although program 3.2 will be able to return a unique nondominated alternative, it will not be able to generate improper nondominated alternatives. An alternative formulation of the above metric called the "lexicographic weighted Tchebycheff" formulation was also suggested in the same reference in order to avoid that. The formulation follows:

Program 3.3: \[ \text{Lex minimize } \{ \alpha , \rho e^{t}(f^{**} - f(x)) \} \]
\[ \text{x} \]
\[ \text{s.t.} \]
\[ w_{i}(f_{i}^{**} - f_{i}(x)) \leq \alpha \quad 1 \leq i \leq k \]
\[ x \in X \]

"Lex" denotes lexicographic implementation, which implies that two stages of optimization will be required if the first stage returns an alternative optima. In the first stage, distance optimization is performed based on the weighted Tchebycheff metric, while the second stage is implemented based on the $L_{1}$-metric. However, performing two stages of optimization might be costly from the computation point of view.

As a sort of a compromise between the above three Tchebycheff-based programs, a similar formulation that is based on a "modified weighted Tchebycheff" metric, developed by Kaliszewski
(1987), can be used. This version has been suggested but never been used in any multiobjective method. Its distance metric is as follows:
\[ d^W_{\infty} = \|f^{**} - f(x)\|_W^W \rho_{\infty} = \max_i \{w_i[|f_i^{**} - f_i(x)| + \rho e^t|f^{**} - f(x)|]\} \]

In order to implement this modified metric version, the following equivalent mathematical program is used:

Program 3.4: minimize \( \{\alpha\} \)

s.t.
\[ w_i[(f_i^{**} - f_i(x)) + \rho e^t(f^{**} - f(x))] \leq \alpha \quad 1 \leq i \leq k \]
\[ x \in X. \]

This formulation exhibits the same important feature that the nondominated alternatives are uniquely computable as in the augmented tchebycheff generating program 3.2. It does not only provide the same result as the augmented version does with respect to the convex problems, but also it can provides better approximation for improper alternatives in the nonconvex case. The reason behind that lies in the difference between the shape of their contours. The contour of the modified version varies only with respect to the parameter "\( \rho \)" which can be controlled easily, while the contour of the augmented version varies not only with "\( \rho \)" but also with tchebycheff weights "\( w_i \)" which should assume various values at different alternatives. The angles are \( \theta_i = \theta = \tan^{-1}[\rho/1+\rho] \) and \( \theta_i = \tan^{-1}[\rho/1-w_i+\rho] \) for the two contours respectively.

An important final remark is that for all of the above formulations, the generated alternatives are not guaranteed to be globally nondominated in the case of problems with a nonconvex
feasible region. This is because the generated alternative may only be a local, and not a global minimum for the formulated problem. However, any global optimum for a Tchebycheff-based generating program is also nondominated.

3.4 Classes Of Utility Functions

The underlying preference structure of the decision maker, assumed to be monotonic, is usually represented as an implicit real-valued monotonic function in interactive methods. Different shapes of utility functions have been assumed throughout the literature. These forms include linear (e.g., Malakooti and Ravindran 1985/86), concave (e.g., Zions and Wallenius 1983), psuedoconcave (e.g., Korhonen and Laakso 1986) and quasiconcave (e.g., Malakooti 1988b). Although quasiconcave utility functions have been well received as a viable assumption, it was not until recently that quasiconvex utility functions have been also suggested to be viable (Malakooti 1990). As noted in subsection 1.2.1, multiatribute utility/value functions (Keeney and Raiffa 1976) can be quasiconvex for a risk-seeking decision makers. Recently, Steuer (1990) emphasized the need for considering this type of utility function. In this section, we assume that the global class of utility functions contains the class of quasiconcave as well as the class of quasiconvex functions. The class of quasiconcave utility functions includes concave as well as pseudoconcave functions, since any concave or pseudoconcave function is also quasiconcave. Also, the class of quasiconvex utility functions includes convex as well as
pseudoconvex functions, since any convex or pseudoconvex function is also quasiconvex. Next, definitions of quasiconcave as well as quasiconvex functions are presented.

Definition (Bazaraa and Shetty 1979)

Let \( U: V \to \mathbb{R} \), where \( V \) is a nonempty convex set in \( \mathbb{R}^k \). The function \( U \) is said to be quasiconcave if, for each \( f^i \) and \( f^j \in V \), the following inequality is true:

\[
U[\alpha f^i + (1 - \alpha)f^j] \geq \min \{U(f^i), U(f^j)\} \quad \text{for each } \alpha \in (0,1).
\]

Definition (Bazaraa and Shetty 1979)

Let \( U: V \to \mathbb{R} \), where \( V \) is a nonempty convex set in \( \mathbb{R}^k \). The function \( U \) is said to be quasiconvex if, for each \( f^i \) and \( f^j \in V \), the following inequality is true:

\[
U[\alpha f^i + (1 - \alpha)f^j] \leq \max \{U(f^i), U(f^j)\} \quad \text{for each } \alpha \in (0,1).
\]

In the next subsection, the additive and multiplicative utility/value models of Keeney and Raiffa (1976) are analyzed with respect to their single utility functions. These results are also applicable to Dyer and Sarin's (1979) measurable value functions. The analysis indicates conditions under which the multiattribute utility/value function behaves as a quasiconcave or quasiconvex. In subsection 3.4.2, a utility class testing theory is presented.

3.4.1 Analysis of Standard Utility/Value Models

In this subsection, the utility/value models of Keeney and Raiffa (1976) are analyzed with respect to their single utility
functions. These models are among the pioneering preference models that exist in the literature. Their underlying concept is that a single attribute utility/value function for each objective is constructed. Then, a multiattribute scalar-valued function, called utility or value function "U", defined on the objective space, is used to aggregate all of the single utility functions. The overall utility \( U(f(x)) \) can be of many forms. Based on the assumptions of preferential and utility independence among the objectives, the following additive and multiplicative models are commonly used, assuming that a single utility for each objective function has been identified. The additive model is as follows,

\[
U(f(x)) = \sum_{i=1}^{k} w_i u_i(f_i(x)), \quad \text{where} \quad \sum_{i=1}^{k} w_i = 1,
\]

The multiplicative model is as follows,

\[
1 + wU(f(x)) = \prod_{i=1}^{k} [1 + w w_i u_i(f_i(x))],
\]

where \( 1 + w = \prod_{i=1}^{k} [1 + w w_i] \), \( w > -1 \) and \( w \neq 0 \).

The risk attitude of the decision maker toward different levels of an objective influences the shape of its single attribute utility function. In particular, if the decision maker is risk averse toward a certain objective, the single utility function is concave. Most of the utility functions estimated by Keeney and his colleagues (e.g., Keeney and Raiffa 1976) have been concave. However, if the decision maker is risk prone toward the objective, the single utility function is convex (Keeney and Raiffa 1979). Convex as well as quasiconvex single utility functions have been found to represent the decision
maker's preference in some cases (e.g., Duckstein, Bobee and Ashkar 1990; Anandam and Osolon 1990). Using \( p = 1 \) in the compromise programming model of Duckstein et al. (1990), \( U(c_1, c_2, c_5; \text{given } c_3, c_4) \) is strictly quasiconvex. In the case of the multiplicative model, its properties depends on the value of the scaling constant \( w \). From the decision maker's point of view, a negative value of \( w \) implies that he/she has a multiattribute risk-averse attitude, where a positive value of \( w \) implies that he/she has a multiattribute risk-seeking attitude. Harrison and Rosenthal (1988) analyzed the multiplicative model assuming a strictly concave and increasing single attribute utility/value function \( u_i(f_i) \). It was shown that the composite function \( U(f(x)) \) is strictly quasiconcave on the feasible set \( X \) if the scaling constant \( w \) is positive (i.e. risk seeking). However, they found that the function \( U(f(x)) \) can often behave as a strictly quasiconcave function when the value of \( w \) is negative, but it can also be strictly quasiconvex. In the next subsection, we analyze both the additive and the multiplicative models assuming strictly concave as well as strictly convex and increasing single utility functions for the two value cases of the scaling constant \( w \). The purpose of the analyses for these models is to generalize Harrison and Rosenthal's results and to show sufficient conditions for their multiattribute utility/value functions to be either strictly quasiconcave or strictly quasiconvex. We first demonstrate certain properties of the multiattribute additive and multiplicative utility/value functions, then summarize these results with an illustrative example.
Lemma 3.1

Let \( G(x) = \sum_{i=1}^{k} g_i(x) \). If each \( g_i(x) \) is monotone increasing in \( x \), then \( G(x) \) is also monotone increasing in \( x \).

Proof

Let \( x, y \in \mathbb{R}^n \) such that \( x \geq y \). Since each \( g_i(.) \) is monotone increasing, then \( g_i(x) > g_i(y) \). This implies that \( \sum_{i=1}^{k} w_i g_i(x) > \sum_{i=1}^{k} w_i g_i(y) \). Therefore, \( G(x) > G(y) \) and \( G(.) \) is monotone increasing, Q.E.D.

Lemma 3.2

Let \( G(x) = \sum_{i=1}^{k} g_i(x) \). If each \( g_i(x) \) is strictly quasiconcave and increasing, then \( G(x) \) is also strictly quasiconcave.

Proof

Let \( x, y \in X \) such that \( G(x) > G(y) \). Since each \( g_i(.) \) is increasing, then \( G(.) \) is also increasing by Lemma 3.1. Therefore \( G(x) > G(y) \) implies that \( x \geq y \). Further, since each \( g_i(.) \) is increasing then \( g_i(x) > g_i(y) \). Now, Since \( g_i(.) \) is strictly quasiconcave, then \( g_i(\alpha x + (1-\alpha)y) > g_i(y) \) for each \( i \). Then by taking the sum of both sides, we have \( \sum_{i=1}^{k} g_i(\alpha x + (1-\alpha)y) > \sum_{i=1}^{k} g_i(y) \). This implies that \( G(\alpha x + (1-\alpha)y) > G(y) \). Therefore, by definition, \( G(.) \) is strictly quasiconcave, Q.E.D.

Theorem 3.1

Consider the additive utility model where \( U'(x) = U(f(x)) = \sum_{i=1}^{k} w_i u_i(f_i(x)) \), \( \sum_{i=1}^{k} w_i = 1 \), \( 0 < w_i < 1 \) defined on a compact convex set \( X \). If \( u_i(f_i(x)) \) is a strictly concave and increasing for all \( i \), then \( U'(x) \) is strictly quasiconcave.
Proof

Let $g_i(x) = w_i u_i(f_i(x))$ for each $i$. Since $u_i(f_i(x))$ is a strictly concave and increasing and $w_i > 0$ for each $i$, then $g_i(x) = w_i u_i(f_i(x))$ is also strictly quasiconcave and increasing on $X$. Therefore, $U'(x)$ is strictly quasiconcave by lemma 3.2, Q.E.D.

Lemma 3.3

Let $G(x)$ be defined as $G'(x) = T(G(x))$, where $G(x)$ is a function on a compact convex set $X$ and $T(.)$ is a monotone increasing function. If $G(x)$ is strictly quasiconcave, then $G'(x)$ is also strictly quasiconcave.

Proof

Let $x, y \in X$ such that $G'(x) > G'(y)$. Since $T(.)$ is monotone increasing in its argument, then $G(x) > G(y)$. Given that $G(.)$ is strictly quasiconcave, $G(\alpha x + (1-\alpha)y) > G(y)$. Therefore, $T(G(\alpha x + (1-\alpha)y)) > T(G(y))$. Hence $G'(\alpha x + (1-\alpha)y) > G'(y)$. Therefore, by definition, $G'(.)$ is strictly quasiconcave, Q.E.D.

Theorem 3.2

Consider the multiplicative utility model where $U'(x) = U(f(x)) = (1/w)\{\prod_{i=1}^{k} [1+ww_i u_i(f_i(x))]-1\}$, $w>0$, $0<w_i<1$ defined on a compact convex set $X$. If $u_i(f_i(x))$ is strictly concave and increasing, then $U'(x)$ is strictly quasiconcave.

Proof

Let $G(x) = \Sigma_{i=1}^{k} \ln[1+ww_i u_i(f_i(x))]$, $ww_i>0$. Now $[1+ww_i u_i(f_i(x))]$ is strictly concave and increasing since $u_i(f_i(x))$ is strictly concave
and increasing and \( w_i > 0 \) for all \( i \). This implies that \([1 + w_i u_i(f_i(x))]\) is strictly quasiconcave and increasing for each \( i \). Now, since \( \ln(.) \) is an increasing transformation, then \( \ln[1 + w_i u_i(f_i(x))] \) is also strictly quasiconcave and increasing for all \( i \) by lemma 3.3. Furthermore, \( G(x) = \sum_{i=1}^{k} \ln[1 + w_i u_i(f_i(x))] \) is strictly quasiconcave by lemma 3.2. Now, using the increasing transformation \( \exp(.) \), \( U'(x) \) can be written as follows, \( U'(x) = (1/w) \exp(G(x)) \), and therefore \( U'(x) \) is strictly quasiconcave by lemma 3.3, Q.E.D.

**Lemma 3.4**

Let \( G(x) = \sum_{i=1}^{k} g_i(x) \). If each \( g_i(x) \) is strictly quasiconcave and decreasing, then \( G(x) \) is also strictly quasiconcave.

**Proof**

Let \( x, y \in X \) such that \( G(x) > G(y) \). Since each \( g_i(.) \) is decreasing, then \( G(.) \) is also decreasing, similar to Lemma 3.1. Therefore \( G(x) > G(y) \) implies that \( x \leq y \). Further, since each \( g_i(.) \) is decreasing then \( g_i(x) > g_i(y) \). Now, Since \( g_i(.) \) is strictly quasiconcave, then \( g_i(\alpha x + (1-\alpha)y) > g_i(y) \) for each \( i \). Then by taking the sum of both sides, we have \( \sum_{i=1}^{k} g_i(\alpha x + (1-\alpha)y) > \sum_{i=1}^{k} g_i(y) \). This implies that \( G(\alpha x + (1-\alpha)y) > G(y) \). Therefore, by definition, \( G(.) \) is strictly quasiconcave, Q.E.D.

**Lemma 3.5**

Let \( G(x) = \sum_{i=1}^{k} g_i(x) \). If each \( g_i(x) \) is strictly quasiconvex and increasing, then \( G(x) \) is also strictly quasiconvex.
Proof

Let \( x, y \in X \) such that \( G(x) > G(y) \). Since each \( g_i(.) \) is increasing, then \( G(.) \) is also increasing by Lemma 3.1. Therefore \( G(x) > G(y) \) implies that \( x \geq y \). Further, since each \( g_i(.) \) is increasing then \( g_i(x) > g_i(y) \). Now, Since \( g_i(.) \) is strictly quasiconvex, then \( g_i(\alpha x + (1-\alpha)y) < g_i(x) \) for each \( i \). Then by taking the sum of both sides, we have \( \sum_{i=1}^{k} g_i(\alpha x + (1-\alpha)y) < \sum_{i=1}^{k} g_i(x) \). This implies that \( G(\alpha x + (1-\alpha)y) < G(x) \). Therefore, by definition, \( G(.) \) is strictly quasiconvex, Q.E.D.

Theorem 3.3

Consider the additive utility model where \( U'(x) = U(f(x)) = \sum_{i=1}^{k} w_i u_i(f_i(x)), \sum_{i=1}^{k} w_i = 1, 0 < w_i < 1 \) defined on a compact convex set \( X \). If \( u_i(f_i(x)) \) is strictly convex and increasing, then \( U'(x) \) is strictly quasiconvex.

Proof

Let \( g_i(x) = w_i u_i(f_i(x)) \) for each \( i \). Since \( u_i(f_i(x)) \) is a strictly convex and increasing and \( w_i > 0 \) for each \( i \), then \( g_i(x) = w_i u_i(f_i(x)) \) is also strictly quasiconvex and increasing on \( X \). Therefore, \( U'(x) \) is strictly quasiconvex by lemma 3.5, Q.E.D.

Theorem 3.4

Consider the multiplicative utility model where \( U'(x) = U(f(x)) = (1/w)[\prod_{i=1}^{k}[1+ww_i u_i(f_i(x))] - 1], w < 0, 0 < w_i < 1 \) defined on a compact convex set \( X \). If \( u_i(f_i(x)) \) is strictly convex and increasing, then \( U'(x) \) is strictly quasiconvex.
Proof:
Since $u_i(f_i(x))$ is strictly convex then it is also strictly quasiconvex. This implies that $[1+ww_iu_i(f_i(x))]$ is strictly quasiconcave and decreasing since $ww_i<0$ for all $i$. Since $\ln(\cdot)$ is an increasing transformation, then $\ln[1+ww_iu_i(f_i(x))]$ is also strictly quasiconcave for all $i$ by lemma 3.3 and decreasing. Now, Let $G(x) = \sum_{i=1}^{k} \ln[1+ww_iu_i(f_i(x))]$. $G(x)$ is strictly quasiconcave by lemma 3.4. Now, using the increasing transformation $\exp(\cdot)$, $U'(x)$ can be written as follows, $U'(x) = (1/w) \exp(G(x))$, and therefore $U'(x)$ is strictly quasiconvex by lemma 3.3 and because $w < 0$, Q.E.D.

Finally, the above results for the additive and multiplicative utility/value models can be summarized as follows:

<table>
<thead>
<tr>
<th>$w &gt; 0$</th>
<th>strictly concave $u_i(f_i(x))$</th>
<th>strictly convex $u_i(f_i(x))$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>strictly quasiconcave</td>
<td>indeterminate</td>
</tr>
<tr>
<td>$w = 0$</td>
<td>strictly quasiconcave</td>
<td>strictly quasiconvex</td>
</tr>
<tr>
<td>$w &lt; 0$</td>
<td>indeterminate</td>
<td>strictly quasiconvex</td>
</tr>
</tbody>
</table>

where, the indeterminate case implies that the multiattribute utility/value function might be either quasiconcave or quasiconvex.
depending on the value of the scalar constant \( w \) and the degree of concavity/convexity of the single attribute utility/value functions. The above two indeterminate cases of positive and negative value of \( w \) correspond to the two cases of consistently risk seeking and consistently risk averse respectively. Thus, for many, if not most cases, it will not be clear which class a particular \( U(.) \) belongs to. Consequently, the need for providing algorithms for handling quasiconvex utility functions such as our algorithms is established.

To illustrate the indeterminate case, example 3.1 is provided next.

**Example 3.1**

Consider the multiplicative multiattribute model and let the single attribute utility/value function \( u_i(f_i) \) be

\[
  u_i(f_i) = -0.785[1 - \exp(0.82148f_i)], \quad i = 1, 2;
\]

which is a strictly convex and increasing. Furthermore, consider the following three alternatives, where the first is a convex combination of the other two:

\[
  f^1 = (0.5 \quad 0.5), \quad f^2 = (0.4 \quad 0.6) \text{ and } f^3 = (0.6 \quad 0.4).
\]

Now, we consider two cases with respect to the values of the scalar constants.

**Case 1**

Let \( w_1 = w_2 = 0.4 \) and therefore \( w = 1.25 \). Using the multiplicative multiattribute model, we have:

\[
  U(f^1) = 0.350784, \quad U(f^2) = 0.352726, \text{ and } U(f^3) = 0.352726.
\]
This implies that $U(f^1) < \max \{U(f^2), U(f^3)\}$ and therefore $U(f)$ is quasiconvex by definition with respect to the set \{f^1, f^2, f^3\}.

Case 2

Let $w_1 = w_2 = 0.2$ and therefore $w = 15$. Using the multiplicative multiattribute model, we have:

$U(f^1) = 0.254886$, $U(f^2) = 0.252720$, and $U(f^3) = 0.252720$.

This implies that $U(f^1) > \min \{U(f^2), U(f^3)\}$ and therefore $U(f)$ is quasiconcave by definition with respect to the set \{f^1, f^2, f^3\}.

3.4.1 Utility Class Testing Theory

The concept of testing for the form of the utility function (either linear or quasiconcave) was suggested by Korhonen, Moskowitz and Wallenius (KMW) (1986). They provided tests for detecting linearity as well as quasiconcavity of the utility function with respect to a pairwise preference information of a representative set of nondominated alternatives. Furthermore, Malakooti (1989) suggests testing for the inconsistency of the decision maker toward the assumption of the quasiconcave utility function. This testing is imbedded in a method designed to solve discrete multiple criteria decision problems. Also, Malakooti (1990) suggested a testing procedure for quasiconcave and quasiconvex utility functions based on the definitions of such functions. In this subsection, two testing procedures for both quasiconcave and quasiconvex utility functions are presented. The first testing procedure is an extension of the KMW's concept, where the second one is based on Malakooti
(1989, 1990). These procedures include mathematical programs that provide a diagnostic necessary tests for the two classes of utility functions. Furthermore, in contrast to the KMW's test indifferent responses can be utilized as well.

Let $Z$ be a subset of nondominated alternatives generated from the nondominated frontier set $F$. Also let set $P$ represent the pairwise preference data constructed based on the DM's responses to pairwise questions, where $P \subseteq Z \times Z$ and the set $P$ is defined as:

$$P = \{<f_i, f_j> | f_i, f_j \in Z, f_i \text{ is preferred or indifferent to } f_j, i \neq j, i,j = 1, ... , z \}.$$  

From the above set $P$, we identify all subsets that have a common least-preferred alternative as well as common most-preferred one. These subsets can be defined as:

$$PL_i = \{<f_i, f_j> | f_i \text{ is preferred or indifferent to } f_j, i \neq j, \text{ for all } i \} \quad j = 1, ... , z.$$  

$$PM^i = \{<f_i, f_j> | f_i \text{ is preferred or indifferent to } f_j, i \neq j, \text{ for all } j \} \quad i = 1, ... , z.$$  

Next, we introduce definition 3.1. It provides a condition in order for a pairwise-comparison preference set $P$ to be called consistent with a certain class of utility functions. Based on this definition as well as properties of the quasiconcave utility function, two mathematical programming-based tests (program 3.5 and problem 3.1) are derived. Program 3.5 provides the condition through the optimal value of its objective function. However, problem 3.1 provides it through its set of constraints. It is clear that testing using problem 3.1 requires less computational effort than testing using program 3.5. Those tests provide a necessary condition
for the set $P$ to be consistent with the class of quasiconcave utility functions. This condition is also sufficient for the set $P$ to be inconsistent with the class of quasiconcave utility functions.

**Definition 3.1**

Let $U^*$ be a class of utility functions. A pairwise-comparison preference set $P$ is $U^*$-consistent if and only if there exists a utility function $U \in U^*$ such that $U(f^i) \geq U(f^j)$ for all $<f^i, f^j> \in P, i \neq j$.

**Theorem (Bazaraa and Shetty 1979)**

Let $V$ be a nonempty open convex set in $\mathbb{R}^k$, and let $U:V\rightarrow \mathbb{R}$ be differentiable on $V$. Then $U$ is quasiconcave if and only if either one of the following equivalent statements hold,

1. If $f^i, f^j \in V$ and $U(f^i) \geq U(f^j)$, then $\nabla U(f^i)(f^j-f^i) \geq 0$.
2. If $f^i, f^j \in V$ and $\nabla U(f^i)(f^i-f^j) < 0$, then $U(f^i) < U(f^j)$.

**Lemma 3.6**

Let $U(.)$ be a quasiconcave function on a convex set $V \subset \mathbb{R}^k$. Let $f^i \in V, i = 1, \ldots, m$. If $U(f^i) \geq U(f^j)$ for all $i$ and some $j, i \neq j$, then there exists a set of multipliers $\lambda = (\lambda_1, \ldots, \lambda_k), \lambda \geq 0, \sum_{r=1}^{k} \lambda_r = 1$ such that, for all $i, \lambda^i(f^i-f^j) \geq 0, i = 1, \ldots, m, i \neq j$.

**Proof (by construction)**

Since $U(.)$ is quasiconcave function on $V \subset \mathbb{R}^n$ and $f^i \in V, U(f^i) \geq U(f^j)$ implies that $\nabla U(f^j)(f^i-f^j) \geq 0$. Set $\lambda = \nabla U(f^j)/[\sum_{r=1}^{k} \partial U(f^j)/\partial f_r], \lambda \geq 0$ and $\sum_{r=1}^{k} \lambda_r = 1$. Then $\lambda^i(f^i-f^j) \geq 0, i = 1, \ldots, m, i \neq j$, Q.E.D.
Theorem 3.7

Let $U_{q}v$ be the class of quasiconcave utility functions and $U(.) \in U_{q}v$. Given a pairwise-comparison preference set $P$, if $P$ is $U_{q}v$-consistent, then $\epsilon^* \geq 0$ in the following program,

Program 3.5: $\epsilon^* = \text{Maximize } \epsilon$

s.t. $(\lambda^j)^{t}(f^i - f^j) \geq \epsilon$ for all $<f^i,f^j> \in P$, $i, j=1,...,z, i \neq j$.

$\lambda^j \geq 0, \Sigma_{r=1}^{k} \lambda^j_r=1, \text{ for all } PL^j \neq \emptyset, j=1,...,z$.

Proof (by contradiction)

Let $\epsilon^* < 0$ in the above program. Since the set $P$ is $U_{q}v$-consistent then there exists $U \in U_{q}v$ such that $U(f^i) \geq U(f^j)$ for all $<f^i,f^j> \in P$. From the definition of the subsets $PL^j$ and by Lemma 3.6, there exists a set $\lambda^j = (\lambda^j_1, ..., \lambda^j_k), \lambda^j \geq 0$ such that $(\lambda^j)^{t}(f^i-f^j) \geq 0$ for all $<f^i,f^j> \in PL^j, PL^j \neq \emptyset$, for all $j, j=1,...,z, i \neq j$.

Now in the above program, $\epsilon^* = \text{minimum}_{j=1,...,z}((\lambda^j)^{t}(f^i-f^j))$. This implies that there exist a $j$ such that $(\lambda^j)^{t}(f^i-f^j) = \epsilon^* < 0$, which contradicts (1). Therefore $\epsilon^* \geq 0$, Q.E.D.

Theorem 3.8

Let $U_{q}v$ be the class of quasiconcave utility functions and $U(.) \in U_{q}v$. Given a pairwise-comparison preference set $P$, if $P$ is $U_{q}v$-consistent, then the following problem has a basic feasible solution.

Problem 3.1 $(\lambda^j)^{t}(f^i - f^j) \geq 0$ for all $<f^i,f^j> \in P$, $i, j=1,...,z, i \neq j$.

$\lambda^j \geq 0, \Sigma_{r=1}^{k} \lambda^j_r=1, \text{ for all } PL^j \neq \emptyset, j=1,...,z$. 
Proof (by contradiction)

Assume the above problem does not have a basic feasible solution and therefore it is inconsistent. This implies that there does not exist a set of \( \lambda^j \) vectors with \( \lambda^j \geq 0, \Sigma_{r=1}^{k} \lambda^j_r = 1 \), for some \( j, j = 1, \ldots, z, PL^j \neq \emptyset \); such that \( (\lambda^j)^t (f^j - f^i) \geq 0 \). Since the set \( P \) is \( U_{qv} \)-consistent and \( <f^i, f^i> \in P \), then there exist a quasiconcave function \( U(.) \) such that \( U(f^i) \geq U(f^j) \), by the definition of \( P \) being \( U_{qv} \)-consistent. Now, using lemma 3.6, there exists a set of multipliers \( \lambda \geq 0, \Sigma_{r=1}^{k} \lambda_r = 1 \), such that \( \lambda^t (f^i - f^j) \geq 0 \). Since this true for all \( PL^j \neq \emptyset, j = 1, \ldots, z \); a contradiction results. Therefore, the above problem has a basic feasible solution, Q.E.D.

Furthermore, based on this definition as well as properties of the quasiconvex utility function, two mathematical programming-based tests (program 3.6 and problem 3.2) are derived next. Program 3.6 provides the condition through the optimal value of its objective function. However, problem 3.2 provides it through its set of constraints. It is clear that testing using problem 3.2 requires less computational effort than testing using program 3.6. Those tests provide a necessary condition for the set \( P \) to be consistent with the class of quasiconvex utility functions. This condition is also sufficient for the set \( P \) to be inconsistent with the class of quasiconvex utility functions.
Theorem [Bazaraa and Shetty 1979]

Let $V$ be a nonempty open convex set in $R^k$, and let $U:V \rightarrow R$ be
differentiable on $V$. Then $U$ is quasiconvex if and only if either one of
the following equivalent statements hold,

1. If $f^i, f^j \in V$ and $U(f^i) \leq U(f^j)$, then $\nabla U(f^i)(f^i - f^j) \leq 0$.
2. If $f^i, f^j \in V$ and $\nabla U(f^i)(f^i - f^j) > 0$, then $U(f^i) > U(f^j)$.

Lemma 3.7

Let $U(.)$ be a quasiconvex function on a convex set $V \subset R^k$. Let
$f^j \in V$, $j = 1, \ldots, m$. If $U(f^i) \geq U(f^j)$ for all $j$ and some $i$, $i \neq j$, then there
exists a set of multipliers $\lambda = (\lambda_1, \ldots, \lambda_k)$, $\lambda \geq 0$, $\sum_{r=1}^{k} \lambda_r = 1$ such that,
for all $j$, $\lambda^j (f^i - f^j) \geq 0$, $j = 1, \ldots, m$, $i \neq j$, Q.E.D.

Proof (by construction)

Since $U(.)$ is quasiconvex function on $V \subset R^n$ and $f^j \in V$, $U(f^i) \geq
U(f^j)$ implies that $\nabla U(f^i)(f^i - f^j) \geq 0$. Set $\lambda = \nabla U(f^i)/\sum_{r=1}^{k} \partial U(f^i)/\partial f_r$, where $\lambda \geq 0$ and $\sum_{r=1}^{k} \lambda_r = 1$. Then $\lambda^j (f^i - f^j) \geq 0$, $j = 1, \ldots, m$, $i \neq j$.

Theorem 3.9

Let $U_{qx}$ be the class of quasiconvex utility functions and
$U(.) \in U_{qx}$. Given a pairwise-comparison preference set $P$, if $P$ is $U_{qx}$-
consistent, then $\varepsilon^* \geq 0$ in the following program,

Program 3.6: $\varepsilon^* = \text{Maximize } \varepsilon$

\begin{align*}
\text{s.t. } (\lambda^j)(f^i - f^j) & \geq \varepsilon \quad \text{for all } \langle f^i, f^j \rangle \in P, i, j = 1, \ldots, z, i \neq j. \\
\lambda^i & \geq 0, \quad \sum_{r=1}^{k} \lambda^i_r = 1, \quad \text{for all } \text{PM}^i = \emptyset, i = 1, \ldots, z.
\end{align*}
Proof (by contradiction)

Let $\varepsilon^* < 0$ in the above program. Since the set $P$ is $U_{qX}$-consistent then there exists an $U \in U_{qX}$ such that $U(f^i) \geq U(f^j)$ for all $<f^i, f^j> \in P$. From the definition of the subsets $PM^i$ and by Lemma 3.7, there exists a set $\lambda^i = (\lambda^i_1, \ldots, \lambda^i_k)$, $\lambda^i \geq 0$ such that $(\lambda^i)^T(f^i - f^j) \geq 0$ for all $<f^i, f^j> \in PM^i$, $PM^i = \emptyset$, for all $i, i = 1, \ldots, z, i \neq j$.

(2)

Now in the above program, $\varepsilon^* = \min \{ \lambda^i : (\lambda^i)^T(f^i - f^j) \}$, This implies that there exists an $i$ such that $(\lambda^i)^T(f^i - f^j) = \varepsilon^* < 0$, which contradicts (2). Therefore $\varepsilon^* \geq 0$, Q.E.D.

Theorem 3.10

Let $U_{qX}$ be the class of quasiconvex utility functions and $U(.) \in U_{qX}$. Given a pairwise-comparison preference set $P$, if $P$ is $U_{qX}$-consistent, then the following problem has a basic feasible solution.

Problem 3.2 $(\lambda^i)^T(f^i - f^j) \geq 0$ for all $<f^i, f^j> \in P$, $i, j = 1, \ldots, z, i \neq j$.

$\lambda^i \geq 0$, $\sum_{r=1}^{k} \lambda^i_r = 1$, for all $PM^i \neq \emptyset$, $i = 1, \ldots, z$.

Proof (by contradiction)

Assume the above problem does not have a basic feasible solution and therefore its set is inconsistent. This implies that there does not exist a set of $\lambda^i$ vectors with $\lambda^i \geq 0$, $\sum_{r=1}^{k} \lambda^i_r = 1$, for some $i, i = 1, \ldots, z, PM^i \neq \emptyset$; such that $(\lambda^i)^T(f^i - f^j) \geq 0$. Since the set $P$ is $U_{qX}$-consistent and $<f^i, f^j> \in P$, then there exist a quasiconvex function $U(.)$ such that $U(f^i) \geq U(f^j)$, by the definition of $P$ being $U_{qX}$-consistent. Now, using lemma 3.7, there exists a set of multipliers $\lambda \geq 0$, $\sum_{r=1}^{k} \lambda_r = 1$, such that $\lambda^T(f^i - f^j) \geq 0$. Since this true for all $PM^i \neq \emptyset$, $i = 1, \ldots, z$; a
contradiction results. Therefore, the above problem has a basic feasible solution, Q.E.D.

Utility class testing is executed by generating a representative set of nondominated alternatives (set Z) from the set of nondominated alternatives. This can be accomplished using a weighted Tchebycheff-based generating scheme via the LAMBDA and the FILTER sampling routines of Steuer (1986) or a goal point-based sampling scheme (Malakooti 1987). Since alternatives generated by the two methods suggested above (i.e. the weighted Tchebycheff-based and the goal point-based sampling schemes) are nondominated, more alternatives can be added to them. This can be accomplished by generating a convex combinations of some of those alternatives. Although, these new alternatives might be either dominated or infeasible, generating convex combinations has the potential of enhancing the discriminating power of the utility class tests greatly.

Then, the pairwise comparison preference set P is constructed. This is accomplished by interacting with the decision maker. The next step is to construct the different groupings PL^j and PM^i using their common least-preferred and common most-preferred alternative respectively. Finally, testing procedures (program 3.5 or problem 3.1 for the quasiconvex utility class and program 3.6 or problem 3.2 for the quasiconvex utility class) are used to test for the inconsistency of the decision maker's responses with respect to the
two classes respectively. The two methods for constructing the representative set as well as an example of that will be provided in subsection 3.5.3.

3.5 The Integrated Multiobjective Optimization Approach

In this section, the assumptions and the steps of interactive integrated multiobjective approach are presented. They are provided in subsections 3.5.1 and 3.5.2 respectively. The approach considers either a quasiconcave or a quasiconvex preference structure and provides a default step in case the underlying preference is inconsistent with both of these classes. Therefore, the strategy is to determine which type of utility function class would be consistent with the decision maker preference first and then to branch to the appropriate multiobjective algorithm. The imbedded algorithms will be designed to handle the general multiobjective optimization problem in which the feasible region of the objective space might be nonconvex.

3.5.1 Assumptions

The developed methodology will be based on the following assumptions.

For the Decision maker

(1) The underlying utility function \( U(f): \mathbb{R}^k \rightarrow \mathbb{R} \) is only implicitly known to the decision maker, and is assumed to be either a nondecreasing continuous and differentiable \textsc{Quasiconcave} or
Quasiconvex utility function on a convex set \( V \) that contains the objective feasible set \( F \).

(2) The decision maker would be able to provide consistent preference information, based on his/her implicit utility function, such as preference comparison with respect to a set of nondominated alternatives, paired comparison between two nondominated alternatives together with their associated strength of preference and simple nondominated tradeoff questions.

For the model structure

(1) The objectives are continuous and differentiable functions of the decision variables.

(2) The constraint set is compact (i.e., closed and bounded) with continuous and differentiable function constraints.

(3) Decision variables assume real values.

3.5.2 The Steps of the Approach

In this subsection, the main steps of the integrated approach are presented. The approach includes four main steps. The first step consists of two subtasks. The first subtask is to generate a representative sample of (\( > 2k \)) nondominated alternatives. This subtask can be accomplished using the LAMBDA and FILTER routines of Steuer (1986). The second subtask requires interaction with the decision maker to evaluate the sample of alternatives generated above. This interaction results in a complete pairwise preference
information set P. These two subtasks will be discussed in subsection 3.5.3.

The second main step of the approach checks whether the set P constructed above is consistent with the class of the quasiconcave utility functions or not. If it is so, the quasiconcave utility-based algorithm is used to solve the problem. If it is not, the third step of the approach is invoked. This step checks whether the set P is consistent with the class of the quasiconvex utility functions or not. If it is consistent, the quasiconvex utility-based algorithm that is appropriate to the structure of the multiobjective model is used to solve the problem. If the set is not consistent with the quasiconcave or quasiconvex utility function class, the fourth step of the approach is used. This step is a default which includes two subtasks. In the first subtask, the quasiconcave utility-based algorithm is executed several times, each with a different starting point, and the final alternatives are accumulated in a list. In the second subtask, all of the alternatives that were accumulated in the list are presented to the decision maker for evaluation. He/she identifies the most-preferred alternative for the whole problem. In this case, the algorithm is being used as a heuristic exploratory search process since it has the capability of considering all types of nondominated alternatives. Next, the main steps of the integrated approach are presented.
Step 1  **Initialization**

(i) Generate an representative sample of the nondominated frontier (e.g. 2k number of alternatives).

(ii) Construct the pairwise-comparison preference set.

Step 2  **Check for the Quasiconcave Utility Class**

(i) Test for the Quasiconcave utility class.

(ii) If the test is positive, execute the quasiconcave-based algorithm, else go to Step 3.

Step 3  **Check for the Quasiconvex Utility Class**

(i) Test for the Quasiconvex utility class.

(ii) If the test is positive execute the appropriate quasiconvex-based algorithm,

else go to Step 4.

Step 4  **Default Case (Neither Quasiconcave or Quasiconvex)**

(i) Execute the quasiconcave-based algorithm several times, with a different starting point each time, and accumulate their final alternatives in a list.

(ii) Present this list to the decision maker to select the final alternative.

3.6 **Implementation Issues**

In this subsection, implementation of the first main step of the approach is discussed, where the other three main steps will be discussed in the next chapters. The first main task of this step
requires scanning the nondominated frontier in order to generate a representative sample $Z$ (e.g. $2k$) of nondominated alternatives, where $k$ is the number of the objective functions. This task can be accomplished by one of two methods. They are described next.

The weighted Tchebycheff-based sampling method

This method is based on the weighted Tchebycheff-based generating scheme and Steuer's LAMBDA and FILTER routines (Steuer 1986). The LAMBDA routine is a FORTRAN program that provides a set discretization capability for generating representatives from the continuous set of weighting vectors,

$$
\Lambda = \{ \lambda \in \mathbb{R}^k \mid \lambda_i \in [l_i, u_i], \sum_{i=1}^{k} \lambda_i = 1 \}, \ 0 \leq l_i \leq 1, \ 0 \leq u_i \leq 1.
$$

Desired alternatives are generated using the augmented weighted Tchebycheff-based scheme, where each weighting vector $\lambda \in \Lambda$ yields one nondominated alternative. However the FILTER routine, which is also a FORTRAN program, plays a different role. It provides a forward as well as a reverse filtering capabilities. The forward filtering process obtains a subset (call it $S$) of alternatives from a bigger set (call it $V$) such that those alternatives are the most different from one another. This is accomplished by finding the $S$ vectors in $V$ that are furthest apart from one another according a given metric. However the reverse filtering process, which is not used in our case, obtains the $S$ vectors in $V$ that are the most similar to a vector $v^* \in V$. This is accomplished by finding the $S$ vectors closest to $v^*$ according to a given metric.
In order to construct the representative set \( Z \) using this method, several steps are performed. They are presented next.

Step (1) Calculate the ideal point \( f^* \) and let the utopian point \( f^{**} = f^* + \varepsilon \), where \( \varepsilon > 0 \).

Step (2) Randomly generate a large number (e.g. 50xk) of trial weighting vectors \( \lambda \) from \( \Lambda \),

where \( \Lambda = \{ \lambda \in \mathbb{R}^K | \lambda_i \in [0, 1], \sum_{i=1}^{K} \lambda_i = 1 \} \).

Step (3) Filter this set to obtain a subset of size 2xk of representative weighting vectors, using the FILTER routine.

Step (4) For each representative weighting vector \( \lambda \), solve the associated augmented weighted Tchebycheff program:

Program 3.7: \[
\begin{align*}
\text{minimize} & \quad \alpha + \rho \epsilon (f^{**} - f(x)) \\
\text{s.t.} & \quad \lambda_i(f_i^{**} - f_i(x)) \leq \alpha \\
& \quad x \in X,
\end{align*}
\]

where \( \rho \) is a sufficiently small positive values.

Step (5) Filter the 2xk resulting nondominated alternatives to obtain the subset of \( k \) most different alternatives using the FILTER routine.

Step (6) Accumulate those \( k \) alternatives in a set, call it \( Z \) and Stop.

The goal point-based sampling scheme

In order to construct the representative set \( Z \) using this method, several steps are performed. They are presented next.
Step (1) Calculate the ideal point \( f^* \) and the nadir point \( f_* \). Randomly, generate a large number (e.g. 50\( \times k \)) of trial alternatives \( f^t \), where \( f^* \leq f^t \leq f_* \).

Step (2) Randomly generate a large number (e.g. 50\( \times k \)) of trial weighting vectors \( \lambda \) from \( \Lambda \),

where \( \Lambda = \{ \lambda \in \mathbb{R}^k | \lambda_i \in [0, 1], \sum_{i=1}^k \lambda_i = 1 \} \).

Step (3) Filter the sets obtained in step (1) and Step (2) to obtain a subset of size 2\( \times k \) of representative goal points and weighting vectors respectively, using the FILTER routine.

Step (4) For each representative goal point \( f^t \) and weighting vector \( \lambda \), solve program:

Program 3.7: \[
\text{minimize } M \lambda^t y^+ - \lambda^t y^- \\
\text{s.t. } \]

\[
f_i(x) - y^- + y^+ = f^t_i \quad 1 \leq i \leq k, \]

\[
x \in X, \]

\[
y^-_i, y^+_i \geq 0, \quad 1 \leq i \leq k. \]

where \( f^t \) is a given goal point, \( M > 1.0 \) (e.g. \( M = 1.1 \)), and \( \lambda \geq 0 \), \( \sum_{i=1}^k \lambda_i = 1 \) is a given weight vector.

Step (5) Filter the resulting nondominated alternatives to obtain the subset of \( k \) most different alternatives using the FILTER routine.

Step (6) Accumulate those \( k \) alternatives in a set, call it \( Z \) and stop.

All of alternatives in \( Z \) are nondominated. Extra alternatives can be added to the set by considering some convex combinations of
pairs that have their points the furthermost apart. Although these alternatives might enhance the discriminating power of the test greatly, they might be dominated or infeasible. These type of alternatives will be called dummy alternatives.

The second main subtask of the first step is to construct the pairwise-comparison preference set P. This is accomplished through interacting with the decision maker to evaluate the alternatives of P, where he/she responds to a direct paired-comparison preference questions or goes through an interactive ranking process. In the case of using an interactive ranking process, ranked alternatives are transformed into a paird-comparison set of alternatives in order to construct the set P, assuming the decision maker's preference is transitive. As far as the sample Z is concerned, the decision maker might evaluate dummy alternatives (infeasible or dominated) that are in Z if some of its members are generated by a convex combination. This is a price to pay if the utility class tests need to have more discriminating power.

3.7 Example

In this example, the process of generating a representative sample of nondominated alternatives using the method of the weighted Tchebycheff-based sampling is illustrated. Consider the following multiobjective linear programming problem (Steuer 1986, page 461),

$$\max \quad f_1(x) = -x_1 - x_3 + x_4 + 2x_5 - x_6 + x_7$$
\[
\begin{align*}
\text{max} & \quad f_2(x) = 5x_2 + 2x_6 + 4x_8 \\
\text{max} & \quad f_3(x) = 4x_2 + 6x_4 + 2x_5 + 3x_6 + 4x_7 - x_8 \\
\text{Subject to} & \\
8x_3 + 6x_4 + x_5 + 4x_6 - 3x_7 + 7x_8 & \leq 6 \\
-x_1 + 2x_4 + 4x_6 + 3x_8 & \leq 7 \\
3x_1 + 3x_3 + 5x_4 + 3x_6 + x_7 & \leq 7 \\
-2x_3 + 7x_4 - 2x_5 - 2x_7 + 6x_8 & \leq 10 \\
3x_1 + 4x_2 - 2x_3 + x_4 - x_7 - x_8 & \leq 10 \\
7x_2 + 2x_3 + 7x_5 + 4x_7 & \leq 5 \\
-2x_1 - x_2 + x_3 - 3x_7 + 4x_8 & \leq 6 \\
6x_1 - 3x_2 + 3x_4 + 6x_7 + 4x_8 & \leq 6 \\
x_i \geq 0 & \quad \text{for all } i = 1, \ldots, 8.
\end{align*}
\]

Three cases of the decision maker ranking of the representative alternatives are considered.

Solution

The first task is to generate the representative sample \(Z\). In order to perform this task, the steps already presented in subsection 3.5.3 are performed. Their execution is discussed next.

Step (1): The ideal point is determined by maximizing each objective function individually. This gives,

\[
f^* = (6.333, 7.000, 11.491).
\]

Let the Utopian point \(f^{**} = (7.00, 8.00, 12.00)\).

Step (2): Using LAMDA program, 150 (50x3) \(\lambda\)-vectors are generated from the weighting space:

\[
\Lambda = \{\lambda \in \mathbb{R}^3 \mid \lambda_i \in [0, 1], \sum_{i=1}^{3} \lambda_i = 1\}.
\]
Step (3): Using FILTER program, the 150 \( \lambda \)-vectors are reduced to 12 dispersed representatives. They are given in Table 3.1.

Table 3.1: 12 representatives of \( \lambda \)-vectors (Steuer 1986).

<table>
<thead>
<tr>
<th>#</th>
<th>( \lambda_1 )</th>
<th>( \lambda_2 )</th>
<th>( \lambda_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.2779</td>
<td>0.2756</td>
<td>0.4465</td>
</tr>
<tr>
<td>3</td>
<td>0.0162</td>
<td>0.2693</td>
<td>0.7145</td>
</tr>
<tr>
<td>8</td>
<td>0.3708</td>
<td>0.2693</td>
<td>0.1601</td>
</tr>
<tr>
<td>14</td>
<td>0.0139</td>
<td>0.0234</td>
<td>0.9627</td>
</tr>
<tr>
<td>20</td>
<td>0.9122</td>
<td>0.0000</td>
<td>0.0878</td>
</tr>
<tr>
<td>22</td>
<td>0.0169</td>
<td>0.9829</td>
<td>0.0002</td>
</tr>
<tr>
<td>23</td>
<td>0.5175</td>
<td>0.0437</td>
<td>0.4388</td>
</tr>
<tr>
<td>24</td>
<td>0.7149</td>
<td>0.1278</td>
<td>0.1573</td>
</tr>
<tr>
<td>27</td>
<td>0.0745</td>
<td>0.4643</td>
<td>0.4612</td>
</tr>
<tr>
<td>38</td>
<td>0.1672</td>
<td>0.6587</td>
<td>0.1741</td>
</tr>
<tr>
<td>43</td>
<td>0.2823</td>
<td>0.0319</td>
<td>0.6858</td>
</tr>
<tr>
<td>79</td>
<td>0.4684</td>
<td>0.2485</td>
<td>0.2831</td>
</tr>
</tbody>
</table>

Step (4): Using the LINDO software, the 12 associated augmented weighted Tchebycheff programs are solved to generate 12 nondominated alternatives. They are given in Table 3.2.

Step (5): Using FILTER program, the 12 nondominated alternatives generated above are reduced to the six most different alternatives.
and accumulate them in the set $S$ (Steuer 1986). They are given in Table 3.3.

Table 3.2: 12 nondominated alternatives (Steuer 1986).

<table>
<thead>
<tr>
<th>#</th>
<th>$f_1$</th>
<th>$f_2$</th>
<th>$f_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.910</td>
<td>4.884</td>
<td>10.077</td>
</tr>
<tr>
<td>3</td>
<td>3.115</td>
<td>4.577</td>
<td>10.710</td>
</tr>
<tr>
<td>8</td>
<td>3.762</td>
<td>5.441</td>
<td>8.477</td>
</tr>
<tr>
<td>14</td>
<td>4.600</td>
<td>2.127</td>
<td>11.491</td>
</tr>
<tr>
<td>20</td>
<td>6.305</td>
<td>0.579</td>
<td>4.781</td>
</tr>
<tr>
<td>22</td>
<td>0.000</td>
<td>7.000</td>
<td>2.000</td>
</tr>
<tr>
<td>23</td>
<td>5.069</td>
<td>1.609</td>
<td>9.723</td>
</tr>
<tr>
<td>24</td>
<td>5.801</td>
<td>1.290</td>
<td>6.549</td>
</tr>
<tr>
<td>27</td>
<td>1.272</td>
<td>5.617</td>
<td>9.601</td>
</tr>
<tr>
<td>38</td>
<td>0.677</td>
<td>6.395</td>
<td>5.928</td>
</tr>
<tr>
<td>43</td>
<td>4.733</td>
<td>1.889</td>
<td>11.067</td>
</tr>
<tr>
<td>79</td>
<td>4.741</td>
<td>3.741</td>
<td>9.445</td>
</tr>
</tbody>
</table>

Step (6) Using the euclidian distance, rank all pairs of $S$ according to the distance between the two points of each pair in descending order. The highest two pairs are given in Table 3.4.

Step (7) Select the first $(k-1)$ pairs. For each one, generate one convex combination of its members.
Table 3.3: Set S of 6 most different alternatives (Steuer 1986).

<table>
<thead>
<tr>
<th>#</th>
<th>$f_1$</th>
<th>$f_2$</th>
<th>$f_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.910</td>
<td>4.884</td>
<td>10.077</td>
</tr>
<tr>
<td>14</td>
<td>4.600</td>
<td>2.127</td>
<td>11.491</td>
</tr>
<tr>
<td>22</td>
<td>0.000</td>
<td>7.000</td>
<td>2.000</td>
</tr>
<tr>
<td>24</td>
<td>5.801</td>
<td>1.290</td>
<td>6.549</td>
</tr>
<tr>
<td>27</td>
<td>1.272</td>
<td>5.617</td>
<td>9.601</td>
</tr>
<tr>
<td>38</td>
<td>0.677</td>
<td>6.395</td>
<td>5.928</td>
</tr>
</tbody>
</table>

Table 3.4: The two pairs of the highest distance.

<table>
<thead>
<tr>
<th>pair</th>
<th>distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>14 and 22</td>
<td>11.618</td>
</tr>
<tr>
<td>22 and 24</td>
<td>9.325</td>
</tr>
</tbody>
</table>

Step (8) For illustration, construct the set $Z$ by adding alternatives generated in Step (7) to the two pairs of Step (6). $Z$ is given in Table 3.5.

Table 3.5: The representative set $Z$.

<table>
<thead>
<tr>
<th>#</th>
<th>$f_1$</th>
<th>$f_2$</th>
<th>$f_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>4.600</td>
<td>2.127</td>
<td>11.491</td>
</tr>
<tr>
<td>22</td>
<td>0.000</td>
<td>7.000</td>
<td>2.000</td>
</tr>
<tr>
<td>24</td>
<td>5.801</td>
<td>1.290</td>
<td>6.549</td>
</tr>
<tr>
<td>CON(14,22)</td>
<td>2.300</td>
<td>4.564</td>
<td>6.745</td>
</tr>
<tr>
<td>CON(22,24)</td>
<td>2.900</td>
<td>4.145</td>
<td>4.275</td>
</tr>
</tbody>
</table>
Next, the preference set $P$ is constructed for cases I, II and III.

**Case I:** Decision maker ranking of members of above set $Z$ is given in Table 3.6 next.

<table>
<thead>
<tr>
<th>#</th>
<th>$f_1$</th>
<th>$f_2$</th>
<th>$f_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f^1$</td>
<td>4.600</td>
<td>2.127</td>
<td>11.491</td>
</tr>
<tr>
<td>$f^2$</td>
<td>2.300</td>
<td>4.564</td>
<td>6.745</td>
</tr>
<tr>
<td>$f^3$</td>
<td>5.801</td>
<td>1.290</td>
<td>6.549</td>
</tr>
<tr>
<td>$f^4$</td>
<td>2.900</td>
<td>4.145</td>
<td>4.275</td>
</tr>
<tr>
<td>$f^5$</td>
<td>0.000</td>
<td>7.000</td>
<td>2.000</td>
</tr>
</tbody>
</table>

Therefore, the set $P$ is as follows,

$$
P = \{ <f^1, f^2>, <f^1, f^3>, <f^1, f^4>, <f^1, f^5>, 
         <f^2, f^3>, <f^2, f^4>, <f^2, f^5>, 
         <f^3, f^4>, <f^3, f^5>, 
         <f^4, f^5> \}
$$

The next step is to construct subsets $PL^j$ and $PM^i$, $i, j = 1, \ldots, 5, i \neq j$.

$PL^1 = \emptyset$.

$PL^2 = \{ <f^1, f^2> \}$.

$PL^3 = \{ <f^1, f^3>, <f^2, f^3> \}$.

$PL^4 = \{ <f^1, f^4>, <f^2, f^4>, <f^3, f^4> \}$.

$PL^5 = \{ <f^1, f^5>, <f^2, f^5>, <f^3, f^5>, <f^4, f^5> \}$.

and
\[ \text{PM}^1 = \{ \langle f^1, f^2 \rangle, \langle f^1, f^3 \rangle, \langle f^1, f^4 \rangle, \langle f^1, f^5 \rangle \}. \]
\[ \text{PM}^2 = \{ \langle f^2, f^3 \rangle, \langle f^2, f^4 \rangle, \langle f^2, f^5 \rangle \}. \]
\[ \text{PM}^3 = \{ \langle f^3, f^4 \rangle, \langle f^3, f^5 \rangle \}. \]
\[ \text{PM}^4 = \{ \langle f^4, f^5 \rangle \}. \]
\[ \text{PM}^5 = \emptyset. \]

To perform the test for the quasiconcave utility class, we construct problem 3.1 (Theorem 3.8, page 58) using the pairwise-comparison preference set \( P \) given above. The following program results:

\textbf{Problem 3.1:}

\[ (\lambda^2)^i (f^i - f^2) \geq 0 \quad i = 1. \]
\[ (\lambda^3)^i (f^i - f^3) \geq 0 \quad i = 1, 2. \]
\[ (\lambda^4)^i (f^i - f^4) \geq 0 \quad i = 1, 2, 3. \]
\[ (\lambda^5)^i (f^i - f^5) \geq 0 \quad i = 1, 2, 3, 4. \]
\[ \sum_{r=1}^{3} \lambda^i_r = 1, \quad \lambda^i_j \geq 0, \quad j = 2, 3, 4, 5. \]

Using the component values for the alternatives of the set \( Z \), the above program is rewritten as follows:

\textbf{Problem 3.1:}

\[ 2.300\lambda^2_1 - 2.437\lambda^2_2 + 4.746\lambda^2_3 \geq 0 \]
\[ -1.201\lambda^3_1 + 0.837\lambda^3_2 + 4.942\lambda^3_3 \geq 0 \]
\[ -3.501\lambda^3_1 + 3.274\lambda^3_2 + 0.196\lambda^3_3 \geq 0 \]
\[ 1.700\lambda^4_1 - 2.018\lambda^4_2 + 7.216\lambda^4_3 \geq 0 \]
\[ -0.600\lambda^4_1 + 0.419\lambda^4_2 + 2.470\lambda^4_3 \geq 0 \]
\[2.901\lambda_1^4 - 2.855\lambda_2^4 + 2.274\lambda_3^4 \geq 0\]
\[4.600\lambda_1^5 - 4.873\lambda_2^5 + 9.491\lambda_3^5 \geq 0\]
\[2.300\lambda_1^5 - 2.436\lambda_2^5 + 4.745\lambda_3^5 \geq 0\]
\[5.801\lambda_1^5 - 5.710\lambda_2^5 + 4.549\lambda_3^5 \geq 0\]
\[2.900\lambda_1^5 - 2.855\lambda_2^5 + 2.275\lambda_3^5 \geq 0\]
\[\lambda_1^2 + \lambda_2^2 + \lambda_3^2 = 1\]
\[\lambda_1^3 + \lambda_2^3 + \lambda_3^3 = 1\]
\[\lambda_1^4 + \lambda_2^4 + \lambda_3^4 = 1\]
\[\lambda_1^5 + \lambda_2^5 + \lambda_3^5 = 1\]
\[\lambda_j \geq 0, \quad j = 2, 3, 4, 5.\]

Using LINDO, the above problem was found to be consistent. Hence, the set \(Z\) is declared not to be inconsistent with the quasiconcave utility function class.

**Case II**: Decision maker ranking of members of above set \(Z\) is given in Table 3.7 next.

**Table 3.7: Ranking for members of \(Z\) (Case II).**

<table>
<thead>
<tr>
<th>#</th>
<th>(f_1)</th>
<th>(f_2)</th>
<th>(f_3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(f^1)</td>
<td>4.600</td>
<td>2.127</td>
<td>11.491</td>
</tr>
<tr>
<td>(f^2)</td>
<td>5.801</td>
<td>1.290</td>
<td>6.549</td>
</tr>
<tr>
<td>(f^3)</td>
<td>2.300</td>
<td>4.564</td>
<td>6.745</td>
</tr>
<tr>
<td>(f^4)</td>
<td>0.000</td>
<td>7.000</td>
<td>2.000</td>
</tr>
<tr>
<td>(f^5)</td>
<td>2.900</td>
<td>4.145</td>
<td>4.275</td>
</tr>
</tbody>
</table>
Therefore, the set \( P \) is as follows,

\[
P = \{ <f^1, f^2>, <f^1, f^3>, <f^1, f^4>, <f^1, f^5>, <f^2, f^3>, <f^2, f^4>, <f^2, f^5>, <f^3, f^4>, <f^3, f^5>, <f^4, f^5> \}
\]

The next step is to construct subsets \( PL^j \) and \( PM^i \), \( i, j = 1, \ldots, 5, i \neq j \).

\[
PL^1 = \emptyset.
\]

\[
PL^2 = \{ <f^1, f^2> \}.
\]

\[
PL^3 = \{ <f^1, f^3>, <f^2, f^3> \}.
\]

\[
PL^4 = \{ <f^1, f^4>, <f^2, f^4>, <f^3, f^4> \}.
\]

\[
PL^5 = \{ <f^1, f^5>, <f^2, f^5>, <f^3, f^5>, <f^4, f^5> \}.
\]

and

\[
PM^1 = \{ <f^1, f^2>, <f^1, f^3>, <f^1, f^4>, <f^1, f^5> \}.
\]

\[
PM^2 = \{ <f^2, f^3>, <f^2, f^4>, <f^2, f^5> \}.
\]

\[
PM^3 = \{ <f^3, f^4>, <f^3, f^5> \}.
\]

\[
PM^4 = \{ <f^4, f^5> \}
\]

\[
PM^5 = \emptyset.
\]

To perform the test for the quasiconcave utility class, we construct problem 3.1 (theorem 3.8, page 58) using the pairwise-comparison preference set \( P \) given above. The following program results:

**Problem 3.1:**

\[
(\lambda^2)^i (f^i \cdot f^2) \geq 0 \quad i = 1.
\]
\[(\lambda^3)^i(f_i - f^3) \geq 0 \quad i = 1, 2.\]
\[(\lambda^4)^i(f_i - f^4) \geq 0 \quad i = 1, 2, 3.\]
\[(\lambda^5)^i(f_i - f^5) \geq 0 \quad i = 1, 2, 3, 4.\]
\[\sum_{i=1}^{3} \lambda^i = 1, \quad \lambda^j \geq 0, \quad j = 2, 3, 4, 5.\]

Using the component values for the alternatives of the set \(Z\), the above program is rewritten as follows:

**Problem 3.1:**

\[
\begin{align*}
2.300\lambda^2_1 &- 2.437\lambda^2_2 + 4.746\lambda^2_3 \geq 0 \\
-1.201\lambda^3_1 + 0.837\lambda^3_2 + 4.942\lambda^3_3 \geq 0 \\
-3.501\lambda^3_1 + 3.274\lambda^3_2 + 0.196\lambda^3_3 \geq 0 \\
1.700\lambda^4_1 - 2.018\lambda^4_2 + 7.216\lambda^4_3 \geq 0 \\
-0.600\lambda^4_1 + 0.419\lambda^4_2 + 2.470\lambda^4_3 \geq 0 \\
2.901\lambda^4_1 - 2.855\lambda^4_2 + 2.274\lambda^4_3 \geq 0 \\
4.600\lambda^5_1 - 4.873\lambda^5_2 + 9.491\lambda^5_3 \geq 0 \\
2.300\lambda^5_1 - 2.436\lambda^5_2 + 4.745\lambda^5_3 \geq 0 \\
5.801\lambda^5_1 - 5.710\lambda^5_2 + 4.549\lambda^5_3 \geq 0 \\
2.900\lambda^5_1 - 2.855\lambda^5_2 + 2.275\lambda^5_3 \geq 0
\end{align*}
\]

\[
\begin{align*}
\lambda^2_1 + \lambda^2_2 + \lambda^2_3 &= 1 \\
\lambda^3_1 + \lambda^3_2 + \lambda^3_3 &= 1 \\
\lambda^4_1 + \lambda^4_2 + \lambda^4_3 &= 1 \\
\lambda^5_1 + \lambda^5_2 + \lambda^5_3 &= 1 \\
\lambda^j &\geq 0, \quad j = 2, 3, 4, 5.
\end{align*}
\]
Using LINDO, the above problem was found to be inconsistent. Hence, the set Z is declared to be inconsistent with the quasiconcave utility function class. Therefore, we check for the quasiconvex utility class.

To perform the test for the quasiconvex utility class, we construct problem 3.2 (theorem 3.10, page 61) using the pairwise-comparison preference set P given above. The following program results:

\[ \begin{align*}
(\lambda^1)(f^1 - f^j) &\geq 0 \quad j = 2, 3, 4, 5. \\
(\lambda^2)(f^2 - f^j) &\geq 0 \quad j = 3, 4, 5. \\
(\lambda^3)(f^3 - f^j) &\geq 0 \quad j = 4, 5. \\
(\lambda^4)(f^4 - f^j) &\geq 0 \quad j = 5. \\
\sum_{r=1}^{3} \lambda^i_r = 1, \quad \lambda^i \geq 0, \quad i = 1, 2, 3, 4.
\end{align*} \]

Using the component values for the alternatives of the set Z, the above program is rewritten as follows:

\[ \begin{align*}
-1.201\lambda^1_1 + 0.837\lambda^1_2 + 4.942\lambda^1_3 &\geq 0 \\
2.300\lambda^2_1 - 2.437\lambda^2_2 + 4.746\lambda^2_3 &\geq 0 \\
4.600\lambda^3_1 - 4.873\lambda^3_2 + 9.491\lambda^3_3 &\geq 0 \\
1.700\lambda^4_1 - 2.018\lambda^4_2 + 7.216\lambda^4_3 &\geq 0 \\
3.501\lambda^1_2 - 3.274\lambda^2_2 - 0.196\lambda^3_2 &\geq 0 \\
5.801\lambda^2_1 - 5.710\lambda^2_2 + 4.549\lambda^2_3 &\geq 0 \\
2.900\lambda^3_2 - 2.855\lambda^2_2 + 2.275\lambda^3_2 &\geq 0
\end{align*} \]
\[2.300\lambda^3_1 - 2.436\lambda^3_2 + 4.745\lambda^3_3 \geq 0\]
\[-0.600\lambda^3_1 + 0.419\lambda^3_2 + 2.470\lambda^3_3 \geq 0\]
\[-2.900\lambda^4_1 + 2.855\lambda^4_2 - 2.275\lambda^4_3 \geq 0\]
\[
\begin{align*}
\lambda^1_1 + \lambda^1_2 + \lambda^1_3 &= 1 \\
\lambda^2_1 + \lambda^2_2 + \lambda^2_3 &= 1 \\
\lambda^3_1 + \lambda^3_2 + \lambda^3_3 &= 1 \\
\lambda^4_1 + \lambda^4_2 + \lambda^4_3 &= 1
\end{align*}
\]
\[\lambda^i \geq 0, \quad i = 1, 2, 3, 4.\]

Using LINDO, the above problem was found to be consistent. Hence, the set \(Z\) is declared not to be inconsistent with the quasiconvex utility function class.

**Case III:** Decision maker ranking of members of above set \(Z\) is given in Table 3.8 next.

<table>
<thead>
<tr>
<th>#</th>
<th>(f_1)</th>
<th>(f_2)</th>
<th>(f_3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(f^1)</td>
<td>2.300</td>
<td>4.564</td>
<td>6.745</td>
</tr>
<tr>
<td>(f^2)</td>
<td>4.600</td>
<td>2.127</td>
<td>11.491</td>
</tr>
<tr>
<td>(f^3)</td>
<td>5.801</td>
<td>1.290</td>
<td>6.549</td>
</tr>
<tr>
<td>(f^4)</td>
<td>0.000</td>
<td>7.000</td>
<td>2.000</td>
</tr>
<tr>
<td>(f^5)</td>
<td>2.900</td>
<td>4.145</td>
<td>4.275</td>
</tr>
</tbody>
</table>

Therefore, the set \(P\) is as follows,
\[
P = \{<f^1, f^2>, <f^1, f^3>, <f^1, f^4>, <f^1, f^5>, \}
\]
To perform the test for the quasiconcave utility class, we construct problem 3.1 (theorem 3.8, page 58) using the pairwise-comparison preference set P given above. The following program results:

Problem 3.1:

\[
(\lambda^2)^t(f_i - f^2) \geq 0 \quad i = 1.
\]

\[
(\lambda^3)^t(f_i - f^3) \geq 0 \quad i = 1, 2.
\]

\[
(\lambda^4)^t(f_i - f^4) \geq 0 \quad i = 1, 2, 3.
\]

\[
(\lambda^5)^t(f_i - f^5) \geq 0 \quad i = 1, 2, 3, 4.
\]

\[
\sum_{r=1}^{3} \lambda^j_r = 1, \quad \lambda^j \geq 0, \quad j = 2, 3, 4, 5.
\]

Using the component values for the alternatives of the set Z, the above program is rewritten as follows:

Problem 3.1:

\[-2.300\lambda^2_1 + 2.437\lambda^2_2 - 4.746\lambda^2_3 \geq 0\]
\[-3.501\lambda^3_1 + 3.274\lambda^3_2 + 0.196\lambda^3_3 \geq 0\]
\[-1.201\lambda^3_1 + 0.837\lambda^3_2 + 4.942\lambda^3_3 \geq 0\]
\[2.300\lambda^4_1 - 2.436\lambda^4_2 + 4.745\lambda^4_3 \geq 0\]
\[4.600\lambda^4_1 - 4.873\lambda^4_2 + 9.491\lambda^4_3 \geq 0\]
\[5.801\lambda^4_1 - 5.710\lambda^4_2 + 4.549\lambda^4_3 \geq 0\]
\[-0.600\lambda^5_1 + 0.419\lambda^5_2 + 2.470\lambda^5_3 \geq 0\]
\[1.700\lambda_1^5 - 2.018\lambda_2^5 + 7.216\lambda_3^5 \geq 0\]
\[2.901\lambda_1^5 - 2.855\lambda_2^5 + 2.274\lambda_3^5 \geq 0\]
\[-2.900\lambda_1^5 + 2.855\lambda_2^5 - 2.275\lambda_3^5 \geq 0\]
\[\lambda_1^2 + \lambda_2^2 + \lambda_3^2 = 1\]
\[\lambda_1^3 + \lambda_2^3 + \lambda_3^3 = 1\]
\[\lambda_1^4 + \lambda_2^4 + \lambda_3^4 = 1\]
\[\lambda_1^5 + \lambda_2^5 + \lambda_3^5 = 1\]
\[\lambda^j \geq 0, \quad j = 2, 3, 4, 5.\]

Using LINDO, the above problem was found to be inconsistent. Hence, the set \(Z\) is declared to be inconsistent with the quasiconcave utility function class. Therefore, we check for the quasiconvex utility class.

To perform the test for the quasiconvex utility class, we construct problem 3.2 (theorem 3.10, page 61) using the pairwise-comparison preference set \(P\) given above. The following program results:

**Problem 3.2:**

\[(\lambda^1)^j(f^1 - f^j) \geq 0 \quad j = 2, 3, 4, 5.\]
\[(\lambda^2)^j(f^2 - f^j) \geq 0 \quad j = 3, 4, 5.\]
\[(\lambda^3)^j(f^3 - f^j) \geq 0 \quad j = 4, 5.\]
\[(\lambda^4)^j(f^4 - f^j) \geq 0 \quad j = 5.\]
\[\Sigma_{i=1}^{3} \lambda_i^1 = 1, \quad \lambda_i^i \geq 0, \quad i = 1, 2, 3, 4.\]

Using the component values for the alternatives of the set \(Z\), the above program is rewritten as follows:
Problem 3.2:

\[-2.300\lambda_1 + 2.437\lambda_2 - 4.746\lambda_3 \geq 0\]
\[-3.501\lambda_1 + 3.274\lambda_2 + 0.196\lambda_3 \geq 0\]
\[2.300\lambda_1 - 2.436\lambda_2 + 4.745\lambda_3 \geq 0\]
\[-0.600\lambda_1 + 0.419\lambda_2 + 2.470\lambda_3 \geq 0\]
\[-1.201\lambda^2_1 + 0.837\lambda^2_2 + 4.942\lambda^2_3 \geq 0\]
\[4.600\lambda^2_1 - 4.873\lambda^2_2 + 9.491\lambda^2_3 \geq 0\]
\[1.700\lambda^2_1 - 2.018\lambda^2_2 + 7.216\lambda^2_3 \geq 0\]
\[5.801\lambda^3_1 - 5.710\lambda^3_2 + 4.549\lambda^3_3 \geq 0\]
\[2.901\lambda^3_1 - 2.855\lambda^3_2 + 2.274\lambda^3_3 \geq 0\]
\[-2.900\lambda^4_1 + 2.855\lambda^4_2 - 2.275\lambda^4_3 \geq 0\]
\[\lambda_1 + \lambda_2 + \lambda_3 = 1\]
\[\lambda^2_1 + \lambda^2_2 + \lambda^2_3 = 1\]
\[\lambda^3_1 + \lambda^3_2 + \lambda^3_3 = 1\]
\[\lambda^4_1 + \lambda^4_2 + \lambda^4_3 = 1\]
\[\lambda^i \geq 0, \ i = 1, 2, 3, 4.\]

Using LINDO, the above problem was found to be inconsistent. Hence, the set $Z$ is declared to be inconsistent with the quasiconcave utility function class. Consequently, the set $Z$ is declared to be inconsistent with both utility function classes.

3.8 Conclusions

In this chapter, an integrated unified framework for multiobjective decision making is presented. In this framework, two
general classes of utility functions are handled. The least restrictive form of generalized concavity or convexity (i.e., quasiconcave or quasiconvex) is assumed to represent the decision maker's preference. Procedures for utility class testing, quasiconcave and quasiconvex utility classes, are developed. Furthermore, if the DM's preference happens to be inconsistent with both of these classes, the algorithm for the quasiconcave utility function can be used as an exploratory search method with multiple starting points. Also, two methods for generating a representative set of alternatives as well as an illustrative example are included.
4. The Quasiconcave Utility-based Multiobjective Approach

4.1 Introduction

In this chapter, we present an interactive multiobjective algorithm that will handle quasiconcave utility functions. The algorithm is similar to the Geoffrion, Dyer and Feinberg (GDF) method in the sense that it is based on the Frank and Wolf (FW) algorithm for the single objective nonlinear programming problems. Since the approach is developed in the spirit of the Frank and Wolf algorithm, it should exhibit similar desirable features such as computational simplicity and high initial rate of convergence (Wolf 1970). Our approach includes the concept of projecting the gradient improvement direction on the nondominated frontier as well as an interactive tradeoff-based termination criterion. As a result of that, the drawbacks encountered in the GDF approach, discussed in the next section, will be avoided. Furthermore, although our approach is similar to the method of Korhonen and Laakso (KL) (1986) in the sense that a certain direction is projected on the nondominated frontier, our improvement direction is more meaningful with respect to the DM's implicit utility function. This stems from the fact that it is based on the assessed gradient of his/her utility function, where as Korhonen and Laakso's direction is based on a reference point. Furthermore, KL's reference direction is going to be random at the early iterations of the method since the DM does not have enough information about the size of the nondominated frontier. However,
the gradient-based direction in our algorithm can be assessed using local paired-comparison preference information. Also, KL's projection method will not be able to project from a dominated point (Malakooti 1988b). In addition to that, although KL's algorithm provides a termination criterion with respect to pseudoconcave utility functions, its vital element (the set of generators of the feasible directions cone) was not developed. In the next section, Frank and Wolf's algorithm and its GDF multiobjective extension will be presented in detail. Steps of the interactive algorithm is provided in Section 4.3. In Section 4.4, convergence of the algorithm is discussed. Implementation issues are discussed in Section 4.5. These include issues such as gradient assessment, projection on the nondominated frontier, one dimensional curve search and an interactive termination criterion. Three illustrative examples, computational experiments and conclusions are presented in Sections 4.6, 4.7 and 4.8 respectively.

4.2 Frank-Wolf's Algorithm and its GDF Version

Frank and Wolf (1956) developed a feasible direction approach to solve the quadratic programming problem which generates a sequence of improved points that ultimately converges to the optimal solution which maximizes the objective function. Although the algorithm was originally designed to solve the quadratic programming problem, it could be used to handle the following general concave programming problem:

\[
\text{maximize } f(x) \text{ s.t. } x \in X
\]
where, $f(.)$ is a real-valued differentiable concave function of $n$-vector $x = (x_1, x_2, \ldots, x_n)$ of real variables and $X \subset \mathbb{R}^n$ is a convex compact constraint set of feasible decisions. The steps of the Frank-Wolf procedure follow:

Step 0: Choose $x^1 \in X$, and set $i = 1$.

Step 1: Determine the $y^i$ that solves the direction-finding problem,
maximize $\nabla_x f(x)y$ s.t. $y \in X$.

Let $d^i = y^i - x^i$

Step 2: Determine the step size $\alpha^i$ that solves the step-size finding problem:
maximize $f(x^i + \alpha^i d^i)$ s.t. $\alpha \in (0, 1)$.

Step 3: Check for termination, if $\|x^{i+1} - x^i\| \leq \delta$ STOP; otherwise set $x^{i+1} = x^i + \alpha^i d^i$, $i = i + 1$, and go to step 1.

Geoffrion, Dyer and Feinberg (1972) were one of the first to recognize the possibility of adopting a single objective optimization technique to solve the multiobjective problem. Therefore, they developed an interactive multiobjective approach based on the above algorithm. Their motivation for choosing this algorithm stems from the fact that it is not only easy to implement but it also exhibit appealing features such as its computational simplicity and robust convergence properties (Wolf 1970). Their interactive methodology assumes that the decision maker has an implicit concave utility function "$U(.)" (i.e. its explicit form is not known) as his/her underlying preference structure. Due to the this assumption, Steps 1 and 2 can not be performed in a straightforward manner, since both
the gradient of the utility function as well as utility values of the points along the feasible direction are needed at each iteration of the interactive solution process. Hence, the algorithm requires that local preference information from the DM be elicited in order to implement the main steps of the procedure successfully.

In order to solve the direction-finding problem at Step 1, the concept of The Marginal Rate of Substitution (MRS) (See subsection 4.5.1.1) between objectives based at their current levels was utilized to formulate an equivalent direction-finding problem. This problem is based on the relation between the MRS and the gradient of the implicit utility function "U(.)" as follows: Using the chain rule of calculus, we have,

\[ \nabla_x U(f(x)) = \Sigma_{i=1}^{k} (\partial U(f(x))/\partial f_i(x))\nabla f_i(x). \]

Without loss of generality, assume the first objective "f_1" is the reference objective function and \( \partial U(.)/\partial f_1 > 0 \). Therefore, the gradient vector could be scaled by dividing every component by \( \partial U(.)/\partial f_1 \). So, the scaled gradient vector to be used in the direction-finding problem will be:

\[ \nabla_x U(f(x)) = \Sigma_{i=1}^{k} MRS_{1i}(f(x))\nabla f_i(x), \text{ where } MRS_{11} = 1. \]

The MRS values between the reference objective function "f_1" and each of the other objectives (one at a time) are elicited directly form the DM, keeping the rest of the objectives at their current levels. An MRS question that is posed to the DM might be: "How many units of objective f_1 would you like to acquire in order to give up one unit of objective f_i?"
The step-size in step (3) of the algorithm is obtained indirectly by generating feasible alternatives along the direction $d^i$, then presenting them to the DM in order to select his/her most-preferred one. This alternative is taken to be the starting point of the next iteration unless a stopping criterion holds.

Although the Geofferion et al. concept of adopting the FW algorithm in interactive framework enables the decision maker to systematically explore the possible outcomes (alternatives) in the objective space, the procedure has the following four severe drawbacks:

(1) The requirement to directly elicit the Marginal Rate of Substitution (MRS) is considered to be hard on the decision maker (Wallenius 1975). This results not only in errors in assessing the gradient, but also makes the decision maker uncomfortable with the method.

(2) The alternatives generated during the line search might be dominated. This hurts the credibility of the method if the DM discovers that he was provided with undesirable (dominated) alternatives. That also means the algorithm is not as efficient as it might be.

(3) In practical setting, the final recommended alternative might be dominated, as reported by Wallenius (1975). That was due to premature termination. This serious setback might limit its use in applications.
(4) The termination criterion is fully automated. Consequently, the decision maker does not have control over the conclusion of the solution process.

4.3 The Interactive Gradient Projection-based Algorithm

The developed interactive algorithm builds on the FW as well as the GDF algorithms. It differs from the GDF approach in that it does not preform a one dimensional line search; rather, it searches a nondominated curve. Alternatives on this curve are generated using a mathematical programming-based generation scheme to be evaluated by the decision maker. This process will eliminate the possibility of providing the DM with a dominated outcomes. This will ensure that the interaction will be meaningful due to the fact that nondominated alternatives are the only competitive outcomes to the problem. Our method also differs in that it provides an interactive tradeoff-based termination criterion. This will give the DM control over the final outcome of the process. The steps of the algorithm is presented in the next subsection.

4.3.1 The Steps of the Algorithm

In this subsection, the steps of the quasiconcave utility-based interactive algorithm are presented. However, its convergence and the implementation issues concerning its steps are discussed in Sections 4.4 and 4.5 respectively. Step 0 of the algorithm is the initialization stage, where any of the nondominated alternatives generated previously for the utility class testing can be used. Step 1
includes two tasks. In first task, the gradient of the utility function is assessed. Two methods for accomplishing this assessment are presented in subsection 4.5.1. Using the assessed gradient, a gradient-based improvement direction is obtained in the second task. This is done by solving a direction finding problem. Step 2 of the algorithm includes one of the main departures of the GDF method. It includes projecting the improvement direction obtained in the previous step on the nondominated frontier. Two methods for accomplishing the projection task are presented in subsection 4.5.2. Step 3 includes one dimensional curve search. This is accomplished by presenting the sample of nondominated alternatives resulted in the previous step to the decision maker for evaluation. The result of this step identifies the most-preferred alternative among the set presented to the DM. Step 4 includes checking for termination. If the current most-preferred alternative is the same as the one for the previous iteration, the current most-preferred alternative is considered to be a candidate for optimality with respect to the implicit quasiconcave utility function and the interactive termination criterion is invoked; otherwise, a new direction of improvement is determined. The interactive termination criterion is discussed in subsection 4.5.4. The steps of the algorithm is presented next.

Step 0 (Initialization)

(0.a) Generate an initial efficient solution. Denote it \( x^1 \).

(0.b) Set \( i = 1 \).
Step 1 (Direction Finding)

(1.a) Assess DM's utility function gradient at the \( f(x^i) \).

(1.b) Determine the point \( f(y^i) \) such that \( y^i \) solves the following direction problem:

\[
\begin{align*}
\text{maximize} & \quad \nabla_x U(f_1(x^i), f_2(x^i), \ldots, f_k(x^i))y \\
\text{s.t.} & \quad y \in X.
\end{align*}
\]

(c) Set \( d^i = f(y^i) - f(x^i) \)

Step 2 (Projection)

Generate \( M^i \) nondominated alternatives along a nondominated curve by projecting the gradient-based improvement direction \( d^i \) on the nondominated frontier, where \( M^i \) is predefined.

Step 3 (One-dimensional Curve Search)

Interact with the decision maker to select the best among the generated \( (M^i+2) \) alternatives, denote the associated efficient solution as \( y^* \).

Step 4 (Check for Termination)

(4.a) \( \text{IF} \ f(y^*) \) is the same as \( f(x^i) \), then check the interactive termination criterion (ITC). If the conditions are satisfied, declare \( f(x^i) \) as the most-preferred alternative and \( x^i \) as the most-preferred decision, STOP; otherwise, determine any alternative that is better than \( f(x^i) \), call it \( f(y^*) \) and go to (4.b). \( \text{ELSE} \) go to (4.b).

(4.b) set \( x^{i+1} = y^* , \ i = i + 1 \), go to Step 1.
4.4 Convergence

The algorithm would generate, through Step 3, a sequence of efficient points as well as nondominated alternatives with strictly increasing utility values unless the alternative is a candidate for termination, assuming the DM is consistent in his/her responses. Frank and Wolf's (1956) algorithm, which our interactive algorithm is based on, is a point-to-set composite mapping which would converge to a solution under the conditions of the global convergence theorem assuming a convex constraint set (Zangwill 1969). However, our algorithm guarantees that alternatives generated at each iteration are as good as or better than the ones generated by the GDF algorithm in terms of their utility values. Consequently, our algorithm is expected to converge faster than the GDF. To illustrate the difference between our algorithm and the GDF algorithm in terms of convergence, a simple problem (similar to example 4.1) is solved. It turns out that our algorithm took only four iterations to conclude (there was no one-dimensional curve search in the fourth iteration), where the GDF algorithm took more than six iteration to conclude.

4.5 Implementation Issues

The basic software requirement as far as implementation is concerned is an appropriate single optimization code. An initial nondominated alternative could be easily obtained by selecting one of the nondominated alternative generated at the stage of testing for the class of utility function. However, implementation of the next
step of the algorithm involves an extra work since the utility function, whose gradient is required, is not explicitly known. Although the assessment of utility gradient is required based on the local preference information elicited from the DM, he/she should not be burdened with difficult questions regarding his preference, such as providing the MRS values directly. Once the gradient of the utility function is assessed (See subsection 4.5.2), the direction finding problem is solved to obtain an improvement direction. Precisely, the solution $f(y^i)$ to the direction problem indicates a feasible point on the direction ray emanating from the current nondominated alternative $f(x^i)$, where the utility is increasing in the objective space.

Having obtained the improvement direction, the next step is to generate an equally spaced points on the line between $f(x^i)$ and $f(y^i)$ and project them onto the nondominated frontier. This process results in generating a sample of nondominated alternatives which will be displayed to the decision maker for preference evaluation. Having decided on a certain alternative, another iteration begins unless the chosen alternative is the same as the one in the previous iteration. If this happens, the interactive termination criterion of subsection 4.5.4 is checked. If condition for termination is satisfied, the solution process is concluded; otherwise one of the tradeoffs being accepted is used to generate a direction of improvement. Next, methods for assessing the gradient are presented. In subsection 4.5.2, methods for accomplishing the projection process are
presented. Discussion for performing the one-dimensional curve search is presented in subsection 4.5.3. Finally, the interactive termination criterion (ITC) is presented in subsection 4.5.4.

4.5.1 Gradient Assessment

Due to different styles and levels of sophistication of the decision makers, two different methods for assessing the gradient of the utility function are presented. The first method is based on assessing the marginal rate of substitution (MRS) indirectly. However, the second method is based on paired-comparison preference information through the strength of preference (Malakooti 1988). Next, the MRS-based method is presented. The second method is presented in subsection 4.5.1.2.

4.5.1.1 Marginal Rate of Substitution-based Assessment

This method of assessment is based on obtaining the marginal rate of substitution (MRS) indirectly through interaction with the decision maker. MRS components are obtained using the Dyer's interactive routine (Dyer 1973), which is discussed later in the subsection. The marginal rate of substitution between two objectives, keeping the rest of the objectives at their current levels, is expressed as follows,

\[ MRS_{ij} = \frac{\partial U(f)/\partial f_j}{\partial U(f)/\partial f_i}, \quad MRS_{ii} = 1. \]

Consider the following matrix whose entries \( a_{ij} = MRS_{ji} \) and \( a_{ji} = MRS_{ij} = (1/MRS_{ji}) \).
\[
M = \begin{bmatrix}
M_{11} & M_{12} & \cdots & M_{1k} \\
M_{21} & M_{22} & \cdots & M_{2k} \\
\vdots & \vdots & \ddots & \vdots \\
M_{k1} & M_{k2} & \cdots & M_{kk}
\end{bmatrix}
\]

The above matrix has positive entries and called a reciprocal matrix since it satisfies the reciprocal property \( a_{ji} = 1/a_{ij} \), as \( M_{ij} = (1/M_{ji}) \). Using this matrix \( M \) as well as the gradient of the utility function \( \nabla U(f) \), the following relation is true,

\[
M \nabla U(f) = \lambda \nabla U(f), \quad \text{where } \lambda = \text{eigenvalue of the matrix } M.
\]

Given the matrix \( M \), \( \nabla U(f) \) is recovered by solving the system \((M - \lambda I) \nabla U(f) = 0\) which has a nonzero solution if and only if \( k \) is an eigenvalue of \( M \). Because of the reciprocal property, every row is a constant multiple of the first row and hence the matrix \( M \) has unit rank. Consequently, all of the eigenvalues are zero except for only one, which is equal to \( k \). Since the above matrix entries will be assessed based on the decision maker's responses and the ratios obtained might not be perfectly consistent, the actual system which is solved becomes

\[
M' \nabla U(f) = \lambda_{\text{max}} \nabla U(f).
\]

In order to recover the estimated gradient vector \( \nabla U(f) \), the eigenvector method of Saaty (1977) can be used since it was developed to handle a system with a similar structure. It computes
\( \nabla'W(f) \) as the principal right eigenvector of the matrix \( M' \) in the above system, where \( \lambda_{\text{max}} \) is the maximum eigenvalue of \( M' \). The computation of the principal right eigenvector can be accomplished by raising the matrix \( M' \) to increasing powers of \( n \) and then normalizing the resulting system (Harker 1989):

\[
\mathbf{w} = \lim_{n \to \infty} \mathbf{M}^n \mathbf{e}/\mathbf{e}^T \mathbf{M}^n \mathbf{e}, \quad \text{where } \mathbf{w} = \nabla'W(f) \text{ and } \mathbf{e} = (1, 1, \ldots, 1)^T.
\]

In practice, successive values for \( \mathbf{w} \) are calculated starting at the value of \( k \) equals to 1. This process concludes with a final value for \( \mathbf{w} \) whenever the values of \( \mathbf{w}^n \) stabilizes (i.e. \( \mathbf{w}^{n+1} = \mathbf{w}^n \)).

An advantage of the above gradient assessment formulation, in addition to its ability to recover the gradient by the eigenvector method, is that the degree of consistency can be determined. This is accomplished using Saaty's consistency ratio (C.R.) to determine whether the assessed matrix \( M' \) needs its components to be revised. The process of checking for consistency starts by computing the eigenvalue \( \lambda_{\text{max}} \) as follows:

\[
\lambda_{\text{max}} = (\Sigma_{j=1}^k a_{ij} w_j)/w_i \quad \text{for any } i = 1, \ldots, k,
\]

where \( \mathbf{w} = (w_1, w_2, \ldots, w_k) = \mathbf{w}^n \) as calculated above. Having determined \( \lambda_{\text{max}} \), Saaty's consistency index (C.I.) is calculated. This index defined as follows:

\[
\text{C.I.} = (\lambda_{\text{max}} - k)/(k-1).
\]

For different matrix sizes, random matrices were generated and their mean C.I. value, called the random index (R.I.), was computed; These values are illustrated in Table 4.1 next.
Table 4.1: random indices for n x n matrices.

<table>
<thead>
<tr>
<th>n</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>R. I.</td>
<td>0.00</td>
<td>0.58</td>
<td>0.90</td>
<td>1.12</td>
<td>1.24</td>
<td>1.32</td>
</tr>
</tbody>
</table>

Using these values, Saaty's consistency ratio (R.I.) is defined as the ratio of the C.I. to the R.I.; therefore, the C.R. value is a measure of how a given matrix compares to a purely random matrix in terms of their C.I.'s. Therefore,

\[
\text{C.R.} = \frac{\text{C.I.}}{\text{R.I.}}
\]

A value of the consistency ratio (C.R.) \( \leq 0.1 \) is considered to be acceptable, where larger values require the matrix \( M' \) to be revised in order to reduce the inconsistencies in judgements.

As far as assessing the marginal rate of substitution \( \text{MRS}_{ij} \) components are concerned, it can be accomplished using Dyer's interactive routine (Dyer 1973). Let \( f^* \) be an alternative where the gradient of the utility function needs to be assessed. The process is done in a column by column fashion of the matrix \( M \). As an example, consider the first column. Objective function \( f_1 \) is taken to be the reference objective and the \( \text{MRS} \) values for this column are as follows:

\[
\text{MRS}_{1j} = \frac{\partial U(f^*)/\partial f_j}{\partial U(f^*)/\partial f_1}, \ j = 2, \ldots, k; \ \text{MRS}_{11} = 1.
\]
Dyer's interactive routine (Dyer 1972) assesses those MRS values using a series of pairwise comparisons. The decision maker is asked to compare the following two alternatives:

\[ f^* = (f^*_1, f^*_2, \ldots, f^*_j, \ldots, f^*_k) \]

and

\[ f'^* = (f^*_1 + \Delta_1, f^*_2, \ldots, f^*_j - \Delta_j, \ldots, f^*_k), \]

where \( \Delta_1 \) and \( \Delta_j \) are small changes relative to \( f^*_1 \) and \( f^*_j \) levels respectively, but large enough to be significant to the decision maker. If the decision maker prefers \( f^* \) over \( f'^* \), \( \Delta_j \) is decreased. However, if the decision maker prefers \( f'^* \) over \( f^* \), \( \Delta_j \) is increased. This process is repeated till indifference is obtained; then

\[ \text{MRS}_{1j} = \Delta_j / \Delta_1. \]

Since \( M \) is reciprocal matrix, \( \text{MRS}_{j1} = 1 / \text{MRS}_{1j} \) and consequently assessments of the first column yields assessments the first row. Therefore, total number of MRS assessments required = \( \Sigma_{i=1}^{k} i - k = k(k-1)/2 \), since every diagonal element equals 1. This is also termed the "indifference tradeoff" weighting method (Hobbs 1980). He has found that these judgments can be quite inconsistent.

4.5.1.2 Paired Comparison-based Assessment

This method of assessment is an extension of a method suggested by Malakooti (1988a) in a discrete alternative setting. Let \( f^* \) be an alternative where the gradient of the utility function needs to be assessed. Let first order approximation of the utility function in the vicinity of \( f^* \) be represented as follows:

\[ V(f) = U(f^*) + \nabla U(f^*)(f - f^*), \]
where \( \nabla U(f^*) \) is the gradient of \( U(f) \) at \( f^* \).

The method starts by generating a number (e.g. \( k \)) of alternatives that are close to \( f^* \). This can be accomplished by changing randomly components of \( f^* \) by a significant amounts. This process yields \( k \) alternatives as follows:

\[
\begin{align*}
    f^1 &= (f_1^* \pm \Delta_1, f_2^*, \ldots, f_k^*), \\
    f^2 &= (f_1^*, f_2^* \pm \Delta_2, \ldots, f_k^*), \\
    \quad & \vdots \\
    f^k &= (f_1^*, f_2^*, \ldots, f_k^* \pm \Delta_k).
\end{align*}
\]

Let the linear approximation \( V \) of \( U \) at alternative \( f^* \) be represented by the following linear function:

\[
V(f) = \Sigma_{i=1}^{k} w_i f_i + cU(f^*),
\]

where \((w_1, \ldots, w_k) = w\) is the normalized gradient assessed at \( f^* \) and \( c \) is a constant. Next, paired comparisons with strength of preference are elicited from the decision maker using the following categories:

- **A**: \( f^i \) is preferred to \( f^j \) very strongly
- **E**: \( f^i \) is preferred to \( f^j \) strongly
- **I**: \( f^i \) is preferred to \( f^j \) moderately
- **O**: \( f^i \) is preferred to \( f^j \) weakly
- **U**: \( f^i \) is preferred to \( f^j \) very weakly, or the DM is indifferent between the two alternatives.

Using the obtained preference information, two types of constraints are formulated. They are presented next.
(a) **Preference Constraints**: if \( f^i \) is preferred to \( f^j \), the following constraint is generated, where \( \varepsilon \) is a small positive number.

\[
V(f^i) - V(f^j) + IP_{ij} \geq \varepsilon.
\]

If the DM is indifferent between \( f^i \) and \( f^j \), the above constraint is written as:

\[
V(f^i) - V(f^j) + IP_{ij} - IP_{ji} = 0,
\]

where \( IP_{ij} \) and \( IP_{ji} \) are inconsistency compensation variables.

(b) **Preference Rating Constraints**: translating the A, E, I, O and U rating into numerical ratings, constraints are generated on the weight vector \( w \). Therefore, using the DM strength of preference response for a preference of \( f^i \) over \( f^j \), the following constraint is generated:

\[
V(f^i) - V(f^j) + IC_{ij} - IC_{ji} = Ay, Ey, Iy, Oy or Uy,
\]

where \( IC_{ij} \) and \( IC_{ji} \) are inconsistency compensation variables and \( y \) is a positive normalizing variable.

Having elicited the decision maker preferences for all of the \((k + 1)\) alternatives, the following linear programming problem is formulated. It provides an approximation of the normalized weighting vector \( w \).

**Program 4.1**: minimize

\[
z = P_1(\Sigma IP_{ij}) + P_2(\Sigma IC_{ij})
\]

s.t.

\[
V(f^i) - V(f^j) + IP_{ij} \geq \varepsilon.
\]

\[
V(f^i) - V(f^j) + IP_{ij} - IP_{ji} = 0,
\]

\[
V(f^i) - V(f^j) + IC_{ij} - IC_{ji} = Ay, Ey, Iy, Oy or Uy,
\]

\[
IP_{ij}, IP_{ji}, IC_{ij}, IC_{ji} \geq \varepsilon, y \geq 1,
\]

\[
\Sigma_{i=1}^{k} w_i = 1, w_i \geq 0,
\]
where \( P_1 >> P_2 > 0 \) and \( \epsilon \) is a very small positive real number. \( P_1 \) and \( P_2 \) are positive real numbers that can be assumed. Inconsistency in paired comparisons is more heavily penalized than in strength of preference since \( P_1 >> P_2 \). As an example, the values \( P_1 = 20 \), \( P_2 = 5 \) and \( \epsilon = 0.001 \) can be used.

### 4.5.2 Projection

In the second step of the algorithm, \( M^i \) nondominated alternatives along a nondominated curve associated with the line connecting the current alternative \( f(x^i) \) and the alternative \( f(y^i) \) are generated using a generating scheme, where \( M^i \) is predefined at the \( i \)-th iteration. The process begins by generating evenly spaced \( M^i \) convex combinations of the end points \( f(x^i) \) and \( f(y^i) \) of the direction line. These points, denoted as \( f^j, \ j = 1, \ldots, M^i; \) are generated using the following formula:

\[
f^j = ((M^i-j)/(M^i-1))f(x^i) + ((j-1)/(M^i-1))f(y^i), \ j = 1, \ldots, M^i \quad (4.1)
\]

Then, those points are projected on the nondominated frontier, since the convex combinations will likely be either dominated (in the convex feasible region case) or infeasible (in the nonconvex case). The projection process can be accomplished in either of two ways. They are discussed in the next two subsections. See Figures 4.1 and 4.2 for the Tchebycheff-based and the Deviation-based schemes respectively. In these two figures, point A is dominated and point B is infeasible and points A' and B' are their projections respectively.
Figure 4.1: Tchebycheff-based Projection

Figure 4.2: Deviation-based Projection.
4.5.2.1 Tchebycheff-based Projection

Having generated the convex combinations using formula 4.1 above, the scheme begins by calculating the associated Tchebycheff weights for each point \( \tilde{f}_j, j = 1, \ldots, M \); with respect to the Utopian point \( f^{**} \), using the following formula (Steuer 1986):

\[
    w_i = \frac{1/(f^{**}_i - \tilde{f}_i(x))}{\sum_{i=1}^{k} (1/(f^{**}_i - \tilde{f}_i(x)))}, \quad i = 1, \ldots, k
\]  

(4.2)

Using those weights, the augmented weighted Tchebycheff-based generating program, presented next, is solved to determine the associated nondominated alternatives. Theorem 4.1, next, shows that program 4.2 generates an efficient point.

Program 4.2:  minimize \( ||f^{**} - f(x)||_w \cdot \rho = ||f^{**} - f(x)||_w + \rho e^t (f^{**} - f(x)) \)

\( x \in X \)

where \( \rho \) is a sufficiently small positive scalar and \( e^t \) is a vector of ones.

Theorem 4.1

Consider Program 4.2. If \( x^* \) is an optimal solution, then \( x^* \) is efficient.

Proof (by contradiction)

Let \( x^* \) solves program 4.2 and \( x^* \) is not an efficient point. Since \( x^* \) is not efficient, then there exist an \( x' \in X \) such that \( f(x') \geq f(x^*) \).

Then \( (f^{**} - f(x')) \leq (f^{**} - f(x^*)) \). Because \( w \geq 0 \) then \( \max_i \{w_i(f^{**}_i - \tilde{f}_i(x'))\} \leq \max_i \{w_i(f^{**}_i - \tilde{f}_i(x^*))\} \). Therefore \( ||f^{**} - f(x')||_w \leq ||f^{**} - f(x^*)||_w \). Also, since \( \rho > 0 \) and \( f(x') \geq f(x^*) \), then \( \rho e^t (f^{**} - f(x')) < \rho e^t (f^{**} - f(x^*)) \). Therefore \( ||f^{**} - f(x')||_w + \rho e^t (f^{**} - f(x')) < ||f^{**} - f(x^*)||_w + \rho e^t (f^{**} - f(x^*)) \). This
implies that $\|\|f^{**} - f(x^*)\|\|_\rho < \|\|f^{**} - f(x^*)\|\|_\rho$ and hence $x^*$ does not solve program 4.2, a contradiction. Therefore $x^*$ is efficient, Q.E.D.

Program 4.2 can be implemented by program 4.3 presented next.

Program 4.3: minimize $\{ \alpha + \rho e^{t}(f^{**} - f(x)) \}$

\[
\begin{align*}
& x \\
& \text{s.t.} \\
& w_i(f_i^{**} - f_i(x)) \leq \alpha \quad 1 \leq i \leq k \\
& x \in X,
\end{align*}
\]

4.5.2.2 Deviation-based Projection

A second way is based on a mathematical programming formulation, in which the overdeviation and the underdeviation of a given alternative from the nondominated frontier are optimized lexicographically. This a generalization of a method suggested by Malakooti (1987). Let $f^j, j = 1, \ldots, M^j$ be the convex combinations generated using formula 4.1. As mentioned before, each one of these points is likely to be either dominated (in the convex feasible feasible region case) or infeasible (in the nonconvex case). Consider one of those points $f^j$. In the case of being dominated, let $y^-$ be the underdeviation of an $f^j$ from the desired nondominated point $f(x^j)$ of $x^j$ which is to be solved for. Also, in the case of being infeasible, let $y^+$ be the overdeviation from the desired $f(x^j)$. Consider one of these alternatives, call it $f^j$, and let $f^j(x^j)$ be its projection on the nondominated frontier. Therefore, the following relations are true, in the case that $f^j$ is dominated:
\[ \exists \; x^j, y^{-j} \; \ni \; f_i(x^j) = f_i^j + y^{-j}, \; y_i^{-j} \geq 0, \; i = 1, \ldots, k \]  
(4.3)

in the case that \( f^j \) is infeasible:

\[ \exists \; x^j, y^{+j} \; \ni \; f_i(x^j) = f_i^j - y^{+j}, \; y_i^{-j} \geq 0, \; i = 1, \ldots, k \]
(4.4)

Therefore, to project \( f^j \), one of two optimization problems should be constructed and solved. One minimizes the overdeviation in the case of \( f^j \) being dominated and the other maximizes the underdeviation in the case of \( f^j \) being infeasible.

Let's consider the first case where \( f^j \) is dominated. It can be projected by maximizing the underdeviation of \( f^j \) from the nondominated frontier using the following formulation:

Program 4.4: \[ Z^- = \maximize \; w^t y^- \]
\[ \{x, y^-\} \]

s.t.
\[ f_i(x) - y_i^- = f_i^j \quad 1 \leq i \leq k, \]
\[ x \in X, \; y_i^- \geq 0, \quad 1 \leq i \leq k. \]

However, in the second case where \( f^j \) is infeasible, it can be projected by minimizing the overdeviation of \( f^j \) from the nondominated frontier using the following formulation:

Program 4.5: \[ Z^+ = \minimize \; w^t y^+ \]
\[ \{x, y^+\} \]

s.t.
\[ f_i(x) + y_i^+ = f_i^j \quad 1 \leq i \leq k, \]
\[ x \in X, \; y_i^+ \geq 0, \quad 1 \leq i \leq k. \]
The projection process using this method can accomplished using the following steps:

Step 1: Solve program 4.4.

If $Z^- > 0$ stop, else go to Step 2.

Step 2: Solve program 4.5 and exit.

Theorem 4.2, next, shows that programs 4.4 and 4.5 each generates an efficient point individually for each case. The cases of feasible and infeasible $f^i$, respectively.

**Theorem 4.2**

Consider Programs 4.4 and 4.5. If $(x^*, y^-)$ and $(x^*, y^+)$ is an optimal solution for Programs 4.4 and 4.5 respectively, then $x^*$ is efficient.

**Proof (by contradiction)**

It will be shown for one case, since the proof for the other case is similar. Consider the optimization problem 4.5 where the point $f^j$ is infeasible and let $(x^*, y^+)$ solve program 4.5 and assume $x^*$ is not an efficient point. Since $x^*$ is not efficient, then there exists an $x' \in X$ such that $f(x') \geq f(x^*)$. Now $f(x') = g - y^+$ and $f(x^*) = g - y^+$, hence $y^+ \leq y^+$. Since $w \geq 0$ then $w^ty^+ < w^ty^+$, and hence $x^*$ does not solve program 4.5. This is a contradiction and, therefore, $x^*$ is efficient, Q.E.D.

In order to combine the above two optimization problems in one program, we consider the two deviations at the same time in one relationship. Consider alternative $f^j$ to be projected, and let $f^j(x^j)$ be
its projection on the nondominated frontier. Therefore, the following relations are true,
\[
\hat{f}(x^j) = f^j + y^{-j} - y^{+j}, \quad y_i^{+j}, y_i^{-j} \geq 0, \\
(\sum_{i=1}^{k} y_i^{+j})(\sum_{i=1}^{k} y_i^{-j}) = 0, \quad i = 1, \ldots, k
\]  
\[(4.5)\]
A combined mathematical programming formulation is presented next, using a scalar constant \(M\) to account for the two cases,

Program 4.6: \begin{align*}
\text{minimize} & \quad Mw^t y^+ - w^t y^- \\
\text{s.t.} & \quad f_i(x) - y^- + y^+ = f_i^j, \quad 1 \leq i \leq k, \\
& \quad x \in X, \\
& \quad y_i^{-}, y_i^{+} \geq 0, \quad 1 \leq i \leq k.
\end{align*}

where \(f^j\) is a given point, \(M > 1.0\) (e.g. \(M = 1.1\)), and \(w \geq 0, \sum_{i=1}^{k} w_i = 1\) are given weights.

The two criteria for selection between the two schemes are computational efficiency and the quality of the resulted alternatives. Those two criteria might be assessed through simulations of decision maker-based computational experiments. Furthermore, weights in the Deviation-based projection method must be selected by the DM—an extra step. This might yield a projected alternative where only one objective value is improved while keeping the others at their current levels. This due to the fact that the deviation of only one objective (say \(f_p\)) is optimized, if its associated deviation is the only one that has a nonzero weight. However, weights in the Tchebycheff-based projection method are set using formula 4.2.
4.5.3 One-dimensional Curve Search

In the third step of the algorithm, nondominated alternatives resulted from the projection and the two end points are presented to the decision maker for evaluation. He/she is asked to indicate his/her most preferred alternative. If this alternative is the same as the one preferred in the previous iteration, the interactive termination criterion (subsection 4.5.4) is checked. If it is satisfied, the solution process is concluded with the current nondominated alternative as well as its associated efficient decision taken to be the most-preferred alternative and the most-preferred decision for the whole problem respectively. Otherwise, a better alternative close to the current one is identified using one of the rejected tradeoffs (See subsection 4.5.4) and the algorithm reiterated with this solution. But, if the recent preferred alternative is not the same as the previous preferred one, the algorithm iterates with the recent alternative as the starting point for the next iteration.

4.5.4 The Interactive Termination Criterion

In order to ensure that decision maker has control over the conclusion of the algorithm, an interactive termination criterion for the solution process is clearly needed. The desired criterion ought not only avoid asking for unreasonable amounts of preference information but it should also be able to provide a new improvement direction, if termination condition is not satisfied.
The concept of tradeoff in the objective space will be utilized to play a key role in our termination method. Since every tradeoff in the objective space is induced by a direction in the decision space, it is amenable to work in the decision space because it is already constructed through its function constraints. This will be very clear when the model is linearly constrained. In general, the method for termination begins by generating a sufficient set of directions in the decision space at the point of interest. Then, using their associated tradeoffs, the decision maker is asked to answer simple questions regarding his preference toward some or all of them. Finally, the generated preference information as well as the above directions is utilized to check for the conclusion of the solution process.

Let the convex feasible set of the multiobjective programming model be represented as follows:

\[ X = \{ x \mid g_i(x) \leq 0, \ i = 1, \ldots, m \}. \]

Let \( x^* \) be an efficient point where it is desired to test for termination. The concept of feasible direction plays a key role in the process due to the fact the set of all feasible directions emanating from \( x^* \) should be considered.

**Definition 4.1 (Zangwill 1969)**

A direction \( d \) is called a feasible direction at point \( x^* \in X \) if there exists a \( \sigma > 0 \) such that \( (x^* + \tau d) \in X \) for all \( 0 \leq \tau \leq \sigma \).
Definition 4.2

The set denoted \( D(x^*) \) is called the set of feasible directions at \( x^* \) if it consists of all feasible directions \( d = x - x^*, x \in X \).

Using the active constraints at point \( x^* \), a convex polyhedral cone \( C(x^*) \) can be constructed. Let the set \( I(x^*) = \{ 1, \ldots , r \} \) be the set of the active constraints (i.e. \( g_i(x^*) = 0 \)). Then \( C(x^*) = \{ d | \nabla g_i(x^*)d \leq 0 \} \) and the set \( D(x^*) \) is contained in the cone \( C(x^*) \) (Bazaraa and Shetty 1979). Since the cone \( C(x^*) \) is a convex polyhedral cone with a vertex \( x^* \), it can be represented using its minimal generators and can be written as follows:

\[
C(x^*) = \{ x | x = x^* + \sum_{i=1}^{q} \mu_i d^i, \mu_i \geq 0 \},
\]

where \( d^i, i = 1, \ldots , q \) are generation directions.

Consider the utility maximization problem,

\[
\text{maximize } U(x) = U(f(x))
\]

s.t. \( x \in X \).

Next, theorems 4.3 and 4.4 establish a sufficient condition for point \( x^* \) to be a global optimal solution, assuming the utility function "\( U(\cdot) \)" is quasiconcave or strictly quasiconcave on the convex constraint set \( X \) respectively using the enclosing cone \( C(x^*) \). The conditions are based on the directional derivative value with respect to the generators of \( C(x^*) \). It should be mentioned that these conditions are also necessary in the case of linear constraint set where every direction in the cone \( C(x^*) \) is feasible and therefore will be in the set \( D(x^*) \).
Theorem 4.3

Let the feasible region $X$ be convex, the utility function $U(f(x))$ be quasiconcave on $X$, and $x^* \in X$. Furthermore, consider the cone $C(x^*) = \{ x \mid x = x^* + \sum_{i=1}^{q} \mu_i d_i, \mu_i \geq 0 \}$. If $\nabla U(x^*)d^i \leq 0$ for all $d^i$, $i = 1, \ldots, q$ and there exist an $i$ such that $\mu_i > 0$ with $\nabla U(x^*)d^i < 0$, then $x^*$ is a strict global optimal solution.

Proof (by contradiction)

It is sufficient to show that $x^*$ is a strict local optimum (i.e. $U(x^*) > U(x)$ for all $x$ in the neighborhood of $x^*$) (Avriel 1976). Therefore for the contradiction, let $x'$ be a feasible point in the neighborhood of $x^*$ such that $U(x') \geq U(x^*)$. Since $x'$ is a feasible point and $X$ is convex, then $d' = (x' - x^*)$ is a feasible direction. This implies that $d' \in D(x^*)$ and hence, by assumption, $d' \in C(x^*)$. Therefore it could be expressed as a linear combination of $C$'s generators. Then there exist $\mu_i \geq 0$ such that $d' = \sum_{i=1}^{q} \mu_i d_i$. Therefore $\nabla U(x^*)d' = \nabla U(x^*)(\sum_{i=1}^{q} \mu_i d_i) = \mu_1 \nabla U(x^*)d^1 + \ldots + \mu_q \nabla U(x^*)d^q < 0$, since $\nabla U(x^*)d^i \leq 0$ for all $d^i$, $i = 1, \ldots, q$ and there exist at least one $i$ such that $\mu_i > 0$ with $\nabla U(x^*)d^i < 0$. Now, since $U(.)$ is quasiconcave, $U(x') \geq U(x^*)$ implies $\nabla U(x^*)(x' - x^*) \geq 0$, and therefore $\nabla U(x^*)d' \geq 0$. This is a contradiction. Thus, $x^*$ is a strict local optimal solution. Since $U(.)$ is quasiconcave, then $x^*$ is also a strict global optimal solution, Q.E.D.

Lemma 4.1

Let $X$ be a convex set and $U(.) : \mathbb{R}^k \rightarrow \mathbb{R}$ be a strictly quasiconcave and differentiable on $X$. If $U(x') > U(x^*)$, then $\nabla U(x^*)(x' - x^*) > 0$. 
Proof

Let \( x', x^* \in X \) such that \( U(x') > U(x^*) \). Since \( U(.) \) is strictly quasiconcave and differentiable, \( U((1-\lambda)x^* + \lambda x') > U(x^*) \) for \( \lambda \in (0,1) \). Also, by Taylor series expansion, \( U((1-\lambda)x^* + \lambda x') = U(x^*) + \lambda \nabla U(x^*)(x' - x^*) + \lambda \|x' - x^*\| \alpha(x^*; \lambda(x'-x^*)) \) for \( \lambda \in (0,1) \), where \( \lambda \|x' - x^*\| \alpha(x^*; \lambda(x'-x^*)) \to 0 \) as \( \lambda \to 0 \). Since \( U((1-\lambda)x^* + \lambda x') - U(x^*) > 0 \), then \( \lambda \nabla U(x^*)(x' - x^*) + \lambda \|x' - x^*\| \alpha(x^*; \lambda(x'-x^*)) > 0 \). Now dividing by \( \lambda \) and as \( \lambda \to 0 \), we have \( \nabla U(x^*) (x' - x^*) > 0 \), Q.E.D.

Theorem 4.4

Let the feasible region \( X \) be convex, the utility function \( U(.) \) be strictly quasiconcave on \( X \), and \( x^* \in X \). Furthermore, consider the cone \( C(x^*) = \{ x | x = x^* + \sum_{i=1}^q \mu_i d_i, \mu_i \geq 0 \} \). If \( \nabla U(x^*) d_i \leq 0 \) for all \( d_i, i = 1, \ldots, q \), then \( x^* \) is a global optimal solution.

Proof (by contradiction)

It is sufficient to show that \( x^* \) is a local optimum (i.e. \( U(x^*) \geq U(x) \) for all \( x \) in the neighborhood of \( x^* \)) (Bazaraa and Shetty 1976).

Therefore for the contradiction, let \( x' \) be a feasible point in the neighborhood of \( x^* \) such that \( U(x') > U(x^*) \). Since \( x' \) is a feasible point and \( X \) is convex, then \( d' = (x'-x^*) \) is a feasible direction. This implies that \( d' \in D(x^*) \) and hence, by assumption, \( d^* \in C(x^*) \). Therefore it could be expressed as a linear combination of \( C \)'s generators. Then there exist \( \mu_i \geq 0 \) such that \( d' = \sum_{i=1}^q \mu_i d_i \). Therefore \( \nabla U(x^*) d' = \nabla U(x^*) (\sum_{i=1}^q \mu_i d_i) = \mu_1 \nabla U(x^*) d_1 + \ldots + \mu_q \nabla U(x^*) d_q \leq 0 \). Now, since \( U(.) \) is strictly quasiconcave, \( U(x') > U(x^*) \) implies that \( \nabla U(x^*)(x'-x^*) > 0 \), by lemma 4.1 above, and therefore \( \nabla U(x^*) d' > 0 \). This is a contradiction.
Thus, $x^*$ is a local optimal solution. Since $U(.)$ is strictly quasiconcave, then $x^*$ is also a global optimal solution, Q.E.D.

The main issue in the above generators-based representation of the enclosing cone $C(x^*)$ is the efficient construction of the generation directions $d^i$, $i = 1, \ldots, q$. For this reason, we introduce the concept of the enclosing polyhedral set next. Through this concept, the original feasible direction set $D(x^*)$ of the set $X$ at $x^* \in X$ will be enclosed by a larger polyhedral cone $C(x^*)$, where $C(x^*)$ is the set of feasible directions of $X^*$ at $x^* \in X^*$ that consists of all feasible directions $d = x - x^*$, $x \in X^*$. Next, we provide a formal definition for the enclosing cone.

**Definition 4.3**

A set $X^* = \{Ax \leq b, x \geq 0\}$ is an enclosing polyhedral set of the set $X = \{ x \mid g_i(x) \leq 0, i = 1, \ldots, m \}$ if $X$ is a subset of $X^*$.

We assume that the enclosing set $X^*$ can be constructed using the linear approximation of the active constraints at point $x^*$. Let $I(x^*) = \{ i \mid g_i(x^*) = 0 \}$ be the set of active constraints indices at point $x^*$. Using the Taylor series expansion, the linear approximation of $g_i(x)$ around the point $x^*$ is as follows, $g^*_i(x) = g_i(x^*) + \nabla g_i(x^*)(x - x^*)$. Therefore the enclosing set is represented as $X^* = \{ x \mid g^*_i(x) \leq 0, i \in I(x^*), x \geq 0 \}$. It can also be expressed in canonical form as $X^* = \{ Ax = b, x \geq 0 \}$, where $A$ is an $r \times n$ matrix with row $a_i = (\nabla g_i(x^*), 1_i)$ adjusting for the slack variable and $b$ is a $r \times 1$ vector.
Theorem 4.5

Let \( x^* \in \text{boundary}(X) \) and \( X^* = \{ x \mid g_i(x^*) + \nabla g_i(x^*)(x - x^*) \leq 0, i \in I(x^*) \} \). Then \( X^* \supset X \).

Proof

Since \( x^* \in \text{boundary}(X) \), the hyperplane \( g_i^*(x) = g_i(x^*) + \nabla g_i(x^*)(x - x^*) \) supports \( X \) at \( x^* \) for each \( i \in I(x^*) \) and hence \( \nabla g_i(x^*)(x - x^*) \leq 0 \) for all \( x \in X \). Since \( g_i(x^*) = 0 \), then \( g_i^*(x) = g_i(x^*) + \nabla g_i(x^*)(x - x^*) \leq 0 \) for all \( x \in X \). Now consider the set \( X^i = \{ x \mid g_i(x^*) + \nabla g_i(x^*)(x - x^*) \leq 0 \} \). Therefore \( x \in X \) implies \( x \in X^i \) for each \( i \in I(x^*) \), and hence \( x \in X \) implies \( x \in \cap_{i=1}^r X^i = X^* \). Then \( X^* \supset X \), Q.E.D.

The advantage of working with the set \( X^* \) is that the number of constraints is much smaller than the ones for the set \( X \) and hence the computational saving will be substantial. Furthermore, the enclosing set \( X^* \) should be constructed even though the set \( X \) is a polytope. This is due to the fact that we need to consider only active constraints at the point of interest.

The crucial idea needed to characterize generator directions is the concept of extreme directions (Bazaraa and Shetty 1979). Consider the enclosing set \( X^* = \{ Ax = b, x \geq 0 \} \). Let the set of decision variables at an extreme point \( x^* \in X^* \) be partitioned into two subsets, dependent (basic) variables and independent (nonbasic) variables, such that matrix \( A \) is partitioned into two submatrices \( B \) and \( N \) where \( B \) is an \( r \times r \) nonsingular matrix and \( N \) is an \( r \times (n-r) \) matrix. Therefore \( x^* = (y^*, z^*)^t \), where the basic variables \( y^* = (y^*_{-1}, \ldots, y^*_r)^t \).
$y^{*}_{2}, \ldots, y^{*}_{r})^l$ and the nonbasic variables $z^* = (z^{*}_{1}, z^{*}_{2}, \ldots, z^{*}_{n-r})^l$.

Based on this partition, the extreme direction $d_j$ is constructed for each nonbasic variable $z^{*}_{j}$, $j = 1, \ldots, n-r$, and is expressed as follows,

$$d_j = \frac{-B^{-1}N_j}{I_j} \quad \text{for each} \quad j = 1, \ldots, n-r.$$

However in the case that $x^* \in X^*$ is not an extreme point, some of the values of nonbasic variables might be positive and therefore each nonbasic variable with a positive value can be increased or decreased in order to generate a feasible direction in the decision space (a tradeoff in the objective space) (Malakooti 1988b). This implies that for each such nonbasic variable, two directions opposite to each other, should be considered.

Therefore in order to construct the desired set of generators of $C(x^*)$, let $s = n-r$ be the number of nonbasic variables at point $x^* \in X^*$. Furthermore, let the sets $z^{*+} = \{z^{*}_{1}, \ldots, z^{*}_{p}\}$ and $z^{*o} = \{z^{*}_{p+1}, \ldots, z^{*}_{s}\}$ represent the sets of positive and zero nonbasic variables respectively. Then the generators set $\{d^1, d^2, \ldots, d^p, d^{p+1}, \ldots, d^s, d^{s+1}, \ldots, d^q\}$ follows, where $s \leq q \leq 2s$.

$$d_j = \frac{-B^{-1}N_j}{I_j} \quad \text{for each} \quad z^{*}_{j} \in z^*.$$

and

$$d^{s+j} = -d_j \quad \text{for each} \quad z^{*}_{j} \in z^{*+}.$$
Because of the construction of the above directions in the sense that the directions in two sets \( \{d^1, d^2, \ldots, d^p\} \) and \( \{d^{s+1}, \ldots, d^q\} \) are opposite to each other, the cone \( C(x^*) \) can be represented using only the set \( \{d^1, d^2, \ldots, d^p, d^{p+1}, \ldots, d^s\} \). Since any polyhedral cone can be represented by a finite number of generators, Theorem 4.6, next, establishes the fact that the set \( \{d^1, d^2, \ldots, d^s\} \) spans the polyhedral cone \( C(x^*) \), where \( C(x^*) \) is the set of feasible directions of \( X^* \) at \( x^* \in X^* \) that consists of all feasible directions \( d = x - x^*, \ x \in X^* \). Let the set \( X^* \) be in its canonical form (i.e. \( X^* = \{ Ax = b, \ x \geq 0 \} \)). Then \( x^* \in X^* \) can be partitioned into \( y^* \in \mathbb{R}^m \) and \( z^* \in \mathbb{R}^s \) and the matrix \( A \) can be partitioned into matrices \( B \) and \( N \) such that,

\[
By^* + Nz^* = b
\]

where, \( B \) is \( r \times r \) nonsingular matrix. Therefore,

\[
y^* = B^{-1}b - B^{-1}Nz^*.
\]

If \( x^* \) is an extreme point then \( z^* = 0 \), otherwise nonbasic variables \( z^*_j, \ j = 1, \ldots, s \) may have positive values. The proof of Theorem 4.6, next, is given in Appendix I.

**Theorem 4.6**

Let \( X^* = \{ Ax = b, \ x \geq 0 \} \) be a convex polyhedral set. Let \( x^* \in X^* \) be partitioned into \( y^* \in \mathbb{R}^m \) and \( z^* \in \mathbb{R}^s \) such that \( B^{-1} \) exists. Let \( C(x^*) \) be the set of feasible directions of \( X^* \) at \( x^* \in X^* \). Consider the directions \( d^j, \ j = 1, \ldots, s \) that are characterized as follows:
\[
d^j = \begin{vmatrix}
-B^{-1}N_j \\
\ldots \\
I_j
\end{vmatrix}
\text{ for each } z^*_j \in z^*.
\]

Then the set \{d^1, d^2, \ldots, d^s\} spans C(x^*).

Therefore the cone \(C(x^*)\) can be rewritten as follows,
\[
C(x^*) = \{x \mid x = x^* + \sum_{j=1}^{p} \mu_j d^j + \sum_{j=p+1}^{s} \mu_j d^j, \mu_j \geq 0, j = p+1, \ldots, s\},
\]
where the \(\mu_j, j = 1, \ldots, p\) are unrestricted and and therefore any feasible direction \(d \in C(x^*)\) of \(X^*\) emanating from the point \(x^* \in X^*\) can be expressed as follows:
\[
d = \sum_{j=1}^{p} \mu_j d^j + \sum_{j=p+1}^{s} \mu_j d^j, \mu_j \geq 0, j = p+1, \ldots, s;
\]
where, only \(s\)-number of directions are used.

Next, corollary 4.1 establishes that when the original constraint set \(X\) is a convex polytope, the set of generators \{d^1, d^2, \ldots, d^s\} also span the set of feasible directions \(D(x^*)\) at \(x^* \in X\). This is due to the fact that the set \(D(x^*)\) is equivalent to the polyhedral cone \(C(x^*)\), where \(C(x^*)\) is the set of feasible directions of \(X^*\) at \(x^* \in X^*\).

**Corollary 4.1**

Let \(X\) be a convex polytope. Let \(x^* \in X\) be partitioned into \(y^* \in R^m\) and \(z^* \in R^s\) such that \(B^{-1}\) exists. Consider the cone \(C(x^*) = \{x \mid x = x^* + \sum_{j=1}^{p} \mu_j d^j + \sum_{j=p+1}^{s} \mu_j d^j, \mu_j \geq 0, j = p+1, \ldots, s\}\). Let the directions \(d^j, j = 1, \ldots, s\) be characterized as follows:
\[
d^j = \begin{vmatrix} -B^{-1}N_j \\ \ldots \\ I_j \end{vmatrix}
\text{ for each } z^*, \text{ for each } z^*.
\]

Then \{d^1, d^2, \ldots, d^s\} spans D(x*).

**Proof**

Let \( d \in D(x*) \). By construction of the enclosing polyhedral set \( X* \), for each \( d \in D(x*) \) there exist an \( x \in X* \) such that \( d = x - x^* \). Then \( d^* \in C(x*) \) and hence \( C(x*) \supset D(x*) \). Since the set \{d^1, d^2, \ldots, d^s\} spans \( C(x*) \) by theorem 4.6, then this set also spans \( D(x*) \).

Tradeoffs in the objective space plays a central role in the interactive termination criterion. This is due to the fact that they are the only meaningful elements for the interaction with the decision maker. Those tradeoffs are determined with respect to each independent (nonbasic) variables \( z_j \). This can be done using the expression of the reduced gradient for the objective functions at the point \( x^* = (y^*,z^*) \),

\[
t^j = \nabla_{z_j} f(y^*,z^*) - \nabla_y f(y^*,z^*) B^{-1} N_j
\]

Next, we establish the relationship between the above directions and their associated tradeoffs through the utility gradient in the decision space as well as in the objective space. This relation is important to transform DM's tradeoff responses to the decision space where termination is checked.
Lemma 4.2

Let $x^* \in X$ be a convex set. Consider the convex cone $C(x^*) = \{ x \mid x = x^* + \sum_{j=1}^{p} \mu_j d^j + \sum_{j=p+1}^{s} \mu_j d^j, \mu_j \geq 0, j = p+1, \ldots, s \}$. Let the directions $d^j, j = 1, \ldots, s$ be characterized as follows:

$$d^j = \begin{bmatrix} -B^{-1}N_j \\ \hdots \\ I_j \end{bmatrix}$$

for each $z^*_j \in z^*$.

Then $\nabla_x U(f(x^*))d^j = \nabla_T U(f(x^*))t^j$ for $j = 1, \ldots, s$.

Proof

Consider the directions $d^j, j = 1, \ldots, s$. $\nabla_x U(f(x^*))d^j = \nabla_T U(f(x^*)) \nabla_x f(x^*)d^j$ by the chain rule of calculus. Let $C_{y^*} = \nabla_y f(y^*,z^*)$ and $C_{z^*} = \nabla_z f(y^*,z^*)$. Then $\nabla_x U(f(x^*))d^j = \nabla_T U(f(x^*)) \nabla_y,z f(y^*,z^*)d^j = \nabla_T U(f(x^*))[C_{y^*}d^j + I_j C_{z^*}]d^j = \nabla_T U(f(x^*)) [-C_{y^*}B^{-1}N_j + I_j C_{z^*}] = \nabla_T U(f(x^*))t^j, j = 1, \ldots, s$, Q. E. D.

Next, definitions related to the tradeoffs are presented.

Definition 4.3

A tradeoff $t^j$ is nondominated with respect to a set of tradeoffs $T = \{ t^i, i = 1, \ldots, p; i \neq j \}$ if and only if it is not dominated by any linear combination of the tradeoffs of $T$. 
Definition 4.4

Let $x^*$ and $f(x^*)$ be an efficient point and a nondominated alternative respectively. Furthermore, consider the nondominated tradeoff $t^j$ at $f(x^*)$, then

1. $\nabla_f U(f(x^*)) t^j \leq 0$ if $t^j$ is unpreferred (i.e. DM either declines or is indifferent to $t^j$)

2. $\nabla_f U(f(x^*)) t^j < 0$ if the decision maker declines tradeoff $t^j$.

3. $\nabla_f U(f(x^*)) t^j = 0$ if the decision maker is indifferent to tradeoff $t^j$.

Next, lemma 4.3 establishes that quasiconcavity of the utility function $U(.)$ on the objective feasible region $F$ implies its quasiconcavity on the convex constraint set $X$ assuming that every objective function $f_i(x)$ is concave on $X$, $i = 1, \ldots, k$.

Lemma 4.3

Let $X$ be a convex set. Let $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$ and $f_i$ be concave and differentiable on $X$ for each $i = 1, \ldots, m$. If $U(f): \mathbb{R}^m \rightarrow \mathbb{R}$ is a nondecreasing quasiconcave, then $U'(x) = U(f(x))$ is also quasiconcave.

Proof

Let $x, y \in X$. Since each $f_i(x)$ is concave, we have $f_i(x) - f_i(y) \geq \nabla_x f(x)(x-y)$. Putting this in a matrix form, $f(x) - f(y) \geq \nabla_x f(x)(x-y)$. Since $U(.)$ is nondecreasing then $\nabla_f U(f(x)) \geq 0$. Therefore, $\nabla_f U(f(x))(f(x) - f(y)) \geq \nabla_f U(f(x)) \nabla_x f(x)(x-y)$. Now let $\nabla_x U'(x)(x-y) = \ldots$
\[ \nabla f U(f(x)) \nabla_x f(x)(x - y) > 0, \text{ then } \nabla f U(f(x))(f(x) - f(y)) > 0. \] Since \( U(f(x)) \) is quasiconcave with respect to \( f \) then \( U(f(x)) > U(f(y)) \). In other words \( U'(x) > U'(y) \). This, by definition, implies that \( U'(x) \) is quasiconcave on \( X \), Q.E.D.

The interactive termination criterion begins by generating the directions spanning the enclosing cone \( C(x^*) \) at the point of interest, say \( x^* \). Next, their associated tradeoffs are generated which includes efficient as well as inefficient ones. The tradeoff induced by the positive nonbasic variable are all efficient (Malakooti 1988b), where the ones induced by the zero nonbasic variables may have inefficient ones. Therefore, the tradeoffs associated with zero-nonbasic variables are screened for efficiency. This can be accomplished by an efficiency routine developed by Malakooti (1988b). Consequently, only efficient tradeoffs are presented to the decision maker for evaluation, where the inefficient ones are rejected automatically. Furthermore, if a tradeoff associated with one of the positive nonbasic variable is rejected, the tradeoff in the opposite direction, which is also efficient, should be accepted. This stems from the fact for any \( t^j \in \{t^1, \ldots, t^p\} \) that is rejected, \( \nabla U(f)(t^j) < 0 \) and hence \( \nabla U(f)(-t^j) > 0 \) and therefore \( t = -t^j \) should be accepted. Finally, the decision maker's responses to the positive-nonbasic variable tradeoff as well as efficient zero-nonbasic variable tradeoff questions are utilized to check for termination. Interestingly enough, the automatic rejection responses of the inefficient tradeoffs will be effective in declaring
the most-preferred solution. Let the set of positive-nonbasic variable tradeoffs be denoted as \( T^+ \) and the set of zero-nonbasic variable tradeoffs be denoted as \( T^0 \). The following two theorems provides sufficient conditions for declaring the most-preferred solution.

**Theorem 4.7**

Let the utility function \( U'(x) = U(f(x)) \) be quasiconcave on the convex feasible set \( X \) and \( x^* \) be an efficient point. Consider the cone \( C(x^*) = \{ x \mid x = x^* + \sum_{i=1}^{q} \mu_i d^i, q=s+p, \mu_i \geq 0 \} \). If (1) the decision maker is indifferent to every tradeoff of the set \( T^+ = \{ t^1, t^2, \ldots, t^p \} \) and (2) every tradeoff of the set of \( T^0 = \{ t^{p+1}, \ldots, t^s \} \) is unpreferred and there exist an \( i \) such that \( \mu_i > 0 \) with \( d^i \) that has its associated tradeoff \( t^i \) rejected, then \( x^* \) is the most-preferred solution.

**Proof**

Since the decision maker is indifferent to each one of the tradeoffs \( \{ t^1, t^2, \ldots, t^p \} \) then, by definition 4.2, \( \nabla_f U(f(x^*))t^j = 0 \) for all \( j = 1, \ldots, p \) and therefore \( \nabla_x U'(x^*)d^i = \nabla_f U(f(x^*))t^j = 0 \) by lemma 4.2. It is also true that for the opposite directions \( d^{s+j} = -d^j, j = 1, \ldots, p, \nabla_x U'(x^*)d^{s+j} = 0 \). Furthermore, Since each one of the tradeoffs \( \{ t^{p+1}, \ldots, t^s \} \) is unpreferred with one of them being rejected then \( \nabla_f U(f(x^*))t^j \leq 0 \) for all \( j = p+1, \ldots, s \) with one of them \( \nabla_f U(f(x^*))t^j < 0 \) with \( \mu_j > 0 \) and by lemma 4.2, \( \nabla_x U'(x^*)d^j = \nabla_f U(f(x^*))t^j \leq 0 \), with one of them \( \nabla_x U'(x^*)d^j < 0 \) with \( \mu_j > 0 \). Finally, since the set \( \{ d^1, \ldots, d^p, d^{p+1}, \ldots, d^s, d^{s+1}, \ldots, d^{s+p} \} \) span the cone \( C(x^*) \), then \( x^* \) is optimal by theorem 4.3. Hence \( x^* \) is the most-preferred solution, Q.E.D.
Corollary 4.2

Let the utility function \( U(f(x)) \) be quasiconcave on the objective feasible region \( F \) and each \( f_i(x) \) be concave of the convex feasible set \( X \). Let \( x^* \) be an efficient point. If the decision maker is indifferent to every tradeoff of the set \( T^+ = \{ t^1, t^2, \ldots, t^p \} \) and every tradeoff of the set of \( T^0 = \{ t^{p+1}, \ldots, t^s \} \) is unpreferred with at least one of them being rejected, then \( x^* \) is the most-preferred solution.

Proof

Since \( U(f) \) is quasiconcave on \( F \) and each \( f_i(x) \) is concave of the convex feasible set \( X \), then, by lemma 4.3, \( U'(x) = U(f(x)) \) is also quasiconcave on \( X \). Therefore, by theorem 4.7 above, \( x^* \) is the most-preferred solution, Q.E.D.

Theorem 4.8

Let the utility function \( U'(x) = U(f(x)) \) be strictly quasiconcave on the convex feasible set \( X \) and \( x^* \) be an efficient point. If the decision maker is indifferent to every tradeoff of the set \( T^+ = \{ t^1, t^2, \ldots, t^p \} \) and every tradeoff of the set of \( T^0 = \{ t^{p+1}, \ldots, t^s \} \) is unpreferred, then \( x^* \) is the most-preferred solution.

Proof

Since the decision maker is indifferent to each one of the tradeoffs \( \{ t^1, t^2, \ldots, t^p \} \) then, by definition 4.2, \( \nabla_{t^j} U(f(x^*)) t^j = 0 \) for all \( j = 1, \ldots, p \) and therefore \( \nabla_{x} U'(x^*) d^j = \nabla_{t^j} U(f(x^*)) t^j = 0 \) by lemma 4.2. It is also true that for the opposite directions \( d^{s+j} = -d^i, j = 1, \ldots, p \), \( \nabla_{x} U'(x^*) d^{s+j} = 0 \). Furthermore, Since each one of the tradeoffs
\{ t^{p+1}, \ldots, t^p \} is unpreferred, then \( \nabla_f U(f(x^*)) t^j \leq 0 \) for all \( j = p+1, \ldots, s \), and, by lemma 4.2, \( \nabla_x U'(x^*) d^j = \nabla_f U(f(x^*)) t^j \leq 0 \). Finally, since the set \( \{ d^1, \ldots, d^p, d^{p+1}, \ldots, d^s, d^{s+1}, \ldots, d^{s+p} \} \) span the cone \( C(x^*) \), then \( x^* \) is optimal by theorem 4.4. Hence \( x^* \) is the most-preferred solution, Q.E.D.

**Corollary 4.3**

Let the utility function \( U(f(x)) \) be strictly quasiconcave on the objective feasible region \( F \) and each \( f_i(x) \) be concave of the convex feasible set \( X \). Let \( x^* \) be an efficient point. If the decision maker is indifferent to every tradeoff of the set \( T^+ = \{ t^1, t^2, \ldots, t^p \} \) and every tradeoff of the set of \( T^0 = \{ t^{p+1}, \ldots, t^s \} \) is unpreferred with at least one of them being rejected, then \( x^* \) is the most-preferred solution.

**Proof**

Since \( U(f) \) is strictly quasiconcave on \( F \) and each \( f_i(x) \) is concave of the convex feasible set \( X \), then, by a special case of lemma 4.3, \( U'(x) = U(f(x)) \) is also strictly quasiconcave on the set \( X \). Therefore, by theorem 4.8 above, \( x^* \) is the most-preferred solution, Q.E.D.

### 4.6 Examples

Two examples to illustrate the steps of the algorithm of this chapter are in this section.

**4.6.1 Example 4.1 (The Convex Case)**

maximize \[ \{ \ f_1 = x_1, \ f_2 = x_2, \ f_3 = x_3 \ \} \]

s.t. \[ x_1^2 + x_2^2 + x_3^2 \leq 225 \]
\[ x_i \geq 0 \quad \text{for} \quad i = 1,2,3. \]

Assume that the implicit DM's utility function
\[ U(f_1,f_2,f_3) = -(f_1-16)^2-(f_2-16)^2-(f_3-16)^2 \]

Solution: Initialization

By maximizing \( f_i(x) \) s.t. \( x \in X \) for objectives \( f_1, f_2, f_3 \) yield \( f_1^* = 15, f_2^* = 15, f_3^* = 15 \) respectively, then the ideal point \( f^* = (f_1^*, f_2^*, f_3^*) = (15, 15, 15) \). Assuming the \( \varepsilon \) vector is \( (1.0,1.0,1.0) \), the Utopian point \( f^{**} \) is \( (16.0,16.0,16.0) \).

(a): take the nondominated alternative that is associated with the decision maximizing the first objective as the initial alternative. So, \( f^1 = (15,0,0) \). (b) Set \( i = 1 \).

Iteration(1)

Step(1): using the implicit utility function, \( \nabla U(f) = 2(1,16,16)' \), the direction finding problem will be:
\[ \max y_1 + 16y_2 + 16y_3 \quad \text{s.t.} \quad y \in X. \]
Using GINO (Liebman et al. 1986), \( y^1 = (0.66, 10.60, 10.59)' = f(y^1) \).

Step(2): in order to generate the Tchebycheff weights, we generate three \( (M_1 = 3 \text{ arbitrary taken as the number of the objectives}) \) convex combination points of \( f(x^1) \) and \( f(y^1) \). These are given in Table 4.2 next,
Table 4.2

<table>
<thead>
<tr>
<th>Convex combinations of ( f(x^1) ) and ( f(y^1) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(11.42, 2.65, 2.65)</td>
</tr>
<tr>
<td>(7.83 , 5.30 , 5.30)</td>
</tr>
<tr>
<td>(4.25, 7.95, 7.44)</td>
</tr>
</tbody>
</table>

Using these points generated in objective space, the associated weights are calculated using equation (3). So,

\[
   w_i = \frac{(1/(f_i^{**}-f_i'))}{(\Sigma_{i=1}^{3}(1/(f_i^{**}-f_i')))}, \text{ for } i = 1, 2, 3;
\]

where \( f_i^{**} \) and \( f_i' \) are the \( i \)-th level of the Utopia point and generated point on the line respectively. For example the weights associated with the point (11.42, 2.65, 2.65) are obtained as follows, using the reference point (16,16,16),

\[
   w_1 = \frac{(1/(16-11.42))}{(1/(16-11.42)+1/(16-2.65)+(1/16-2.65))} = 0.5928,
\]

\[
   w_2 = \frac{(1/(16-2.65))}{(1/(16-11.42)+1/(16-2.65)+(1/16-2.65))} = 0.2036,
\]

\[
   w_3 = \frac{(1/(16-2.65))}{(1/(16-11.42)+1/(16-2.65)+(1/16-2.65))} = 0.2036.
\]

Notice that the two end points (15.00, 0.00, 0.00) and (0.66, 10.60, 10.59) are both nondominated, therefore their associated Tchebycheff weights are not needed.

As an example, in order to determine point "A" in Table 4.3, we solved the following program:

\[
   \text{Minimize} \quad \alpha + 0.01 \Sigma_{i=1}^{3} w_i (16 - f_i)
\]
st. \( w_i(16 - f_i) \leq \alpha \quad i = 1, 2, 3. \)
\[ f_i = x_i \quad i = 1, 2, 3. \]
\[ x_1^2 + x_2^2 + x_3^2 \leq 225 \]
\[ x_i \geq 0, \]

where \( w = (0.5928, 0.2036, 0.2036). \)

For this iteration, we have the following table that contains five nondominated alternatives denoted as \( f(x^1), A, B, C \) and \( f(y^1) \), These are given in Table 4.3 next.

Table 4.3: Projection results in the first iteration.

<table>
<thead>
<tr>
<th>generated point</th>
<th>projection point</th>
<th>utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>(15.00,0.00,0.00)</td>
<td>(15.00,0.00,0.00)</td>
<td>-513.00 ( f(x^1) )</td>
</tr>
<tr>
<td>(11.42,2.65,2.65)</td>
<td>(12.51,5.89,5.89)</td>
<td>-216.60 ( A )</td>
</tr>
<tr>
<td>(7.83,5.30,5.30)</td>
<td>(9.88,7.98,7.98)</td>
<td>-166.10 ( B^* )</td>
</tr>
<tr>
<td>(4.25,7.95,7.95)</td>
<td>(6.57,9.54,9.53)</td>
<td>-172.52 ( C )</td>
</tr>
<tr>
<td>(0.66,10.60,10.59)</td>
<td>(0.66,10.60,10.59)</td>
<td>-294.07 ( f(y^1) )</td>
</tr>
</tbody>
</table>

Step(3): The five nondominated alternatives, as in the above table (the three generated by the Tchebycheff scheme and the two nondominated end points of the direction line) are evaluated by the simulated DM, where alternative "B" is the most-preferred. Since this alternative is not the same as the current solution, i.e. \( f(x^1) \), the second iteration is started with the alternative "B" as its starting solution.
Iteration(2)

By following the same process as before, two nondominated alternatives are generated (assuming \( M^2 = 2 \)). These alternatives together with the end points of the direction line will make a set of four alternatives denoted as \( f(x^2) \), D, E and \( f(y^2) \) as shown by Table 4.4 next.

Table 4.4: Projection results in the second iteration.

<table>
<thead>
<tr>
<th>generated point</th>
<th>projected point</th>
<th>utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>(9.88, 7.98, 7.98)</td>
<td>(9.88, 7.98, 7.98)</td>
<td>-166.10 ( f(x^2) )</td>
</tr>
<tr>
<td>(8.96, 8.43, 8.43)</td>
<td>(9.01, 8.48, 8.48)</td>
<td>-161.96 ( D^* )</td>
</tr>
<tr>
<td>(8.04, 8.88, 8.88)</td>
<td>(8.10, 8.93, 8.93)</td>
<td>-162.38 ( E )</td>
</tr>
<tr>
<td>(7.12, 9.33, 9.33)</td>
<td>(7.12, 9.33, 9.33)</td>
<td>-167.83 ( f(y^2) )</td>
</tr>
</tbody>
</table>

Those four alternatives are evaluated by the simulated DM, where alternative "D" is the most-preferred. Since this alternative is not the same as the starting solution, the third iteration begins with alternative "D" as the starting point.

Iteration(3)

Again using the same process, one nondominated alternative (i.e. \( M^3 = 1 \)) is generated using one point on the direction line. This alternative together with the two alternatives associated with the
end points make a set of three nondominated alternatives denoted as \( f(x^3) \), \( F \) and \( f(y^3) \) as shown by Table 4.5 next,

Table 4.5: Projection results in the third iteration.

<table>
<thead>
<tr>
<th>generated point</th>
<th>projected point</th>
<th>utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>(9.01,8.48,8.48)</td>
<td>(9.01,8.48,8.48)</td>
<td>-161.96 f(x^3)</td>
</tr>
<tr>
<td>(8.63,8.67,8.67)</td>
<td>(8.63,8.67,8.67)</td>
<td>-161.77 F *</td>
</tr>
<tr>
<td>(8.25,8.86,8.86)</td>
<td>(8.25,8.86,8.86)</td>
<td>-162.02 f(y^3)</td>
</tr>
</tbody>
</table>

Evaluating those alternatives by the simulated DM, alternative "F" is the most preferred, which not is the same as the current solution (i.e. \( f(x^3) \)).

**Iteration (4)**

Repeating the process again using \( f(x^4) = (8.63, 8.67, 8.67) \) as the starting point, the direction finding problem yields the same point. This implies that this point is a candidate for checking the optimality condition.

**Interactive Termination**

(a) Constructing the enclosing set \( X^* \)

In this example, \( x^* = (8.63, 8.67, 8.67) \) and \( g_1(x) = x_1^2 + x_2^2 + x_3^2 - 225. \)

Since \( \nabla g_1(x) = (2x_1, 2x_2, 2x_3) \), then \( \nabla g_1(x^*) = (17.26, 17.34, 17.34) \). Therefore, the linear approximation of \( g_1(x) \) around the point \( x^* \) is as follows,
\[ g^*_{i}(x) = g_i(x^*) + \nabla g_i(x^*)(x - x^*) \]
\[ g^*_{i}(x) = g_i(x^*) + (17.26, 17.34, 17.34)(x - x^*) \]
\[ g^*_{i}(x) = 0.0 + 17.26x_1 + 17.34x_2 + 17.34x_3 - 449.63 \]
\[ g^*_{i}(x) = 17.26x_1 + 17.34x_2 + 17.34x_3 - 449.63 \]

Then the enclosing set \( X^* \),
\[ X^* = \{ x \in \mathbb{R}^3 \mid 17.26x_1 + 17.34x_2 + 17.34x_3 \leq 449.63, \ x_i \geq 0.0, \ i = 1, \ldots, 4 \} \]

Putting it in the canonical form,
\[ X^* = \{ x \in \mathbb{R}^3 \mid 17.26x_1 + 17.34x_2 + 17.34x_3 + x_4 = 449.63, \ x_i \geq 0.0, \ i = 1, 2, 3 \} \]
and therefore \( A = [17.26 \ 17.34 \ 17.34 \ 1.0] \) and \( b = [449.63] \). Since the number of active constraints at point \( x^*(r) \) is 1, let the variables at point \( x^* \) be partitioned such that \( y^* = [x_1] \) and \( z^* = [x_2 \ x_3 \ x_4] \)
with \( B = [17.26] \) and \( N = [17.34 \ 17.34 \ 1.0] \). Since \( x^* = (8.63, 8.67, 8.67, 0.0) \), if we use \( z^* = (8.67, 8.67, 0.0) \) then
\[ y^* = B^{-1}b - B^{-1}Nz^*. \]
\[ y^* = [1/17.26][449.63] - [17.34 \ 17.34 \ 1.0][x_2 \ x_3 \ x_4]^t \]
\[ y^* = [8.63]. \]

This implies that in the above construction of \( B \) and \( N \), the nonbasic variables \( x_2 \) and \( x_3 \) have positive values, and therefore two extreme directions are associated with each one of them. Using the expression of the extreme directions as in subsection 4.5.4,
\[ d_j = \begin{vmatrix} -B^{-1}N_j \\ \vdots \\ I_j \end{vmatrix} \quad \text{for each } z_{*j} \in z^*. \]
where \( z^* = [z_{*1}, z_{*2}, z_{*3}] = [x_2 \ x_3 \ x_4] \). Then, for the nonbasic variable \( x_2 \), we have
\[ d^1 = \begin{bmatrix} -[1/17.26][17.34] \\ 1 \\ 0 \\ 0 \end{bmatrix} \]

Then \( d^1 = [-1.00 \ 1.0 \ 0.0 \ 0.0]^t \). Using similar calculations, we have:
\( d^2 = [-1.00 \ 0.0 \ 1.0 \ 0.0]^t \) and \( d^3 = [-0.06 \ 0.0 \ 0.0 \ 1.0]^t \).

Therefore, the generators of the cone of feasible directions of \( X^* \) at \( x^* \) are given as follows,
\[
\begin{align*}
  d^1 &= [-1.00 \ 1.0 \ 0.0]^t \\
  d^2 &= [-1.00 \ 0.0 \ 1.0]^t \\
  d^3 &= [-0.06 \ 0.0 \ 0.0]^t
\end{align*}
\]

Hence the cone of feasible directions \( C(x^*) \) of \( X^* \) at \( x^* \) is follows,
\[
C(x^*) = \{ x \mid x = x^* + \sum_{j=1}^{2} \mu_j d^j + \mu_3 d^3, \ \mu_3 \geq 0 \}.
\]

The next task is determine the associated tradeoffs. The following expression is used:
\[
t^j = \nabla_{z_j} f(y^*, z^*) - \nabla_y f(y^*, z^*) B^{-1} N_j
\]

We have \( \nabla_y f(y^*, z^*) = [1.0 \ 0.0 \ 0.0] \) and
\[
\nabla_{z_j} f(y^*, z^*) = \begin{bmatrix} 0.0 & 0.0 & 0.0 \\ 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \end{bmatrix}
\]

Then,
the tradeoff associated with \( d^1 \),
\[
t^1 = [0.0 \ 1.0 \ 0.0]^t - [1.0 \ 0.0 \ 0.0]^t[1/17.26][17.34]
\]
\[
\mathbf{t}^2 = [0.0 \ 0.0 \ 1.0]^t - [1.0 \ 0.0 \ 0.0]^t [1/17.26][17.34] = [-1.00 \ 0.00 \ 1.00]^t.
\]

the tradeoff associated with \( d^2 \),

\[
\mathbf{t}^3 = [0.0 \ 0.0 \ 0.0]^t - [1.0 \ 0.0 \ 0.0]^t [1/17.26][1.0] = [-0.06 \ 0.00 \ 0.00]^t.
\]

Now, having generated the desired tradeoffs, the next task is to screen them for efficiency. Tradeoff \( t^3 \) is clearly the only inefficient tradeoff, and therefore it is considered to be rejected by the decision maker. Hence, only tradeoffs \( t^1 \) and \( t^2 \) are presented to the DM for evaluation. Let use the assumed utility function to simulate the responses of the DM, where \( \nabla_f U(f(x^*)) = 2(7.37, 7.33, 7.33) \).

for tradeoff \( t^1 \), we have

\[
\nabla_f U(f(x^*))|t^1 = 2[7.37, 7.33, 7.33][1.0 \ 0.0 \ 0.0]^t = 0,
\]

and therefore the decision maker is indifferent with respect to \( t^1 \).

for tradeoff \( t^2 \), we have

\[
\nabla_f U(f(x^*))|t^2 = 2[7.37, 7.33, 7.33][1.0 \ 0.0 \ 1.0]^t = 0.
\]

and therefore the decision maker is indifferent with respect to \( t^2 \).

Since the tradeoffs \( t^1 \) and \( t^2 \) are associated with positive nonbasic variables \( x_2 \) and \( x_3 \) respectively, then the optimality condition is satisfied. Therefore, the solution process is concluded with the most-preferred alternative as \((8.63, 8.67, 8.67)\) associated with \((8.63, 8.67, 8.67)\) as the most-preferred decision for the whole problem, assuming the utility function \( U(.) \) is quasiconcave.
In order to check how close we are to the DM's true best-compromise alternative based on the given utility function, this function was optimized explicitly. The resulted optimal alternative was found to be (8.66, 8.66, 8.66) with a utility value of -161.62. However, our resulted final alternative was (8.63, 8.67, 8.67) with a utility value of -161.77, which is 99.99% of the true one.

If, on the other hand, the deviation-based programming formulation is used, point (11.42, 2.65, 2.65) is projected using the following formulation,

\[
\begin{align*}
\text{maximize } & \quad w^\top y^- \\
\text{s.t. } & \quad x_1 - y_1^- = 11.42 \\
& \quad x_2 - y_2^- = 2.65 \\
& \quad x_3 - y_3^- = 2.65 \\
& \quad x \in X, \ x \geq 0, \\
& \quad y_i^- \geq 0, \quad i = 1, 2, 3.
\end{align*}
\]

Alternatives generated in iteration (1) are given in Table 4.6 next,

Table 4.6: Projection results in the first iteration.

<table>
<thead>
<tr>
<th>generated point</th>
<th>projected point</th>
<th>utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>(15.00, 0.00, 0.00)</td>
<td>(15.00, 0.00, 0.00)</td>
<td>-513.00 f(x^1)</td>
</tr>
<tr>
<td>(11.42, 2.65, 2.65)</td>
<td>(11.42, 6.88, 6.88)</td>
<td>-187.33 A'</td>
</tr>
<tr>
<td>(7.83, 5.30, 5.30)</td>
<td>(8.66, 8.66, 8.66)</td>
<td>-161.62 B' *</td>
</tr>
<tr>
<td>(4.25, 7.95, 7.95)</td>
<td>(8.66, 8.66, 8.66)</td>
<td>-161.62 B' *</td>
</tr>
<tr>
<td>(0.66, 10.60, 10.59)</td>
<td>(0.66, 10.60, 10.59)</td>
<td>-294.07 f(y^1)</td>
</tr>
</tbody>
</table>
By evaluating these alternatives, alternative B' is the best of the sample. Hence, this solution (i.e. B') is taken to be the current solution of the second iteration. When solving the direction finding problem, point B' is still the solution. If optimality condition is assumed to be satisfied at alternative B', it will be the best-compromise solution. In fact, solution B' is the true solution based on GINO's result. Furthermore, the solution process is concluded just after one iteration, which we believe it is a concidence.

4.6.2 Example 4.2 (The Nonconvex Case)

maximize \( f_1 = x_1 , f_2 = x_2 , f_3 = x_3 \)

s.t. \( x_1^2 + x_2^2 - 900 \exp(-x_3) \leq 0.0 \)

\( x_3 \leq 1.0 , x_i \geq 0.0 \) for \( i = 1,2,3 \).

Assume that the implicit DM's utility function is

\( U(f_1,f_2,f_3) = -0.2(f_1-28)^2 -0.3(f_2-28)^2 -10.0(f_3-5.0)^2. \)

Solution

The first iteration gives the following sample in Table 4.7 next,

Table 4.7: Projection results in the first iteration.

<table>
<thead>
<tr>
<th>generated point</th>
<th>projected point</th>
<th>utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>(21.21,21.21,0.00)</td>
<td>(21.21,21.21,0.00)</td>
<td>-273.03 f(x^1)</td>
</tr>
<tr>
<td>(17.51,19.19,0.33)</td>
<td>(15.81,19.54,0.35)</td>
<td>-267.08 A*</td>
</tr>
<tr>
<td>(13.80,17.17,0.67)</td>
<td>(11.65,17.85,0.68)</td>
<td>-270.72 B</td>
</tr>
<tr>
<td>(10.09,15.14,1.00)</td>
<td>(10.09,15.14,1.00)</td>
<td>-273.75 f(y^1)</td>
</tr>
</tbody>
</table>
The best alternative from the above sample is "A", since it is not the same as \( f(x^1) \), steps (1)-(2) are executed again with alternative "A" as the initial point. The second iteration gives the following sample in Table 4.8 next.

Table 4.8: Projection results in the second iteration.

<table>
<thead>
<tr>
<th>generated point</th>
<th>projected point</th>
<th>utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>(15.80,19.54,0.35)</td>
<td>(15.80,19.54,0.35)</td>
<td>-267.08 ( f(x^2) )*</td>
</tr>
<tr>
<td>(14.21,16.33,0.67)</td>
<td>(13.19,16.64,0.69)</td>
<td>-268.266 C</td>
</tr>
<tr>
<td>(12.61,13.12,1.00)</td>
<td>(12.61,13.12,1.00)</td>
<td>-273.82 ( f(y^2) )</td>
</tr>
</tbody>
</table>

In the above sample, \( f(x^2) \) is the best alternative. Since no other better alternative could be found, the algorithm is concluded with the point \( (15.801,19.542,0.354) \) as the best decision as well as the best outcome for the problem with a utility value of -267.08. To compare this solution with GINO's, the solution \( (15.748,18.437,0.426) \) with a utility of -266.69. This is a 99.85% of the true utility value. The interactive termination criterion is not applicable here since the constraint set is nonconvex.

4.7 Computational Experiments

Computational experiments can address some performance aspects of the algorithm such as total alternatives and tradeoffs presented to the decision maker, total iterations, and the deviation from the true solution for each simulated decision situation.
Therefore, in order to test the developed algorithm, thirteen different problems are arbitrarily selected from the literature. These problems have up to 5 objectives, 8 decision variables (slack and surplus variables are not included), and 10 constraints (nonnegativity constraints are not included), and are chosen for their computational feasibility. The simulated utility function for all of these problems has the following general form:

$$U(f) = -(f_1 - f_1^*)^2 - (f_2 - f_2^*)^2 - \ldots - (f_k - f_k^*)^2,$$

where $f_i^*$ is the constrained maximum for objective function $f_i$. The simulations were conducted via an off-line strategy using GINO and a small routine GENERATE_EVALUATE written in the TURBO PASCAL programming language. GINO was used to (1) generate an initial nondominated alternative, (2) solve the direction-finding problem, and (3) solve the projection problem. However, the GENERATE_EVALUATE routine was used to generate alternatives in one-dimensional line sampling and to simulate the DM response towards a set of nondominated alternatives.

The initial nondominated alternative for each simulation was obtained by maximizing a reference objective function. Objective function $f_1$ was used as a reference objective in all of the simulations except in one case, where objective function $f_2$ was used instead. The choice of constructing the initial alternative in this way made it possible to observe the behavior of the algorithm starting from one of the extremes of nondominated frontier. The gradient vector of the simulated utility, which is analytically found, was used to determine
the gradient value needed for iteration. Furthermore, the Tchebycheff-based scheme was used to accomplish the projection.

Summary of the experimental observations are presented in Table 4.1. Table 4.1's entries include,

1. Problem size: \( k \times n \times m \), where \( k \) is the number of objective functions, \( n \) is the number of decision variables (slack and surplus variables are not included), and \( m \) is the number of constraints (nonnegativity constraints are not included).

2. Number of alternatives in the first and the second iteration: includes total number of alternatives presented to the decision Table maker during the first and the second iteration, where number of alternatives generated varied for some problem simulations.

3. Total number of iterations: total number of iterations gives also the number of gradient assessments needed as well as direction-finding problems that need to be solved.

4. Total number of alternatives / Total number of tradeoffs: Total number of alternatives gives total number of alternatives generated during the whole solution process, in which the decision maker was faced with a set of preference comparison question. Each question requires the DM to identify the best alternative among a set of nondominated alternatives. The total number of alternatives in each set was kept at a maximum of seven. However, the total number of tradeoffs gives the number of efficient tradeoffs presented to the DM for evaluation during the execution of the interactive termination criterion.
4.1: Experiments for the quasiconcave utility-based algorithm.

<table>
<thead>
<tr>
<th>Problem #</th>
<th>Problem Size</th>
<th>Altern. in iterations 1/2</th>
<th>Total iterations</th>
<th>Total Alter. / Tradeoffs</th>
<th>% Δ U</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2 x 2 x 1</td>
<td>5/4</td>
<td>2</td>
<td>9 / 1</td>
<td>0.0 %</td>
</tr>
<tr>
<td>2</td>
<td>2 x 3 x 1</td>
<td>5/4</td>
<td>2</td>
<td>9 / 1</td>
<td>0.0 %</td>
</tr>
<tr>
<td>3</td>
<td>2 x 2 x 2</td>
<td>5/4</td>
<td>2</td>
<td>9 / 1</td>
<td>4.0 %</td>
</tr>
<tr>
<td>4</td>
<td>2 x 2 x 3</td>
<td>0/-</td>
<td>1</td>
<td>0 / 1</td>
<td>0.0 %</td>
</tr>
<tr>
<td>4</td>
<td>2 x 2 x 3</td>
<td>5/4</td>
<td>3</td>
<td>9 / 1</td>
<td>0.5 %</td>
</tr>
<tr>
<td>5</td>
<td>3 x 4 x 2</td>
<td>5/0</td>
<td>2</td>
<td>5 / 2</td>
<td>2.7 %</td>
</tr>
<tr>
<td>5</td>
<td>3 x 4 x 2</td>
<td>10/5</td>
<td>3</td>
<td>20 / 2</td>
<td>1.1 %</td>
</tr>
<tr>
<td>6</td>
<td>3 x 3 x 3</td>
<td>5/4</td>
<td>4</td>
<td>19 / 2</td>
<td>4.5 %</td>
</tr>
<tr>
<td>7</td>
<td>3 x 3 x 3</td>
<td>5/-</td>
<td>1</td>
<td>5 / 2</td>
<td>7.1 %</td>
</tr>
<tr>
<td>7</td>
<td>3 x 3 x 3</td>
<td>12/5</td>
<td>2</td>
<td>17 / 2</td>
<td>0.7 %</td>
</tr>
<tr>
<td>8</td>
<td>3 x 7 x 4</td>
<td>5/4</td>
<td>2</td>
<td>9 / 2</td>
<td>3.5 %</td>
</tr>
<tr>
<td>9</td>
<td>3 x 8 x 4</td>
<td>5/4</td>
<td>2</td>
<td>9 / 2</td>
<td>3.1 %</td>
</tr>
<tr>
<td>9</td>
<td>3 x 8 x 4</td>
<td>12/5</td>
<td>2</td>
<td>19 / 2</td>
<td>0.0 %</td>
</tr>
<tr>
<td>10</td>
<td>3 x 8 x 8</td>
<td>5/4</td>
<td>3</td>
<td>13 / 2</td>
<td>1.1 %</td>
</tr>
<tr>
<td>10</td>
<td>3 x 8 x 8</td>
<td>10/5</td>
<td>3</td>
<td>19 / 2</td>
<td>1.1 %</td>
</tr>
<tr>
<td>11</td>
<td>3 x 6 x 10</td>
<td>5/5</td>
<td>4</td>
<td>15 / 2</td>
<td>0.1 %</td>
</tr>
<tr>
<td>12</td>
<td>5 x 4 x 7</td>
<td>5/-</td>
<td>1</td>
<td>5 / 4</td>
<td>0.1 %</td>
</tr>
<tr>
<td>13</td>
<td>5 x 8 x 8</td>
<td>5/5</td>
<td>2</td>
<td>10 / 4</td>
<td>3.9 %</td>
</tr>
<tr>
<td>13</td>
<td>5 x 8 x 8</td>
<td>12/5</td>
<td>2</td>
<td>17 / 4</td>
<td>1.5 %</td>
</tr>
</tbody>
</table>
5. Percentage utility deviation: using the utility value of the final alternative given by our algorithm and the utility value of the exact algorithm given by GINO, the percentage deviation in utility deviation is calculated according to the following formula:

\[
\% \Delta U = \frac{U(f^{\text{GINO}}) - U(f^{\text{final}})}{U(f^{\text{GINO}})} \times 100\%.
\]


The results showed that generating more alternatives (by increasing the value of M) in the first and the second iterations enhances the final solution of the algorithm since the value of percentage utility deviation (\% \Delta U) is reduced, except in test problem #3, where increasing the value of M did not improve the final alternative. The enhancement is due to the fact that the original F-W algorithm, which our algorithm is based on, has a fast initial rate of convergence. The exception of test problem #3 is due to the need of sampling many alternatives for the one-dimensional search. Figures 4.3 and 4.4 presents the nondominated frontier of test problems #2 and #3 respectively. The results also showed that the number of the tradeoffs presented to the DM increases as the number of objectives increases. Furthermore, the results showed that the value of percentage utility deviation (\% \Delta U) is higher with problem sizes.
Figure 4.3: Nondominated Frontier for Test Problem #2.

Figure 4.4: Nondominated Frontier for Test Problem #3
4.8 Conclusions

In this chapter, an interactive Gradient projection-based multiobjective algorithm to handle quasiconcave utility functions is presented. The algorithm includes utility gradient assessment as well as projection of the gradient-based improvement direction on the nondominated frontier. It also includes an interactive termination criterion to give the decision maker control over the conclusion of the algorithm. Procedures for implementing the process of projection and interactive termination are developed. Computational experiments showed that the algorithm converge within just a few iterations. Their results showed that generating more alternatives in early iterations enhances the quality of the final solution of the algorithm with respect to its utility value. The results also showed that the number of the tradeoffs presented to the DM increases as the number of the objective functions increases. Furthermore, the results showed that the value of percentage utility deviation (% ΔU) is higher with problem sizes.
5. The Quasiconvex Utility-based Multiobjective Approach

5.1 Introduction

Global nonconcave maximization is considered to be a difficult optimization problems because there may exist many local optimal points and there is no criterion to conclude that a given local optimal point is the global solution. However, research attempts have been reported concerning a more tractable subclass referred to as global convex maximization (Heising 1981; Pardalos and Rosen 1986). Even for problems with special structure such as maximizing a quadratic convex function over the unit hypercube, they are considered to be an NP-hard problem (Pardalos and Rosen 1986). Nevertheless, all of research efforts surveyed in the above two references attempted to tackle the special structure of maximizing a convex objective function over a convex constraint set, mostly a polyhedral set. The appealing feature of this particular structure is that the global maximizer always occur at an extreme point.

Interestingly enough, two research attempts have been found to handle a more relaxed form of the objective function and constraints. Sniedovich (1986) proposed an algorithm for optimizing a pseudolinear objective function, which is both pseudoconcave and pseudoconvex, over some constraint set. Its fundamental concept is based on a relationship established between the original problem and an auxiliary problem that is easier to solve. Despite the author's claim about the convergence of the method, no computational
experiments have been reported. Furthermore, Katoh and Ibaraki (1987) developed an $\epsilon$-approximation algorithm based on an extension of the Sniedovich's relationship to the class of quasiconvex functions. Their algorithm depends very heavily on the explicit form of the objective function.

In this chapter, we assume that the decision maker's underlying preference is represented by a quasiconvex utility function. It was not, until recently, that this class of functions is suggested to be used as a viable assumption (Malakooti 1990). The possibility that some decision makers might behave with an underlying quasiconvex utility function can be supported, as was shown in Chapter 3, using the additive as well as multiplicative utility/value models of Keeney and Raiffa (1976). Also, recently Steuer (1990) emphasized the need for considering this type of utility functions. Furthermore, convex as well as quasiconvex single utility functions have been used (Duckstein, Bobee and Ashkar 1990; Anandalingam and Olsson 1990).

This chapter consists of five sections. In Section 5.2, Basic definition and theories are presented. In section 5.3, quasiconvex utility-based multiobjective linear programming is discussed. It includes basic theory and an interactive algorithm together with its implementation and computational experiments. Quasiconvex utility-based multiobjective nonlinear programming is discussed in Section 5.4, where an interactive heuristic algorithm is included. Conclusions are presented in Section 5.5.
5.2 Basic Definition and Theories

In this section, basic definition and theorems are presented. First, the definition of the quasiconvex functions as well as a related theorem from the literature are presented.

**Definition** (Bazaraa and Shetty 1979)

Let \( U:V\to R \), where \( V \) is a nonempty convex set in \( R^k \). The function \( U(,.) \) is said to be quasiconvex if, for each \( f^1 \) and \( f^2 \in V \), the following inequality is true:

\[
U(\alpha f^1 + (1-\alpha)f^2) \leq \max \{U(f^1), U(f^2)\} \quad \text{for each } \alpha \in (0,1).
\]

**Theorem** (Bazaraa and Shetty 1979)

Let \( V \) be a nonempty open convex set in \( R^k \), and let \( U: V\to R \) be differentiable on \( V \). Then \( U(f) \) is quasiconvex if and only if either one of the following equivalent statements hold,

1. If \( f^1, f^2 \in V \) and \( U(f^1) \leq U(f^2) \), then \( \nabla^T U(f^2)(f^1-f^2) \leq 0 \).
2. If \( f^1, f^2 \in V \) and \( \nabla U(f^2)(f^1-f^2) > 0 \), then \( U(f^1) > U(f^2) \).

Next, it is shown that when the utility function \( U(f) \) is quasiconvex with respect to every objective function \( f_i(x) \) and each objective function \( f_i(x) \) is convex on the set \( X \), the utility function \( U'(x) = U(f(x)) \) is also quasiconvex on the feasible set \( X \).

**Theorem 5.1**

Let \( X \) be a convex set. Let \( f: R^n \to R^m \) and \( f_i(x) \) be convex and differentiable on \( X \) for each \( i = 1, \ldots, m \). If \( U(f): R^m \to R \) is
nondecreasing quasiconvex, then \( U'(x) = U(f(x)) \) is also quasiconvex on \( X \).

**Proof**

Let \( x, y \in X \). Since each \( f_i(x) \) is convex, \( f_i(y) - f_i(x) \geq \nabla_x f(x)(y-x) \). Putting this in a matrix form, \( f(y) - f(x) \geq \nabla_x f(x)(y-x) \). Since \( U(.) \) is nondecreasing then \( \nabla f U(f(x)) \geq 0 \). Therefore, \( \nabla f U(f(x))(f(y) - f(x)) \geq \nabla f U(f(x))\nabla x f(x)(y-x) \). Now let \( \nabla x U'(x)(y-x) = \nabla f U(f(x))\nabla x f(x)(y-x) > 0 \), then \( \nabla f U(f(x))(f(y) - f(x)) > 0 \). Since \( U(f(x)) \) is quasiconvex with respect to \( f \) then \( U(f(y)) > U(f(x)) \). In other words \( U'(y) > U'(x) \). This implies that \( U'(x) \) is quasiconvex on \( X \), Q.E.D.

Interestingly, the underlying structure of the multiobjective optimization model plays a vital role in the development of interactive algorithms for quasiconvex utility functions. This stems from the fact that in the case of multiobjective linear programming, as will be shown latter in Section 5.2, the set of the most-preferred solutions should contain an extreme point. This fact will confine the search process to consider only a subset of the feasible region. However, the search process will consider only the set of proper and supported efficient points in the case of multiobjective nonlinear programming, assuming the utility function satisfies additional mild properties, to be defined in Section 5.3.

In the next section, theories and an interactive algorithm to handle the case of multiobjective linear programming will be presented. Theories and an interactive heuristic algorithm for the
multiobjective nonlinear programming will be presented in Section 5.4.

5.3 Multiobjective Linear Programming

The multiobjective linear programming (MOLP) problems represent an important class of multiobjective optimization models due to the wide spread use of LP models in practice. The structure of the model under consideration is as follows:

Program 5.1: maximize $c_1^t x, \ldots, c_k^t x$

subject to $x \in X = \{ Ax \leq b \}$.

where, $x \in \mathbb{R}^n$, $c_i$ is $k \times 1$ vector for all $i$, $A$ is $m \times n$ matrix and $b$ is $m \times 1$ vector and $X$ is a nonempty compact set that includes nonnegativity constraints.

Basic theory is provided in the next subsection. Then, an interactive algorithm is presented in subsection 5.3.2. Illustrative example is presented in subsection 5.3.3. In subsection 5.3.4, implementation and computational experiments are discussed.

5.3.1 Basic Theory

Due to the structure of the MOLP feasible region, two theorems that characterize the set of the most-preferred solutions are provided, assuming the general case that the MOLP problem has a set of most-preferred solutions for the quasiconvex and strictly quasiconvex utility functions respectively. Therefore, the set of most-preferred solutions and the set of most-preferred alternatives are denoted $X^*$ and $F^*$ respectively.
Theorem 5.2

Let the feasible region $X$ be a convex polyhedron and the utility function $U'(x) = U(f(x))$ be a quasiconvex on $X$. The most-preferred solution set $X^*$ includes an efficient extreme point.

Proof (by contradiction)

First we show that $X^*$ includes an extreme point. So, assume that the set $X^*$ does not contain any extreme points. Let $x^* \in X^*$ be an arbitrary most-preferred solution, this implies that $U'(x^*) > U'(x)$ for all extreme points $x \in X$.

Since $x^*$ is not an extreme point, then it could be expressed in terms of all extreme points (say "p" points) of the feasible region $X$, this due to the representation theorem of the Linear Programming (Bazaraa and Shetty 1979, Theorem 2.5.7, pp. 60). Therefore, there exist multipliers $\mu_i \geq 0$, $\Sigma_{i=1}^{p} \mu_i = 1$, such that $x^* = \Sigma_{i=1}^{p} \mu_i x^i$.

Let's assume that $U(x') = \max \{U'(x^1), \ldots, U'(x^p)\}$. Since $U'(x)$ is quasiconvex, then $U'(x^*) = U'(\Sigma_{i=1}^{p} \mu_i x^i) \leq U'(x')$ by definition. This is a contradiction, since $x'$ is an extreme point. Therefore, the solution set $X^*$ must contain an extreme alternative.

Next, we show that such an extreme point is efficient by contradiction also. So, let the extreme point $x^* \in X^*$ be inefficient. Then, by the definition of efficiency, there exists another point $x' \in X$ such that $f(x') \geq f(x*)$. This implies that $U'(x) = U(f(x')) > U(f(x*)) = U'(x*)$, by the monotonicity assumption. Hence, $x^* \notin X^*$ which is a contradiction. Therefore, the extreme point $x^* \in X^*$ is efficient, Q.E.D.
Theorem 5.3

Let the feasible region $X$ be a convex polyhedron and the utility function $U'(x) = U(f(x))$ be a strictly quasiconvex on $X$. If $x^*$ is a most-preferred solution, then it is an efficient extreme point.

Proof (by contradiction)

First we show that $x^*$ is an extreme point. So, assume that $x^*$ is a most-preferred solution but it is not an extreme point. Now, since $x^*$ is a most-preferred solution, $U'(x^*) \geq U'(x)$ for all $x \in X$.

Since $x^*$ is not an extreme point, then it could be expressed in terms of all extreme points (say "p" points) of the feasible region $X$, this due to the representation theorem of the Linear Programming (Bazaraa and Shetty 1979, Theorem 2.5.7, pp. 60). Therefore, there exist multipliers $\mu_i \geq 0, \sum_{i=1}^{p} \mu_i = 1$, such that $x^* = \sum_{i=1}^{p} \mu_i x^i$.

Let's assume that $U'(x') = \max \{U'(x^1), \ldots, U'(x^p)\}$. Since $U'(x)$ is strictly quasiconvex, then $U'(x^*) = U'(\sum_{i=1}^{p} \mu_i x^i) < U'(x')$ by definition. This is a contradiction. Therefore solution $x^*$ should be an extreme point.

Next, we show that such an extreme point is efficient by contradiction also. So, let the extreme point $x^* \in X^*$ be inefficient. Then, by the definition of efficiency, there exists another extreme point $x' \in X$ such that $f(x') \geq f(x^*)$. This implies that $U'(x) = U(f(x')) > U(f(x^*)) = U'(x^*)$ since by the monotonicity assumption. Hence, $x^* \notin X^*$ which is a contradiction. Therefore, the extreme point $x^* \in X^*$ is efficient, Q.E.D.
Next, Corollary 5.1 establishes the same result given by theorem 5.1 for the case of linear objective functions.

Corollary 5.1

Let \( X \) be a convex set. Let \( f: \mathbb{R}^n \rightarrow \mathbb{R}^m \) and \( f_i(x) \) be linear and differentiable on \( X \) for each \( i = 1, \ldots, m \). If \( U(f): \mathbb{R}^m \rightarrow \mathbb{R} \) is a nondecreasing quasiconvex, then \( U'(x) = U(f(x)) \) is also quasiconvex.

Proof

Since each \( f_i(x) \) is linear, then each \( f_i(x) \) is concave and convex on \( X \). Therefore, \( U'(x) = U(f(x)) \) is also quasiconvex on \( X \) by theorem 5.1 above, Q.E.D.

5.3.2 The Interactive Branch and Bound Algorithm

5.3.2.1 Overview

The developed interactive algorithm is an extension of the Falk and Hoffman's (FH) global convex maximization algorithm for single objective (Falk and Hoffman 1986). Their algorithm is finite, employs only pivoting operations and is guaranteed to terminate at a global solution. What makes it an appealing candidate for an extension to an interactive multiobjective algorithm is that it requires only objective function evaluations. This so because the single objective function is replaced by the DM to evaluate multiobjective decisions through their associated alternatives. Furthermore, for the MOLP problems, a most-preferred solution will be at one of the extreme points.

Their algorithm can be summarized as follows. First, a polytope \( C \) is constructed that has one dimension greater than the
dimension of $X$ in such way that the set $X$ becomes one of its faces. Then, by successive collapsing, enclosing polytopes of the set $X$ are constructed till termination. The process of augmenting the feasible set $X$ in a set $C$ of higher dimension is done by formulating and solving an optimization problem that identifies the center of the set $X$. However, the process of polytopes collapsing is executed by maintaining a partial list of extreme points of $C$, as well as edges emanating from them using a tree structure at each iteration of the algorithm. Those edges are utilized to project the terminal vertices of the tree on the hyperplane containing the set $X$. This projection yields a set of points, whose convex hull defines a polytope that contains $X$. Hence, this process utilizes a branching as well as a projection scheme.

The center-finding problem for the set $X = \{ x \mid a_i^T x \leq b_i, \; i = 1, \ldots, m \}$ is formulated as follows,

$$\text{maximize} \{ \text{minimum} \{ (b_i - a_i^T x)/\|a_i\| \} \}.\quad x \quad i = 1, \ldots, m$$

Equivalently, the problem can be written as follows,

$$\text{maximize } y$$

s.t. \( (b_i - a_i^T x)/\|a_i\| \geq y, \; i = 1, \ldots, m, \)

$$y \geq 0.$$

Or,

Program 5.2: \quad \text{maximize } y

s.t. \( Ax + ay \leq b, \)

$$y \geq 0,$$
where \( \mathbf{a} = (\|\mathbf{a}_1\|, \ldots, \|\mathbf{a}_m\|) \).

The desired set \( \mathbf{C} \) is \( \{(x; y) \mid \mathbf{A}x + \mathbf{a}y \leq \mathbf{b}, y \geq 0\} \). In the next subsection, the steps of the interactive multiobjective algorithm are presented. Through a branch and bound framework, the algorithm builds a tree in the objective space as well as in the decision space. At each iteration, the terminal nodes of the associated tree contain solutions that are feasible as well as infeasible points to the original set \( \mathbf{X} \). Before the decision maker evaluates those points (decisions) at a particular iteration, inefficient decisions with respect to the set itself and inefficient feasible ones with respect to the original set \( \mathbf{X} \) at that iteration are first removed using an efficiency screening and testing process respectively. Then, the associated screened efficient alternatives are presented to the DM to determine the most-preferred alternative in the objective space, and thereby the best solution in the decision space. The associated vertex and thereby the node denoted as \( \mathbf{v}^* \) is chosen for the next expansion, unless the termination criterion is satisfied. During the execution of the algorithm, the decision maker might be considering dummy alternatives that correspond to infeasible decisions among the set presented during this iterative process. In a different context, dummy alternatives have been used for speeding up the convergence process of interactive algorithms (Koksalan et al. 1987; Malakooti 1989). Let the list \( \mathbf{T}_i \) contains all of the i-th tree nodes
generated from the beginning to the i-th iteration of the algorithm and the list $E^i$ contains only the terminal nodes of the i-th tree.

5.3.2.2 Steps of the Algorithm and its convergence

Before the steps of the algorithm are presented, we provide the following assumptions,

(1) The set $X$ of program 5.1 is bounded and nonempty.
(2) every basic solution of $X$ of program 5.1 is nondegenerate.
(3) every basic solution of $C$ of program 5.2 is nondegenerate.
(4) the solution of program 5.2 is unique.

Step 0 Initialization

Solve problem program 5.2. Let the solution be $v^0 = (x^0; y^0)$. Set $T^0 = \{v^0\}$, $E^0 = \emptyset$, $v^* = v^0$ and $i = 0$.

Step 2 Branching and Projection

(i) Expand node $v^*$ to generate all its adjacent vertices $v^*j$ of $v^*$, such that (a) $y^*j < y^*$ and (b) $v^*j \notin T^i$.
(ii) Project each $v^*j$ on the hyperplane $y = 0$ to find its associated decision $x^*j$.
(iii) Update the sets $T^{i+1}$ and $E^{i+1}$ respectively.
(iv) Set $i = i + 1$.

Step 3 Testing and Bounding

(i) Screen alternatives in $E^i$ for efficiency and present only efficient alternatives to the decision maker.
(ii) Ask him/her to select the best alternative, $f^*$. 


(iii) Identify its associated decision $x^s$.

(iv) If $x^s$ is feasible with respect to $X$, STOP; $x^s$ is the solution.

Else Set $v^* = v^s$. Go to Step 2.

**Theorem 5.4**

Let $X \subseteq \mathbb{R}^n$ be a convex polyhedron and the utility function $U(f)$ be quasiconvex on the objective feasible region $F$. Then the above algorithm yields the most-preferred solution.

**Proof**

Since the utility function $U(f)$ be quasiconvex on the objective feasible region $F$ and each objective function is linear on the convex set $X$, then $U(f(x))$ is also quasiconvex on $X$ by corollary 5.1. The algorithm generates a sequence of polytopes $X^0 \supset X^1 \supset X^2 \supset \ldots \supset X$ (Falk and Hoffman 1986). Let the termination occurs at the i-th iteration. This implies that, by responses of the decision maker, $U(f(x^s)) \geq U(f(x))$ for all extreme points $x \in X^i$ and hence $U(f(x^s)) \geq U(f(x))$ for all points $x \in X^i$ since $U(f(x))$ is quasiconvex on $X$. Since $X^i \supset X$, then $U(f(x^s)) \geq U(f(x))$ for all $x \in X$. Now, since $x^s \in X$, then there exist an extreme point of $X$ such that $U(f(x^s)) \geq U(f(x))$ for all $x \in X$ and hence $x^s$ is the most-preferred solution, Q.E.D.
5.3.2.3 Implementation Issues

At the initialization stage of the algorithm, program 5.2 is solved. By augmenting the objectives as equality constraints, the initial tableau of program 5.2 (minimize -y) is as follows,

<table>
<thead>
<tr>
<th></th>
<th>x</th>
<th>y</th>
<th>s</th>
<th>f</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>a</td>
<td>I</td>
<td>0</td>
<td></td>
<td>b</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>0</td>
<td>I</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

where, the constraint set of program 5.2 has been rewritten as Ax + ay + s = b, where s is the set of slack variables. Its solution identifies the largest sphere contained in the set X with \( x^o \in X \) and \( y^o \) as its center and radius respectively assuming that the optimal solution is unique and nondegenerate. An example of the branching and projection stage is that of generating the initial enclosing polyhedron \( X^o \). In order to construct \( X^o \), the tree \( T^0 = \{ v^0 = (x^o; y^o) \} \) is used whose only vertex \( v^0 = (x^o; y^o) \) is chosen to be the branching node. This vertex is denoted as \( v^* \). Then, two steps are needed. First, the \( (n+1) \) adjacent vertices of \( (x^o; y^o) \) are identified. Second, each adjacent vertex \( v^j = (x^j; y^j) \) is projected on the hyperplane \( y = 0 \), where the original set X lies. The projection process is accomplished by extending the extreme direction beginning at \( (x^o; y^o) \) that passes through the vertex \( (x^j; y^j) \) till it intersects the hyperplane \( y = 0 \),
where it defines the point \((x^j;y^j)\). The collection of these points 
\([(x^1;y^1), \ldots, (x^{(n+1)};y^{(n+1)})]\) defines the first enclosing polytope \(X^0\).

However, generating the next enclosing polytopes involves an extra work. First of all, the branching process is a user-driven rather than automatic. The set of efficient decisions (projected values) contained in the terminal nodes are presented to the decision maker for evaluation via their associated outcomes. Having the most-preferred outcome been identified by the decision maker, its associated vertex denoted as \(v^*\) is declared for selection unless it satisfies the feasibility of the set \(X\). Secondly, not all adjacents of the branching node \(v^*\) are generated. Eligible adjacent vertex \(v^*j\) should satisfy the following two properties,

1. \(y^*j < y^*\),  
2. \(v^*j\) has never been generated.

These two properties will make sure that we are approaching the hyperplane \(y = 0\) and the generated nodes are in one-to-one correspondence with vertices of the set \(C\) respectively.

Given the tableau representation for the vertex \(v^*\), there are precisely \((n+1)\) adjacent vertices, one for each nonbasic variable. These can be identified by pivoting in each of these nonbasic columns. The pivot element is determined by using the usual minimum-ratio rule. However, those nonbasic columns that yield a vertex satisfying the first eligibility property can be identified using the row of the tableau corresponding to the variable \(y\). Let the \(i\)-th
row of the $v^*$ tableau contains $y$ as a basic variable, then the associated row equation follows,

$$y + \Sigma a_{ij}x_j = y^*, \text{ where } x_j \text{ is a nonbasic variable.}$$

Pivoting in any of these column forces the value of $y^*$ be decreased only if the associated $a_{ij}$ is positive. However, checking to insure that any adjacent vertex $v^{**j}$ has never been generated requires searching all of the previously generated vertices. Furthermore, this row equation is used to accomplish the projection process. To determine the projected point of any eligible adjacent vertex, a pivoting operation is performed on the element $a_{ij}$ identified earlier. This pivoting will derive the value of the variable $y$ to zero since it becomes a nonbasic variable and the projected value results.

At the testing and bounding stage, the projected values associated with the current terminal nodes are screened to remove inefficient points. These alternatives might include feasible as well as infeasible points with respect to original constraint set $X$. First, they are screened for efficiency with respect to each other. Then, feasible alternatives, if any remains, are tested for efficiency with respect to the set $X$. This due to the fact that these inefficient feasible points will never be candidate to be selected for expansion. Testing for efficiency can be accomplished by the following procedure. Let $f^j$ be the alternative to be tested for efficiency. It can be tested by maximizing the underdeviation $y^-$ of $f^j$ from the nondominated frontier using the following formulation
Program 5.3: \[ Z^- = \max \sum_{i=1}^{k} y_i^- \]
\[ \{x, y^-\} \]
\[ \text{s.t.} \]
\[ f_i(x) - y_i^- = f_i^j \quad 1 \leq i \leq k, \]
\[ x \in X, \ y_i^- \geq 0, \quad 1 \leq i \leq k. \]

If \( Z^- \) is zero then \( \mathcal{F} \) is nondominated; otherwise it is dominated.

The set of efficient points, either feasible or infeasible, are presented to the decision maker through their associated alternatives for evaluation. Having the decision maker decided on the most-preferred solution, termination criterion is checked. If the most-preferred point is feasible with respect to the original constraint set \( X \), the algorithm terminates with this point as the most-preferred solution for the problem. Otherwise, the associated node is declared for expansion in the next iteration.

5.3.2.4 Example 5.1

An example to illustrate the steps of the algorithm of this section is presented in this subsection. See figure 5.1 for the shape of the constraint set.

Program ex5.1:
\[ \maximize f_1 = 5x_1 + x_2, \quad f_2 = x_1 + 4x_2 \]
\[ \text{s.t.} \]
\[ x_1 + x_2 \geq 1, \]
\[ x_1 - 2x_2 \leq 1, \]
\[ 2x_1 - x_2 \leq 5, \]
\[3x_1 + 5x_2 \leq 27,\]
\[-6x_1 + 10x_2 \leq 30,\]
\[x_1, x_2 \geq 0.\]

Let the quasiconvex utility function be
\[U(f(x)) = \exp(f_1(x_1)) + \exp(f_2(x_2))\]

Solution

Initialization

Formulating the center-finding problem as program 5.2 and presenting it in the canonical form yield (see figure 5.2 for the shape of constraint set),

**Program ex5.2:**

maximize \(y\)

s.t.

\[x_1 + x_2 + y - s_1 = 1,\]
\[x_1 - 2x_2 + y + s_2 = 1,\]
\[2x_1 - x_2 + y + s_3 = 5,\]
\[3x_1 + 5x_2 + y + s_4 = 27,\]
\[-6x_1 + 10x_2 + y + s_5 = 30,\]
\[x_1 + y - s_6 = 0,\]
\[x_2 + y - s_7 = 0,\]
\[5x_1 + x_2 - f_1 = 0\]
\[x_1 + 4x_2 - f_2 = 0\]
\[x_1, x_2, y \geq 0.\]

Solving the above optimization problem yields \(v^0 = (x^0, y^0) = (1.80, 2.20; 1.61)\). Hence \(T^0 = E^0 = \{(1.80, 2.20; 1.61)\}\), where every point is expressed in the form \((x; y)\). Set \(v^* = v^0\) and \(i = 0\). The nonbasic portion of the final tableau of program ex5.2 follows, where
the numbers associated with the nonbasic columns are the column numbers in the tableau and likewise for the numbers associated with the rows,

<table>
<thead>
<tr>
<th>basics</th>
<th>5:s2</th>
<th>6:s3</th>
<th>8:s5</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:x2</td>
<td>-0.52</td>
<td>0.35</td>
<td>0.03</td>
<td>2.20</td>
</tr>
<tr>
<td>2:s7</td>
<td>-0.32</td>
<td>0.38</td>
<td>0.07</td>
<td>3.81</td>
</tr>
<tr>
<td>3:x1</td>
<td>-0.48</td>
<td>0.65*</td>
<td>-0.03</td>
<td>1.80</td>
</tr>
<tr>
<td>4:s4</td>
<td>2.89*</td>
<td>-3.87</td>
<td>-0.31</td>
<td>1.21</td>
</tr>
<tr>
<td>5:s1</td>
<td>-0.72</td>
<td>1.04</td>
<td>0.06</td>
<td>5.28</td>
</tr>
<tr>
<td>6:y</td>
<td>0.20</td>
<td>0.03</td>
<td>0.04*</td>
<td>1.61</td>
</tr>
<tr>
<td>7:s6</td>
<td>-0.28</td>
<td>0.67</td>
<td>0.01</td>
<td>3.41</td>
</tr>
<tr>
<td>f1</td>
<td>-2.92</td>
<td>3.58</td>
<td>-0.13</td>
<td>11.20</td>
</tr>
<tr>
<td>f2</td>
<td>-2.56</td>
<td>2.06</td>
<td>0.09</td>
<td>10.60</td>
</tr>
<tr>
<td>y</td>
<td>0.20</td>
<td>0.03</td>
<td>0.04</td>
<td>1.61</td>
</tr>
</tbody>
</table>

Iteration (1)

In the branching and projection stage, all of the three adjacent vertices are eligible for consideration. Those are generated using the minimum-ratio rule and then projected on the hyperplane y = 0 using their associated entry of the y-row in the above tableau. As an example, in order to generate the first adjacent, the nonbasic variable s2 enters the basis and the basic variable s4 exit according to the minimum-ratio rule (i.e. the pivot element is a65). The pivoting operation yields the vertex v0,1 = (2.00,2.42;1.53). Now, to project
Figure 5.1: Constraint set of program ex5.1 of example 5.1.

Figure 5.2: Constraint set of program ex5.2 of example 5.1.
this vertex on the hyperplane \( y = 0 \), we pivot on the entry \( a_{35} \). In other words, the value of the basic variable \( y \) is derived to zero in the following equation,

\[
y + 0.20s_2 + 0.03s_3 + 0.04s_5 = 1.61.
\]

The adjacent vertices are \{(2.00, 2.42; 1.53), (0.00, 1.21; 1.53), (3.00, 1.00; 0.00)\} and their projected points are \{(5.71, 6.42; 0.00), (-35.37, -18.00; 0.00), (3.00, 1.00; 0.00)\}. The vertices of \( T^1 \) and \( E^1 \) follows,

<table>
<thead>
<tr>
<th>vertices of ( T^1 ): ((x;y))</th>
<th>vertices of ( E^1 ): ((x;y))</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1.80, 2.20; 1.61)</td>
<td>(2.00, 4.20; 0.00)</td>
</tr>
<tr>
<td>(2.00, 2.42; 1.53)</td>
<td>(4.00, 3.00; 0.00)</td>
</tr>
<tr>
<td>(0.00, 1.21; 1.53)</td>
<td>(3.00, 1.00; 0.00)</td>
</tr>
<tr>
<td>(3.00, 1.00; 0.00)</td>
<td></td>
</tr>
</tbody>
</table>

However, the list \( E^1 \) is as follows, where its contents have more details about its vertices. The contents include each vertex and its associated projection and alternative.

<table>
<thead>
<tr>
<th>TERMINAL</th>
<th>LIST</th>
<th>( E^1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vertex: ((x;y))</td>
<td>Projection: ((x))</td>
<td>Alternative: ((f))</td>
</tr>
<tr>
<td>(2.00, 2.42; 1.53)</td>
<td>(5.71, 6.43)</td>
<td>(35.0, 31.43)</td>
</tr>
<tr>
<td>(0.00, 1.21; 1.53)</td>
<td>(-35.00, -18.00)</td>
<td>(-193.0, -107.0)</td>
</tr>
<tr>
<td>(3.00, 1.00; 0.00)</td>
<td>(3.00, 1.00)</td>
<td>(16.0, 7.00)</td>
</tr>
</tbody>
</table>
For the testing and bounding stage, the set of alternatives in the list $E^1$ is screened using the efficiency definition. It turns out that the only efficient alternative is $(35.0,31.43)$. Hence the decision maker has no alternatives to evaluate, and therefore the algorithm continues using the tableau of vertex $(2.00,2.42;1.53)$ for expansion. Since this point is infeasible with respect to the original constraint set, its associated node is declared for selection and hence $v^* = (2.00,2.42;1.53)$.

**Iteration (2)**

Expanding the node $v^* = (2.00,2.42;1.53)$ yields the following eligible adjacent vertices, $\{(2.00,4.20;0.00), (4.00,3.00;0.00)\}$. Since they are already lie in the hyperplane $y = 0$, their projected values are the same. Hence the lists $T^2$ and $E^2$ are as follows,

<table>
<thead>
<tr>
<th>vertices of $T^2$: $(x;y)$</th>
<th>vertices of $E^2$: $(x;y)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1.80,2.20;1.61)</td>
<td>(0.00,1.21;1.53)</td>
</tr>
<tr>
<td>(2.00,2.42;1.53)</td>
<td>(3.00,1.00;0.00)</td>
</tr>
<tr>
<td>(0.00,1.21;1.53)</td>
<td>(2.00,4.20;0.00)</td>
</tr>
<tr>
<td>(3.00,1.00;0.00)</td>
<td>(4.00,3.00;0.00)</td>
</tr>
<tr>
<td>(2.00,4.20;0.00)</td>
<td></td>
</tr>
<tr>
<td>(4.00,3.00;0.00)</td>
<td></td>
</tr>
</tbody>
</table>
The list $E^2$ is as follows, where its contents have more details about its vertices. The contents include each vertex and its associated projection and alternative

<table>
<thead>
<tr>
<th>TERMINAL</th>
<th>LIST</th>
<th>$E^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x; y$</td>
<td>$x$</td>
<td>$f$</td>
</tr>
<tr>
<td>(0.00, 1.21; 1.53)</td>
<td>(-35.00, -18.00)</td>
<td>(-193.0, -107.0)</td>
</tr>
<tr>
<td>(3.00, 1.00; 0.00)</td>
<td>(3.00, 1.00)</td>
<td>(16.0, 7.00)</td>
</tr>
<tr>
<td>(2.00, 4.20; 0.00)</td>
<td>(2.00, 4.20)</td>
<td>(14.2, 18.8)</td>
</tr>
<tr>
<td>(4.00, 3.00; 0.00)</td>
<td>(4.00, 3.00)</td>
<td>(23.0, 16.0)</td>
</tr>
</tbody>
</table>

The testing and bounding stage consists of three subtasks. First, the alternatives of the list $E^2$ are screened for efficiency with respect to themselves. This process yields the following set of efficient alternatives $\{(14.2, 18.8), (23.0, 16.0)\}$. Second, these two alternatives are checked for efficiency with respect to the constraint $X$ since both of them are feasible. This is done using program 5.3, where both of them is found to be efficient. As an example, consider alternative $(14.2, 18.8)$. Then, program 5.3 gives

Program ex5.3: 

$$\begin{align*}
Z^- &= \text{maximize } \Sigma_{i=1}^{2} y_i^- \\
\{x, y^-\} &\text{s.t.} \\
&& f_1(x) - y_1^- = 14.2 \\
&& f_2(x) - y_2^- = 18.8
\end{align*}$$
\[ x \in X, \ y_1^-, \ y_2^- \geq 0. \]

Third, these two alternatives are presented to the decision maker for evaluation. The DM is asked to identify the most-preferred alternative among the set (in this case there are only two alternatives). These alternatives and their utilities are follows,

<table>
<thead>
<tr>
<th>Alternative: (f)</th>
<th>Utility: (U)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(14.2, 18.8)</td>
<td>10.7</td>
</tr>
<tr>
<td>(23.0, 16.0)</td>
<td>14.9</td>
</tr>
</tbody>
</table>

Now, by evaluating those two alternatives, alternative (23.0, 16.0) has the highest utility. Since its associated point (4.00, 3.00) is feasible with respect to the set \( X \), it is declared as the most-preferred solution to the whole problem. Solution tree is presented in Figure 5.3 next.
Figure 5.3: The Solution Tree of Example 5.1.
5.3.2.5 Implementation and Computational Experiments

The above interactive algorithm was implemented on an IBM-PC using TURBO PASCAL programming language. The choice of using this language was due to the fact that it has features that does not exist in other classical programming languages usually used such as BASIC and FORTRAN and consequently it will make it possible to write an efficient code. The most important feature that this language provides is dynamic data structures such as linked lists and trees. Through this type of data structures, one does not have to reserve a large part of computer memory, say a large n-dimensional array, but rather starts with one unit or cell and add extra units or cells as needed during the execution of the program. In other words, the data structure grows as needed while the program is running. Furthermore, units of computer memory or cells can be recycled whenever their contents are no longer needed.

The code, which is named QXMOLP, starts by echoing the required data of the multiobjective linear programming problem. This data consists of the following lines of input: first line contains the number of objectives, the number of decision variables (slack and surplus variables are not included), and the number of constraints (nonnegativity constraints are included). Then the objective functions coefficients are added one by one, where each objective function coefficients are separated by a carriage return from the other objective function. Finally, constraints coefficients are added where '> is replaced by 1 and '< ' is replaced by -1. Using the
coefficients of the problem constraints, the center-finding problem is formulated and the initial tableau is constructed. Having set the initial tableau up, a pivoting-based simplex routine is used to solve the linear programming problem where the initial node of the solution tree is constructed.

The next two main steps are the branching-projection and testing-bounding steps. In the branching-projection step, the solution tree is expanded in each iteration till termination. Two linked lists are used to store the nodes of the tree. The first linked list, called TERMINAL, keeps the terminal nodes of the tree. Each terminal node contains the following information:

1. the tableau representation of the associated vertex.
2. the associated solution vector in the (n+1)-space.
3. the associated projected vector in the n-space.
4. the associated alternative in the objective space.

However, the second linked list, called TREE, keeps the nonterminal nodes of the tree. These nodes have been already explored and therefore less information about them is needed. Hence, each of these nodes contains only the associated solution vector, where it is used for a check when a fresh vertex is generated. The union of the two linked lists forms the whole solution tree.

In the testing-bounding step, associated alternatives from the terminal linked list are extracted. Before presenting them to the decision maker, this set of alternatives are screened to remove all dominated alternatives. Then, the DM identifies the best one in the
set. If the solution associated with the best alternative is feasible to the original constraint set $X$, the algorithm is concluded with this solution and its associated alternative as the most-preferred solution and alternative for the whole problem. Otherwise, the associated vertex is declared as the next candidate for expansion. This results in creating a nonterminal node, storing the associated decision vector in it and adding the node to the TREE linked list. Furthermore, once the content of the terminal node being selected for expansion is used, it is removed from the TERMINAL linked list and recycled. New terminal nodes are constructed to store the information of the newly generated vertices and added to the TERMINAL linked list.

Computational experiments can address some performance aspects of the algorithm such as total alternatives and tradeoffs presented to the decision maker, total iterations, and the deviation from the true solution for each simulated decision situation. Therefore, in order to test the developed algorithm, ten different problems are arbitrarily selected from the literature. These problems have up to 5 objectives, 7 decision variables (slack and surplus variables are not included), and 11 constraints (nonnegativity constraints are included) and chosen for their computational feasibility. The simulated utility function for all of these problems have the following general form:

$$U(f) = \exp(\alpha f_1) + \exp(\alpha f_2) + \ldots + \exp(\alpha f_k).$$

Summary of the experimental results are presented in table 5.1 next. Table 5.1 entries include,
Table 5.1: Experiments for the quasiconvex utility-based MOLP.

<table>
<thead>
<tr>
<th>Problem #</th>
<th>Problem Size</th>
<th>Max. # of Alternat.</th>
<th>Min. # of Alternat.</th>
<th>Avg. # of Alternat.</th>
<th>Tot. Itr. / Tot. Ques.</th>
<th>Total Alt. presented</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2 x 2 x 3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1 / 1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2 x 3 x 4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>1 / 1</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>2 x 2 x 4</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1 / 1</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>3 x 4 x 6</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1 / 1</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>3 x 3 x 6</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>2 / 2</td>
<td>8</td>
</tr>
<tr>
<td>6</td>
<td>3 x 4 x 7</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>1 / 1</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>3 x 3 x 7</td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>2 / 2</td>
<td>10</td>
</tr>
<tr>
<td>8</td>
<td>3 x 7 x 11</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>1 / 1</td>
<td>6</td>
</tr>
<tr>
<td>9</td>
<td>3 x 6 x 16</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1 / 1</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>5 x 4 x 11</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>1 / 1</td>
<td>4</td>
</tr>
</tbody>
</table>

1. Problem size: \( k \times n \times m \), where \( k \) is the number of objective functions, \( n \) is the number of decision variables (slack and surplus variables are not included), and \( m \) is the number of constraints (nonnegativity constraints are included).

2. Maximum number of alternatives presented to the decision maker during an iteration.

3. Minimum number of alternatives presented to the decision maker during an iteration.
4. Average number of alternatives presented to the decision maker at any iteration during the solution process.

5. Total number of iterations / Total number of questions: total number of questions gives total number of times in which the decision maker was faced with a preference comparison question. Each question requires the DM to identify the best alternative among a set of nondominated alternatives. Total number of alternatives in each set was kept at an average of five.

6. Total number of alternatives presented to the decision maker during the solution process.

Test problems 3, 4, 5, 6, 7, 8, 9, and 10 were taken from Hwang and Masud (1979), Zionts and Wallenius (1976), Zeleny (1974), Hwang and Masud (1979), Zionts and Wallenius (1983), Zeleny (1974), Hwang and Masud (1979), and Ignizio (1980) respectively.

The results showed that the algorithm converges within just a few iterations. They also showed that the number of total alternatives presented to the decision maker for preference evaluation during the process increases as the problem size increases.

5.4 Multiobjective Nonlinear Programming

The structure of the multiobjective nonlinear programming model exhibits nonlinearity in some of its objective functions and constraints. Its mathematical formulation is as follows,
maximize $f_1(x), f_2(x), \ldots, f_k(x)$

s.t.

$x \in X = \{ x \mid g_i(x) \leq 0, i = 1, \ldots, m \}$.

5.4.1 Basic Theories

Under mild extra properties for the quasiconvex utility function, characterizations for the most-preferred solution set is provided. In the following, we show that one of the results that characterizes the most-preferred solutions to be supported if the utility function is strictly quasiconvex. Furthermore, if the utility function has a strictly positive gradient components everywhere, the most-preferred solution is a properly nondominated alternative. These characterizations will play a vital role in designing the interactive algorithm.

Theorem 5.5

Let the utility function $U(f)$ be strictly quasiconvex on a convex set $V \supset F$. If $f^*$ is a most-preferred alternative, then it is a supported alternative.

Proof (by contradiction)

Let $f^* \in F^*$ be an unsupported alternative. This implies that $U(f^*) \geq U(f)$ for all $f \in F$. Furthermore, since alternative $f^*$ is unsupported, then it is dominated by some convex combination of a finite set of nondominated alternatives, say $f^1, \ldots, f^p$. This implies that there exist multipliers $\mu_i \geq 0$, $\sum_{i=1}^p \mu_i = 1$, such that $\sum_{i=1}^p \mu_i f^i \geq f^*$. 
Assume that \( U(f^*) = \max (U(f^1), \ldots, U(f^P)) \). Since \( U(.) \) is strictly quasiconvex, then by definition \( U(\sum_{i=1}^{P} \mu_i f^i) < U(f^*) \). Also, \( U(\sum_{i=1}^{P} \mu_i f^i) \geq U(f^*) \) since \( U(.) \) is increasing. This implies that \( U(f^*) > U(f^*) \) which is a contradiction. This implies that alternative \( f^* \) should be supported, Q.E.D.

**Theorem 5.6**

Let the utility function \( U(f) \) be quasiconvex of a convex set \( V \supseteq F \). Furthermore let \( \nabla U(f) > 0 \) for all \( f \in F \). If \( f^* \) is a most-preferred alternative, then it is a properly nondominated alternative.

**Proof (by contradiction)**

Let \( f^* \in F^* \) be an improperly nondominated alternative. Since \( f^* \in F^* \), then the pair \( (x^*, f^*) \) can be recovered by solving the following auxiliary problem (Katoh and Ibaraki 1987),

\[
\begin{align*}
\text{maximize} & \quad w^t f(x) \\
\text{s.t.} & \quad x \in X,
\end{align*}
\]

where \( w = \nabla U(f^*) \).

Now, since \( w = \nabla U(f^*) > 0 \), the above problem yields only properly nondominated solutions (Geoffrion 1972) and hence a contradiction. Therefore, \( f^* \) must be a properly nondominated alternative, Q.E.D.

**Theorem 5.7**

Let \( U(f) \) be quasiconvex on the set \( F \) and \( f^* \) be an alternative. Consider the problem \( P(w^*) \):

\[
P(w^*): (\text{maximize} \quad w^t f(x))
\]

\[
\text{s.t.} \quad x \in X, \quad \text{where} \quad w^* = \nabla U(f^*).
\]
If \( f' \in F^*(w') \), then \( f' \) is utility inefficient.

**Proof**

Assume \( f'' \in F^*(w') \), then \( \nabla U(f')f'' > \nabla U(f')f' \) for all \( f \in F \) and \( f \in F^*(w') \). Since \( f' \in F^*(w') \), then \( \nabla U(f')f'' > \nabla U(f')f' \), therefore \( \nabla U(f')(f'' - f') > 0 \). Since \( U(.) \) is quasiconvex, this implies that \( U(f'') > U(f') \). Since there exist an alternative \( f'' \neq f' \) such that \( U(f'') > U(f') \). Therefore, \( f' \) is utility inefficient, Q.E.D.

5.4.2 An interactive Heuristic Algorithm

5.4.2.1 Overview

At the core of designing an interactive multiobjective algorithm is the generation of nondominated alternatives. In general, the nonlinearity of the model requires the use of sophisticated alternative generating schemes such as the weighted-Tchebycheff scheme. This stems from the fact that unsupported as well as improper alternatives have the potential to be members of the most-preferred solution set \( F^* \). Fortunately, due to the above characterization of the most-preferred solutions set, the simple weighted sum of the objective functions is sufficient. This scheme is relatively computationally inexpensive to perform.

The interactive algorithm, presented next, is first initialized by generating a nondominated alternative. Then the gradient of the utility function is assessed interactively and utilized to form the weighted-sum auxiliary problem. Once this problem is solved, it will yield an alternative that has a higher utility unless it returns the current alternative again. Therefore the algorithm has the unique
feature that better alternatives will be automatically provided based on the solution of the P(w) problem (Theorem 5.7). Its solution sets are X*(w) and F*(w) for decision variables and alternatives respectively. Furthermore, the algorithm consists of a few simple steps and requires only assessment of the gradient of the utility function. The algorithm is heuristic in nature in the sense that it does not guarantee to converge to an optimal solution. However, we believe that it has the potential to perform well when multiple executions are done with random starting points. The steps of the interactive algorithms are presented in the next subsection.

5.4.2.2 Steps of the algorithm

Step 0  **Initialization**

Generate a nondominated alternative f^1.

set i = 1. Set list FINAL = \emptyset

Step 1  **Gradient Assessment and Alternative Generation**

Assess \nabla U(f^i) and set w = \nabla U(f^i).

Solve the following problem:

\[
P(w): \text{maximize } w^T f(x) \\
\text{s.t. } x \in X,
\]

to obtain its solution sets X*(w) and F*(w).

Step 2  **Testing**

If f^i \in F*(w^i), set f* = f^i and go to step (3).

Select any f' from F*(w^i), i = i + 1, set f^i = f' and go to Step 1.
Step 3 **Accumulating**

Add $f^*$ to the list **FINAL**.

Check with the DM if he would like to repeat the process;

YES : go to Step 0;

NO : go to Step 4.

Step 4 **Conclusion**

Present the DM with list **FINAL**.

DM chooses the most preferred alternative, denote it as $f^*$;

STOP.

**Remark 5.1**

The sequence of utility values for the alternatives generated by Step (4) of the above algorithm is strictly increasing.

**5.4.2.3 Implementation Issues**

In order to implement the above algorithm, Steps 2 and 3 demand most of the effort. In Step 2, the gradient of the utility function is required. Its value could be assessed using the one of the two methods discussed in subsection 4.5.1. Step 3 requires using an appropriate single optimization code to solve the auxiliary problem $P(w^1)$. Step 4 checks a stopping criterion. If it is satisfied, the current loop (Steps 2 through 4) is concluded heuristically with alternative $f^*$ as the best solution. Multiple executions using different starting points yield a final list of alternatives called **FINAL** of which the most-preferred final solution for the problem is chosen by the DM is
Step 7. Two examples to illustrate the algorithm are given next. For practical implementation of Step 2, the value of $w$ is randomly generated instead of assessing the gradient of the utility function $\nabla U$ at alternative $f^i$. The value of $w$ is generated randomly from the set $\Lambda$, where

$$\Lambda = \{ w \in \mathbb{R}^k | w_i \in [0, 1], \sum_{i=1}^{k} w_i = 1 \}.$$

### 5.4.2.4 Examples

Two examples are presented in this subsection for two cases. The first one is for the convex case, where the second is for the nonconvex case.

**Example 5.2** (the convex case)

maximize \{ $f_1 = x_1$, $f_2 = x_2$, $f_3 = x_3$ \}

s.t. $x_1^2 + x_2^2 + x_3^2 \leq 25$, $x_i \geq 0$ for $i = 1, 2, 3$.

Assume that the implicit DM's utility function

$$U(f_1, f_2, f_3) = 3\exp(f_1) + 2\exp(f_2) + \exp(f_3).$$

**solution**

Using the point $f^1 = (2.89, 2.89, 2.89)$ as a starting point, the gradient value is $(3\exp(8.66), 2\exp(8.66), \exp(8.66))$. This vector is used as the value for $w^1$ in Step 1. Solving the auxiliary problem $P(w^1)$ yields the point $(4.01, 2.67, 1.33)$. Since this point is not the same as the starting point, the algorithm iterates. The following table shows the results of the first three iterations:
<table>
<thead>
<tr>
<th>iteration</th>
<th>Starting point</th>
<th>Auxiliary Solutn</th>
<th>Utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(2.89, 2.89, 2.89)</td>
<td>(4.01, 2.67, 1.33)</td>
<td>198.10</td>
</tr>
<tr>
<td>2</td>
<td>(4.01, 2.67, 1.33)</td>
<td>(4.99, 0.06, 0.01)</td>
<td>447.92</td>
</tr>
<tr>
<td>3</td>
<td>(4.99, 0.06, 0.01)</td>
<td>(4.99, 0.02, 0.01)</td>
<td>447.85</td>
</tr>
</tbody>
</table>

The alternative (4.99, 0.02, 0.01) is taken to be the most preferred alternative for problem. In order to check how close this alternative to the true one, the used utility function is maximized explicitly and the alternative (4.999, 0.022, 0.002) was obtained using GINO (Liebman et al. 1986). If this is the true solution, our concluded alternative is almost exactly the same having 99.99% of the optimum's utility.

Example 5.3 (the nonconvex case)

\[
\text{maximize} \quad \{ f_1 = x_1 , f_2 = x_2 , f_3 = x_3 \} \\
\text{s.t.} \quad x_1^2 + x_2^2 - 900\exp(-x_3) \leq 0.0 \\
\quad x_3 \leq 1.0 , \quad x_i \geq 0 \quad \text{for } i = 1,2,3.
\]

Assume that the implicit DM's utility function

\[
U(f_1,f_2,f_3) = \exp(0.1f_1) + 1.5\exp(0.1f_2) + 5.0\exp(4.0f_3).
\]

\[solution\]

Using the point \( f^1 = (21.2, 21.2, 0.0) \) as a starting point, the gradient is: \( \nabla U(f^1) = [0.1\exp(2.121), 0.15\exp(2.121), 20.0\exp(0.000)] \).
This gradient vector is used as the value for $w^1$ in Step 1. Solving the auxiliary problem $P(w^1)$ yields the point $(10.095, 15.138, 1.000)$. Since this point is not the same as the starting point, the algorithm iterates. The next table shows the results of the first four iterations

<table>
<thead>
<tr>
<th>iteration</th>
<th>Starting point</th>
<th>Auxiliary Solutn</th>
<th>Utility</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(21.21,21.21,0.0)</td>
<td>(10.10,15.14,1.0)</td>
<td>282.55</td>
</tr>
<tr>
<td>2</td>
<td>(10.10,15.14,1.0)</td>
<td>(6.78,16.88,1.0)</td>
<td>283.08</td>
</tr>
<tr>
<td>3</td>
<td>(6.78,16.88,1.0)</td>
<td>(4.29,17.68,1.0)</td>
<td>283.32</td>
</tr>
<tr>
<td>4</td>
<td>(4.29,17.68,1.0)</td>
<td>(0.00,18.20,1.0)</td>
<td>283.24</td>
</tr>
</tbody>
</table>

The alternative $(0.00,18.20,1.00)$ is taken to be the most preferred alternative for problem. In order to check how close this alternative to the true one, the used utility function is maximized explicitly and the alternative $(2.560,18.015,1.000)$ was resulted using GINO (Liebman et al. 1986). If this is the true solution, our concluded alternative is almost exactly the same having 99.99% of the optimum's utility.

5.5 Conclusions

In this chapter, interactive multiobjective algorithms for handling quasiconvex utility functions are presented. It turns out that the structure of the constraint set of the multiobjective programming model plays a vital rule in developing the algorithms. In the case of multiobjective linear programming, the most preferred
solution occurs at an extreme point. For this case, an interactive branch and bound algorithm is developed. The algorithm is exact and requires only preference comparison questions. Computational experiments results showed that the algorithm converges within just a few iterations. They also showed that the number of total alternatives presented to the decision maker for preference evaluation during the process increases as the problem size increases. However, in the case of multiobjective nonlinear programming, an interactive heuristic algorithm is developed assuming extra mild conditions of strict quasiconvexity for the utility function. Both algorithms consist of a sequence of easily executable steps.
6. Conclusions and Future Research

In this research, we developed an interactive integrated optimization approach for decision making under multiple objectives. The approach exhibits several appealing features. It is Flexible, since two general classes of utility functions are handled in a unified framework, for the first time in the literature. The least restrictive form of generalized concavity or convexity (i.e., quasiconcave or quasiconvex) is assumed to represent the decision maker's preference. Furthermore, if the DM's preference happens to be inconsistent with both of these classes, the quasiconcave utility-based algorithm can be used as an exploratory search method with multiple starting points. This feature has not been provided by any previous research, and we believe it would be an effective concept for enhancing the overall DM's satisfaction of the final recommended solution. Also, we are not aware of any algorithm that handle quasiconvex utility functions. It is General, since it includes algorithms that are able to handle multiobjective linear as well as nonlinear programming problems with continuous decision variables. Furthermore, it can be extended to handle multiobjective integer programming as well as discrete multiple criteria decision making problems.

Furthermore, it is Easy to Use, since every algorithm of the approach consists of a sequence of easily executed steps. This feature
is necessary in order for users (especially naive ones) to understand the underlying problem-solving process. Consequently, they are likely to be more satisfied with it. It places \textit{Low Cognitive Load} on users, since the DM will be faced with preference comparison questions. These type of questions are considered to be easy relative to the other types, such as those based on marginal rate of substitution and reference values. Consequently, the cognitive burden would not be burdensome. It also \textit{Credible}, since in the case of the quasiconcave utility-based algorithm, all of alternatives presented to the DM during the process are nondominated and consequently the final solution must also be nondominated. Furthermore, the decision maker has control over the conclusion of the process through the interactive termination criterion. Also, in the case of the quasiconvex utility-based algorithms, the final alternative recommended by both algorithms (MOLP and MONLP) is guaranteed to be nondominated.

From a practical point of view, all steps of the approach and its imbedded algorithms can be implemented. Procedures for testing for utility class, for example, can be implemented using a minimum of two nondominated alternatives plus convex combination of the two. During any particular iteration of the algorithm, no more that seven alternatives need be presented to the DM. Finally, computational experiments results showed that the algorithms converge within just a few iterations.

In the case of the quasiconcave utility function, several
desirable characteristics were included. They include provision of only nondominated alternatives to the decision maker as well as control over the conclusion of the algorithm. Consequently, our algorithm is superior in terms of credibility to the Geoffrion, Dyer and Feinberg algorithm. Furthermore, we do believe that it will compete well with existing multiobjective algorithms that assume an implicit quasiconcave utility function, though they are very few. In the case of quasiconvex utility functions, there is no competition. Our algorithms are the only ones exist in the literature.

For further research, there are several possible directions. At the micro-level, we plan to investigate the effect on the utility class testing of the two sampling strategies of the nondominated frontier, one that includes convex combinations and another that does not. Another direction is to investigate how to implement the interactive termination criterion (ITC) for quasiconcave utility functions and how to handle the large number of directions that results when executing the interactive termination criterion in the case of large-scale multiobjective linear programming problems. At the macro-level, we plan to utilize the overall integrated framework to develop a integrated approach for discrete multiple criteria decision making (MCDM) problems, Finally, we intend to utilize our integrated framework for group decision making and negotiation.
7. REFERENCES


APPENDIX I: Proof of Theorem 4.6

Proof of Theorem 4.6 (by contradiction)

Let \( d \) be a feasible direction of \( C(X^*) \) at \( x^* \in X^* \) and assume that the set \( \{d^1, d^2, \ldots, d^s\} \) does not span the cone \( C(x^*) \). This implies that the system \( \{ \Sigma_{j=1}^q \mu_j d^j = d, \mu_j \geq 0, j = p+1, \ldots, s \} \) does not have a solution. Let \( h = s - p \), then the system can be written as follows, \( \{ \Sigma_{j=1}^p \alpha_j d^j + \Sigma_{j=1}^h \beta_j d^j = d, \beta_j \geq 0 \} \). By putting it into a matrix form, the system becomes \( \{ Q_1 \alpha + Q_2 \beta = d, \beta \geq 0 \} \) where \( Q_1 \) is \( n \times p \) and \( Q_2 \) is a \( n \times h \) matrices whose columns are \( \{d^1, d^2, \ldots, d^p\} \) and \( \{d^{p+1}, \ldots, d^s\} \) respectively. Now, by a corollary of Farkas theorem (Bazaraa and Shetty 1976), the system \( \{ Q_1 \alpha + Q_2 \beta = d, \beta \geq 0 \} \) does not have a solution implies that the system \( \{ \lambda^i Q_1 = 0, \lambda^i Q_2 \leq 0, \lambda^i d > 0 \} \) has a solution. This new system can be written as follows, \( \{ \lambda^i d^j = 0, j = 1, \ldots, p; \lambda^i d^j \leq 0, j = p+1, \ldots, s; \lambda^i d > 0 \} \).

Furthermore, since \( d \) is feasible direction of \( X^* \) at \( x^* \), then there exists an \( x \in X^* \) such that \( d = x - x^* \). Therefore \( Ad = A(x - x^*) = \mathbf{b} - \mathbf{b} = 0 \). Let \( d = [d_y, d_z]^t \), then \( [B \mid N][d_y, d_z]^t = 0 \) and hence \( d_y = -B^{-1}Nd_z \). Therefore \( d = [d_y, d_z]^t = [-B^{-1}Nd_z, d_z]^t = [-B^{-1}N, I]^td_z = \Sigma_{j=1}^s [-B^{-1}N, I]^td_{zj} = \Sigma_{j=1}^p [-B^{-1}N, I]^td_{zj} + \Sigma_{j=p+1}^s [-B^{-1}N, I]^td_{zj} = \Sigma_{j=1}^p d_{zj} + \Sigma_{j=p+1}^s d_{zj} \) by construction.

Now, for any \( \lambda \) that satisfies the system \( \{ \lambda^i d^j = 0, j = 1, \ldots, p; \lambda^i d^j \leq 0, j = p+1, \ldots, s \} \), it follows that \( \lambda^t d = \lambda^t (\Sigma_{j=1}^p d_{zj} + \Sigma_{j=p+1}^s d_{zj} + \Sigma_{j=1}^q \mu_j d^j = \lambda^t d \).
\( \sum_{j=p+1}^{s} \lambda^j d_j z_j \). Now, since \( \lambda^j d_j = 0 \), \( j = 1, \ldots, p \) then \( \sum_{j=1}^{p} \lambda^j d_j z_j = 0 \). Furthermore \( d_j = z_j - z_j^* = z_j \geq 0 \) since \( x = [y, z]^t \geq 0 \) and \( x^* = [y^*, 0]^t \geq 0 \). This also implies that \( \sum_{j=p+1}^{s} \lambda^j d_j z_j \leq 0 \) since \( \lambda^j d_j \leq 0 \), \( j = p+1, \ldots, s \). Therefore \( \lambda^t d = \lambda^t (\sum_{j=1}^{p} d_j z_j + \sum_{j=p+1}^{s} d_j z_j) \leq 0 \). This implies that there does not exist a feasible \( \lambda \) that makes the above system consistent which is a contradiction. Therefore, the set \( \{d_1, d_2, \ldots, d^s\} \) spans \( C(x^*) \), Q.E.D.
Appendix II: The GENERATE_EVALUTE Routine for The Quasiconcave Utility-based Algorithm.

program GENERATE_EVALUTE (input,output);

const
  maxobj = 3;

type
  objvector = array[1..maxobj] of real;

var
  fdstar,fstar,fb,fe,fobj,fin,fbest,g,tchw : objvector;
  i,j,k,status,n,number : integer;
  x,u,alpha,ubest,sum : real;
  ch : char;

procedure getobjvector(var f:objvector);
var
  i : integer;
begin
  for i := 1 to k do
  begin
    read(x);
    f[i] := x;
  end;
end;

procedure printobjvector(var f:objvector);
var
  i : integer;
begin
  write('(',
    write(f[1]:1:2);
  for i := 2 to k do
  begin
    write(',f[i]:1:2);
  end;
end;
write(' ') end;

procedure printwvector(var f:objvector); var i : integer;
begin write(' ['); write(f[1]:1:4); for i := 2 to k do
begin write(' ',f[i]:1:4); end;
write(' ]'); end;

procedure calculate(a:real;var f:objvector); var i : integer;
begin for i := 1 to k do f[i] := (1-a)*fb[i] + a*fe[i];
end;

procedure simulate_utility(f:objvector); var i : integer;
begin u := 0.0; for i := 1 to k do
begin u := u -(f[i]-fstar[i])* (f[i]-fstar[i]);
end;

procedure gradient_utility(f:objvector); var i : integer;
begin for i := 1 to k do
begin g[i] := 0.0;
begin g[i] := -(f[i]-fstar[i]);
end;
begin { main }
writeIn;
write(" HIT RETURN TO START ");
read(ch);
write(" ITERATION # ");
read(number);
writeIn;
writeIn;
write(" INPUT THE NUMBER OF OBJECTIVES, k: ");
read(k);
writeIn; writeln;
write(" INPUT 1 FOR GENERATION OR 2 FOR EVALUATION, 
      CHOICE: ");
read(status);
fstar[1] := 130.0;
fstar[2] := 250.0;
fdstar[1] := 133.0;
fdstar[2] := 255.0;
if status = 1 then
begin
writeIn;
writeIn(" INPUT THE BEGIN VECTOR: ");
getobjvector(fb);
writeIn;
writeIn(" INPUT THE END VECTOR: ");
getobjvector(fe);
writeIn;
write(" INPUT THE NUMBER OF ALTERNATIVES TO 
      GENERATED, n: ");
read(n);
writeIn;
writeIn(" THE GENERATED SET FOLLOWS:");
writeIn;
ubest := -1000000.0;
for i := 0 to n+1 do
begin
  alpha := i/(n+1);
calculate(alpha,fin);
write(i:2,' '); 
printobjvector(fin);
sum := 0.0;
for j := 1 to k do 
  begin 
    tchw[j] := 0.0;
    tchw[j] := 1.0/(fdstar[j]-fin[j]);
    sum := sum + tchw[j];
  end;
for j := 1 to k do tchw[j] := tchw[j]/sum;
write(' ');
printwvector(tchw);
simulate_utility(fin);
writeln(' U = ',u:7:1);
if u > ubest then 
  begin 
    fbest := fin;
    ubest := u;
  end;
end;
writeln;
write('BEST ALTERNATIVE: ');
printobjvector(fbest);
writeln(' best_utility = ',ubest:7:1);
gradient_utility(fbest);
write('GRADIENT VALUE: ');
printobjvector(g);
writeln;writeln;writeln;
end
else if status = 2 then
begin
  writeln;
  write('HOW MANY ? ');
  read(n);
  writeln;
  writeln('INPUT THEIR COMPONENTS: ');
  writeln;
  ubest := -1000000.0;
for i := 1 to n do
begin
  getobjvector(fobj);
  printobjvector(fobj);
  simulate_utility(fobj);
  writeln(' utility = ',u:7:1);
if u > ubest then
  begin
    fbest := fobj;
    ubest := u;
  end;
end;
writeln;
write('BEST ALTERNATIVE: ');
printobjvector(fbest);
writeln(' best_utility = ',ubest:7:1);
gradient_utility(fbest);
write('GRADIENT VALUE: ');
printobjvector(g);
writeln; writeln; writeln;
end
else writeln(' CHOOSE 1 OR 2');
end.
Appendix III: Test Problems for the Quasiconcave Utility-based Algorithm.

Test Problem # 1
# of Objectives: 2
# of Decision variables: 2
# of Constraints: 1
Objectives
\[ f_1 = x_1 \]
\[ f_2 = x_2 \]
Constraints
\[ x_1 + x_2 \leq 2 \]

Test Problem # 2
# of Objectives: 2
# of Decision variables: 3
# of Constraints: 1
Objectives
\[ f_1 = 4x_1 + 2x_2 + x_3 \]
\[ f_2 = -4x_1 - 3x_2 + -2x_3 \]
Constraints
\[ x_1 + x_2 + x_3 \leq 1 \]

Test Problem # 3
# of Objectives: 2
# of Decision variables: 2
# of Constraints: 2
Objectives
\[ f_1 = 0.4x_1 + 0.3x_2 \]
\[ f_2 = x_1 \]
Constraints
\[ x_1 + x_2 \leq 400 \]
\[ 2x_1 + x_2 \leq 500 \]

Test Problem  # 4
# of Objectives: 2
# of Decision variables: 2
# of Constraints: 3
Objectives
\[ f_1 = 4x_1 + x_2 \]
\[ f_2 = x_1 \]
Constraints
\[ 2x_1 + x_2 \leq 20 \]
\[ 0.83x_1 + x_2 \leq 10 \]
\[ x_1 + x_2 \geq 10 \]

Test Problem  # 5
# of Objectives: 3
# of Decision variables: 4
# of Constraints: 2
Objectives
\[ f_1 = 3x_1 + x_2 + 2x_3 + x_4 \]
\[ f_2 = x_1 - x_2 + 2x_3 + 4x_4 \]
\[ f_3 = -x_1 + 5x_2 + x_3 + 2x_4 \]
Constraints
\[ 2x_1 + x_2 + 4x_3 + 3x_4 \leq 60 \]
\[ 3x_1 + 4x_2 + x_3 + 2x_4 \leq 60 \]

Test Problem  # 6
# of Objectives: 3
# of Decision variables: 3
# of Constraints: 3
Objectives

- $f_1 = 4x_1 + x_2 + 2x_3$
- $f_2 = x_1 + 3x_2 - x_3$
- $f_3 = -x_1 + x_2 + 4x_3$

Constraints

- $x_1 + x_2 + x_3 \leq 3$
- $x_1 + 2x_2 + x_3 \leq 4$
- $x_1 - x_2 \leq 0$

Test Problem  #  7

# of Objectives: 3
# of Decision variables: 4
# of Constraints: 3

Objectives

- $f_1 = 10x_1 + 30x_2 + 50x_3 + 100x_4$
- $f_2 = x_1 + x_2$
- $f_3 = x_1 + 4x_2 + 6x_3 + 2x_4$

Constraints

- $5x_1 + 3x_2 + 2x_3 + x_4 \leq 420$
- $3x_1 + 8x_4 \leq 320$
- $2x_1 + 3x_2 + 4x_3 + 6x_4 \leq 180$

Test Problem  #  8

# of Objectives: 3
# of Decision variables: 3
# of Constraints: 4

Objectives

- $f_1 = x_1$
- $f_2 = x_2$
- $f_3 = x_3$

Constraints

- $3x_1 + 2x_2 + 3x_3 \leq 18$
- $x_1 + 2x_2 + x_3 \leq 10$
- $9x_1 + 20x_2 + 7x_3 \leq 96$
7x₁ + 20x₂ + 9x₃ ≤ 96

**Test Problem # 9**

# of Objectives: 3  
# of Decision variables: 7  
# of Constraints: 4

**Objectives**

\[ f₁ = x₂ + x₃ + x₄ + 3x₅ + x₆ \]
\[ f₂ = x₁ + x₃ - x₄ - x₆ \]
\[ f₃ = x₁ + 2x₂ - x₃ + 3x₄ + 2x₅ + x₇ \]

**Constraints**

\[ x₁ + 2x₂ + x₃ + x₄ + 2x₅ + x₆ + 2x₇ ≤ 16 \]
\[ -2x₁ - x₂ + x₄ + 2x₅ + 2x₇ ≤ 16 \]
\[ -x₁ + x₃ + 2x₅ - 2x₇ ≤ 16 \]
\[ x₁ + 2x₂ - x₃ + x₅ - 2x₆ - x₇ ≤ 16 \]

**Test Problem # 10**

# of Objectives: 3  
# of Decision variables: 8  
# of Constraints: 8

**Objectives**

\[ f₁ = 2x₁ + 5x₂ - x₃ + x₄ + 6x₅ + 8x₆ + 3x₇ - 2x₈ \]
\[ f₂ = 5x₁ - 2x₂ + 5x₃ + 6x₅ + 7x₆ + 2x₇ + 6x₈ \]
\[ f₃ = x₁ + x₂ + x₃ + x₄ + 3x₅ + x₆ + x₇ + x₈ \]

**Constraints**

\[ x₁ + 3x₂ - 4x₃ + x₄ - x₅ + x₆ + x₇ + x₈ ≤ 40 \]
\[ 5x₁ - 2x₂ + 5x₃ + 6x₅ + 7x₆ + 2x₇ + 6x₈ ≤ 84 \]
\[ 4x₂ - x₃ - x₄ - 3x₅ + x₈ ≤ 18 \]
\[ -3x₁ - 4x₂ + 8x₃ + 2x₄ + 3x₅ - 4x₆ + 5x₇ - x₈ ≤ 100 \]
\[ 12x₁ + 8x₂ - x₃ + 4x₄ + x₆ + x₇ ≤ 40 \]
\[ x₁ + x₂ + x₃ + x₄ + 3x₅ + x₆ + x₇ + x₈ ≥ 12 \]
\[ 8x₁ - 12x₂ - 3x₃ + 4x₄ - x₅ ≤ 30 \]
\[ -5x₁ - 6x₂ + 12x₃ + x₄ - x₇ + x₈ ≤ 100 \]
Test Problem  # 11
# of Objectives: 3
# of Decision variables: 6
# of Constraints: 10

Objectives
\[ f_1 = -0.225x_1 - 2.2x_2 - 0.8x_3 - 0.1x_4 - 0.05x_5 - 0.26x_6 \]
\[ f_2 = -10x_1 - 20x_2 - 120x_3 \]
\[ f_3 = -24x_1 - 27x_2 - 15x_4 - 1.1x_5 - 52x_6 \]

Constraints
\[ 720x_1 + 107x_2 + 7080x_3 + 134x_4 + 1000x_5 \geq 5000 \]
\[ 0.2x_1 + 10.1x_2 + 13.2x_3 + 0.75x_4 + 0.15x_5 + 1.2x_7 \geq 12.5 \]
\[ 344x_1 + 460x_2 + 1040x_3 + 75x_4 + 17.4x_5 + 240x_6 \geq 2500 \]
\[ 18x_1 + 151x_2 + 78x_3 + 2.5x_4 + 0.2x_5 + 4x_6 \geq 63 \]
\[ x_1 \leq 6, \ x_2 \leq 1, \ x_3 \leq 0.25, \ x_4 \leq 10, \ x_5 \leq 10, \ x_6 \leq 4 \]

Test Problem  # 12
# of Objectives: 5
# of Decision variables: 4
# of Constraints: 7

Objectives
\[ f_1 = x_3 \]
\[ f_2 = x_4 \]
\[ f_3 = x_3 + x_4 \]
\[ f_4 = 2x_3 + x_4 \]
\[ f_5 = x_3 + 2x_4 \]

Constraints
\[ 2x_1 + x_2 \geq 4 \]
\[ 3x_1 + 4x_2 \leq 24 \]
\[ -x_1 + x_2 \leq 10 \]
\[ x_1 + 2x_2 \leq 40 \]
\[ 5x_1 + x_2 \leq 60 \]
\[ -x_1 - 80x_1 + x_3 = 1000 \]
\[ x_1 + 6x_2 + x_4 = 1000 \]
Test Problem # 13

# of Objectives: 5
# of Decision variables: 8
# of Constraints: 8

Objectives

\[
\begin{align*}
  f_1 &= 3x_1 - 7x_2 + 4x_3 + x_4 - x_6 - x_7 + 8x_8 \\
  f_2 &= 2x_1 + 5x_2 - x_3 + x_4 + 6x_5 + 8x_6 + 3x_7 - 2x_8 \\
  f_3 &= 5x_1 - 2x_2 + 5x_3 + 6x_5 + 7x_6 + 2x_7 + 6x_8 \\
  f_4 &= 4x_2 - x_3 - x_4 - 3x_5 + x_8 \\
  f_5 &= x_1 + x_2 + x_3 + x_4 + 3x_5 + x_6 + x_7 + x_8
\end{align*}
\]

Constraints

\[
\begin{align*}
  x_1 + 3x_2 - 4x_3 + x_4 - x_5 + x_6 + x_7 + x_8 & \leq 40 \\
  5x_1 - 2x_2 + 5x_3 + 6x_5 + 7x_6 + 2x_7 + 6x_8 & \leq 84 \\
  4x_2 - x_3 - x_4 - 3x_5 + x_8 & \leq 18 \\
  -3x_1 - 4x_2 + 8x_3 + 2x_4 + 3x_5 - 4x_6 + 5x_7 - x_8 & \leq 100 \\
  12x_1 + 8x_2 - x_3 + 4x_4 + x_6 + x_7 & \leq 40 \\
  x_1 + x_2 + x_3 + x_4 + 3x_5 + x_6 + x_7 + x_8 & \geq 12 \\
  8x_1 - 12x_2 - 3x_3 + 4x_4 - x_5 & \leq 30 \\
  -5x_1 - 6x_2 + 12x_3 + x_4 - x_7 + x_8 & \leq 100
\end{align*}
\]
Appendix IV: The QXMOLP Program for the Quasiconvex Utility-based MOLP algorithm

program QXMOLP (input,output);

(This program implements the interactive Branch and Bound multiobjective linear programming algorithm for quasiconvex utility functions.)

const
  maxobj = 2;
  maxconst = 7;
  maxconstpobj = 9;
  maxdec = 10;
  maxdecpobj = 12;
  epsoln = 0.1;
  alpha = 0.5;
  yepsoln = 0.1;

type
  tablue = array [0..maxconstpobj,0..maxdecpobj] of real;
  decvector = array [1..maxdec] of real;
  objmatrix = array [1..maxobj,1..maxdecpobj] of real;
  objvector = array [1..maxobj] of real;
  vertex = record
    tab : tablue;
    vector : decvector
  end;
  dnode = record
    tab : tablue;
    vector, projvector : decvector;
    objectives : objvector;
  end;
  pnode = record
    vector : decvector
  end;
treeptr = ^treenode;
treenode = record
    data : pnode;
    next : treeptr
end;
termprtr = ^termnnode;
termnnode = record
    data : dnode;
    next : termprtr
end;

var
tree : treeptr;
terminal,vstarptr : termprtr;
kobj,m,n,n0,nplusk,mplusk,knum,ycl,iter : integer;
A,tablp : table;
obj : objmatrix;
f : text;
vstar : vertex;
done : boolean;
objmat : objmatrix;
ch : char;

procedure initobjmat;
var
  i,j : integer;
begin
  for i := 1 to maxobj do
    begin
      for j := 1 to maxdecpobj do objmat[i,j] := 0.0;
    end;
end;

procedure inittab(var tab:table);
var
  i,j : integer;
begin
  for i := 0 to maxconstpobj do
    begin

for  j := 0  to  maxdecobj  do  tab[i,j] := 0.0;
    end
end;

procedure printtab(tab : table);
    var
        x : real;
        i,j : integer;
    begin
        writeln ('m = ',m,'  n = ',n);
        writeln;
        for j := 1  to  nplusk  do  write(j:4,' ');
        writeln('  l  RHS ');
        write('----------------------------------------------------------');
        writeln('----------------------------------------------------------');
        for i := 1  to  m  do
            begin
                for j := 1  to  nplusk  do
                    begin
                        x := tab[i,j];
                        write (x:5:2);
                    end;
                x := tab[i,0];
                writeln(' ',x:5:2)
            end;
        writeln('----------------------------------------------------------');
        writeln('----------------------------------------------------------');
        for i := m+1  to  mplusk  do
            begin
                for j := 1  to  nplusk  do
                    begin
                        x := tab[i,j];
                        write (x:5:2);
                    end;
                x := tab[i,0];
                writeln(' ',x:5:2);
            end;
        writeln('----------------------------------------------------------');
        writeln('----------------------------------------------------------');
for \ j := 1 \ to \ nplusk \ do
\begin{align*}
&\text{begin} \\
&\quad x := \text{tab}[0,j]; \\
&\quad \text{write}(x:5:2) \\
&\quad \text{end}; \\
&\quad x := \text{tab}[0,0]; \\
&\quad \text{write}(\ 'l',x:5:2); \\
&\quad \text{writeln}; \ \text{writeln} \\
&\text{end};
\end{align*}

\text{procedure getmodel (var tab : table);} \\
\begin{align*}
&\text{var} \\
&\quad x,y,b : \text{real}; \\
&\quad dr,i,j,k : \text{integer}; \\
&\text{begin} \\
&\quad \text{initobjmat;} \\
&\quad \text{readln}(f,kobj,n,m); \\
&\quad \text{writeln('NUMBER OF OBJECTIVES: ',kobj);} \\
&\quad \text{writeln('NUMBER OF DECISION VARIABLES: ',n);} \\
&\quad \text{writeln('NUMBER OF CONSTRAINTS: ',m);} \\
&\quad \text{writeln;} \\
&\quad \text{writeln('THE OBJECTIVE MATRIX: ');} \\
&\text{for i := 1 to kobj do} \\
&\quad \text{begin} \\
&\quad \quad \text{for j := 1 to n do} \\
&\quad \quad \quad \text{begin} \\
&\quad \quad \quad \quad \text{read}(f,x); \\
&\quad \quad \quad \quad \text{write}(x:5:2); \\
&\quad \quad \quad \quad \text{objmat[i,j]} := x; \\
&\quad \quad \quad \text{end;} \\
&\quad \quad \text{writeln;} \\
&\quad \text{end;} \\
&\quad \text{writeln;} \\
&\quad \text{tab[0,n+1]} := -1.00; \\
&i := 0; \\
&\text{for i := 1 to m do} \\
&\quad \text{begin} \\
&\quad \quad y := 0.0; \\
&\text{end;}
for j := 1 to n do 
begin 
   read (f,x);
   write (x:5:2);
   y := y + x*x;
   tab[i,j] := x 
end;

y := sqrt (y);
ycl := n+1;
tab [i,ycl] := y;
write(y:5:2);
tab[i,n+1+i] := 1.0;
read (f,dr);
write (dr:5,' ');
if (dr = -1) then 
begin 
   tab[i,n+1+i] := -1.0;
end;
readln (f,b);
write(b:5:2);
writeln;
tab[i,0] := b 
end;
n0 := n + 1;
n := n + 1 + m;
nplusk := n + kobj;
mpusk := m + kobj;
knnum := 1;
for i := m+1 to mplusk do 
begin 
   for j := 1 to n0-1 do 
      tab[i,j] := -1*objmat[i-m,j];
   tab[i,n+knum] := 1;
   knnum := knnum + 1;
end;
end;
procedure pivot(var tab:tablue; p,q:integer);
begin { pivot }
    j,k : integer;
    for j := 0 to mplusk do
        for k := nplusk downto 0 do
            if (j <> p) and (k <> q) then
                tab[j,k] := tab[j,k]-(tab[p,k]*tab[j,q]/tab[p,q]);
    for j := 0 to mplusk do
        if j <> p then tab[j,q] := 0;
    for k := 0 to nplusk do
        if k <> q then tab[p,k] := tab[p,k]/tab[p,q];
    tab[p,q] := 1
end; { pivot }

procedure lpsimplex(var tab:tablue);
begin { lpsimplex }
    j := 1;
    repeat
        q := 0;
            { repeat q := q+1 until (q=n+1) or (tab[0,q]<0); }
        i := 1;
        q := 1;
        while (i < n) do
            begin
                if tab[0,i+1] < tab[0,q] then q := i+1;
                i := i + 1
            end;
        if tab[0,q] >= 0.0 then q := n+1;
    p := 0;
    repeat p := p+1 until (p=m+1) or (tab[p,q]>0);
    for i := p+1 to m do
        if tab[i,q] > 0 then
            begin
                if (tab[i,0]/tab[i,q] < tab[p,0]/tab[p,q]) then p := i;
            end;
    if (q <= n) and (p <= m) then
        begin
            pivot(tablp,p,q)
end;
    j := j + 1;
    until (q=n+1) or (p=m+1)
end;  { lpsimplex }

procedure usersimplex(var tab:tablue; q:integer);
 var
   p,i,j : integer;

begin { usersimplex }
    p := 0;
    repeat p:=p+1 until (p=m+1) or (tab[p,q]>0);
    for i := p+1 to m do
        begin
            if tab[i,q] > 0 then
                if (tab[i,0]/tab[i,q] < tab[p,0]/tab[p,q]) then p:=i
            end;
            if (q<n+1) and (p<m+1) then pivot(tab,p,q);
    end;  { usersimplex }

procedure getdecvector(tab:tablue; var vec : decvector);
 var
   i,j,ic,k,n0,n1 : integer;

begin
   for k := 1 to maxdec do vec[k] := 0.0;
   for j := 1 to n do
        begin
            n0 := 0;
            n1 := 0;
            i := 1;
            while ( i <= m ) do
                begin
                    if (tab[i,j] = 1.00) then
                        begin
                            ic := i;
                            n1 := n1 + 1;
                            end;
                    if (tab[i,j] = 0.00) then n0 := n0 + 1;
                    i := i + 1
                end;
if (n1 = 1) and (n0 = m-1) then vec[j] := tab[ic,0];
end
end;

procedure getvertex(tab:table; var v : vertex);
begin
  v.tab := tab;
  getdecvector(v.tab,v.vector)
end;

procedure printvector(vec : decvector);
var
  i : integer;
  x : real;
begin
  write('(');
  x := vec[1];
  write(x:1:2);
  for i := 2 to n0 do
    begin
      x := vec[i];
      write(',x:1:2)
      end;
    write(')');
  writeln
end;

procedure printvertex(v:vertex);
begin
  printtab(v.tab);
  printvector(v.vector)
end;

procedure addtreenode(vec:decvector);
var
  i,j : treeptr;
begin
  i := nil;
  j := nil;
i := tree;
if tree = nil then
  begin
    new(tree);
    tree^.data.vector := vec;
    tree^.next := nil
  end
else
  begin
    while ((i^.next) <> nil) do  i := i^.next;
    new(j);
    j^.data.vector := vec;
    j^.next := nil;
    i^.next := j;
  end;
end;

procedure initlists;
begin
  tree := nil;
  terminal := nil;
  addtreenode(vstar.vector);
end;

function nonbasic(tab:table; q: integer): boolean;
var
  i,j,ic,k,n0,n1: integer;
begin
  nonbasic := true;
  n0 := 0;
  n1 := 0;
  i := 1;
  while (i <= m) do
  begin
    if (tab[i,q] = 1.00) then
      begin
        n1 := n1 + 1;
      end;
    if (tab[i,q] = 0.00) then n0 := n0 + 1;
  end;
end;
i := i + 1
end;
if ( (n1 = 1) and (n0 = m-1) ) then nonbasic := false
end;

procedure checkeligible (tab:tablue; y,j:integer; var p:integer);
var
  i : integer;
begin
  p := 0;
i := 1;
  while (p = 0) and (i <= m) do
  begin
    if tab[i,y] = 1.00 then p := i;
i := i + 1
  end;
  if (tab[p,j] <= 0.0) then p := 0;
end;

function same(v1,v2:decevector):boolean;
var
  i : integer;
  yes : integer;
begin
  same := true;
  yes := 1;
i := 1;
  while (yes = 1) and (i <= n) do
  begin
    if abs(v1[i] - v2[i]) >= epsoln then yes := 0;
i := i + 1;
  end;
  if yes = 0 then same := false;
end;

function found(v : decevector):boolean;
var
  yes : integer;
i : treeptr;
j : temptr;
vtemp : decvector;
begin
found := false;
i := nil;

i := tree;
yes := 0;
while (yes <> 1) and (i <> nil) do
begin
if same(v,i^.data.vector) then yes := 1;
i := i^.next;
end;
if yes <> 1 then
begin
j := nil;
j := terminal;
while (yes <> 1) and (j <> nil) do
begin
if same(v,j^.data.vector) then yes := 1;
j := j^.next;
end;
end;
if yes = 1 then found := true;
end;

procedure project(var tab:tablue; p,q:integer);
begin
pivot(tab,p,q);
end;

procedure getobjectives(vproject:decvector; var fvector:objvector);
var
i,j : integer;
sum : real;
begin
for i := 1 to kobj do
begin
sum := 0.0;
for j := 1 to n do sum := sum + objmat[i,j]*vproject[j];
    fvector[i] := sum;
end;
end;

procedure printobjvector(fvector:objvector);
var
    i : integer;
    x,u : real;
begin
    write('(');
    x := fvector[1];
    write(x:1:2);
    for i := 2 to kobj do
    begin
        x := fvector[i];
        write(' ,x:1:2);
        end;
    writeln(')');
end;

procedure addtermnode(tab:tabule;vj,vp:decvector; fvector: objvector );
var
    temp : termptr;
begin
    temp := nil;
    new(temp);
    temp^.data.tab := tab;
    temp^.data.vector := vj;
    temp^.data.projvector := vp;
    temp^.data.objectives := fvector;
    temp^.next := terminal;
    terminal := temp;
    temp := nil;
end;
function dominated(f:objvector):boolean;
var
  i : termptr;
  yes : boolean;
  k,more,equal : integer;
begin
  dominated := false;
i := nil;
i := terminal;
yes := false;
while (not yes) and (i <> nil) do
begin
  yes := false;
  more := 0;
  equal := 0;
  for k := 1 to kobj do
  begin
    if (i^.data.objectives[k] > f[k]) then more := more + 1
    else if (i^.data.objectives[k] = f[k]) then equal := equal + 1;
  end;
  if ((more + equal) = kobj) and (more >= 1) then yes := true;
i := i^.next;
end;
if (yes) then dominated := true;
end;

procedure simulate_utility(f:objvector; var u:real);
var
  i : integer;
begin
  u := 0.0;
  for i := 1 to kobj do u := u + exp(alpha*f[i]);
end;

procedure display_termlist;
var
  i : termptr;
  count,j : integer;
  tab : table;
  vj,vp : decvector;
fobj : objvector;
x,u : real;
begin
  i := nil;
count := 0;
i:= terminal;
  writeln; writeln;
  writeln('PLEASE LOOK AT THE FOLLOWING SAMPLE OF
          ALTERNATIVES:');
  writeln('----------');
while i <> nil do
begin
  count := count + 1;
  fobj := i^.data.objectives;
  if not dominated(fobj) then
begin
  tab := i^.data.tab;
  vj := i^.data.vector;
  vp := i^.data.projvector;
  write(' # ',count:2);
  if vj[ycl] <= yepsoln then write(' REAL ')
else write(' DUMMY ');
  write('(',fobj[1]:1:2);
  for j := 2 to kobj do write(' ',fobj[j]:1:2);
  write(')');
  simulate_utility(fobj,u);
  writeln(' Utility = ',u:1:1);
end;
i := i^.next;
end;
writeln('______________');
writeln;
end;

procedure branch_project(tabstar:tablue);
var
  j,p : integer;
  tabadj,tabproj : tablue;
tempproject : dnode;
yes : boolean;
vadj,vproj : decvector;
obj : objvector;
begin
for j := 1 to n do
begin
   tabadj := tabstar;
   if nonbasic(tabstar,j) then
   begin
      usersimplex(tabadj,j);
      {printtab(tabadj)}
      checkeligible(tabstar,ycl,j,p);
      if p <> 0.0 then
      begin
         getdecvector(tabadj,vadj);
         if not found(vadj) then
         begin
            tabproj := tabstar;
            project(tabproj,p,j);
            getdecvector(tabproj,vproj);
            getobjectives(vproj,obj);
            addtermnode(tabadj,vadj,vproj,obj);
         end
      end
   end
end
end;

procedure check_termination(best:integer);
var
   i : termptr;
   count : integer;
y : real;
begin
   vstarptr := nil;
i := nil;
i := terminal;
count := 1;
while count < best do
begin
   i := i^.next;
count := count + 1;
end;
y := i^.data.vector[ycl];
if ( y <= epsoln ) then done := true else vstarptr := i;
if done then
  begin
    writeln;
    writeln('THE MOST-PREFERRED SOLUTION AND ALTERNATIVE ARE ACHIEVED:');
    write('THE MOST-PREFERRED SOLUTION: ');
    printvector(i^.data.vector);
    write('THE MOST-PREFERRED ALTERNATIVE: ');
    printobjvector(i^.data.objectives);
    writeln; writeln;
  end;
end;

procedure removevstar;
var
  i : temptr;
begin
  i := nil;
  i := terminal;
  while i^.next <> vstarptr do i := i^.next;
i^.next := vstarptr^.next;
vstarptr^.next := nil;
end;

procedure test_bound;
var
  best : integer;
begin
  display_termlist;
  writeln('IF THERE IS ONLY ONE ALTERNATIVE, ENTER ITS NUMBER:');
  writeln('OTHERWISE, ENTER THE NUMBER FOR THE ONE YOU PREFER:');
  write('SELECT ALTERNATIVE # '); read(best); check_termination(best);
if not done then
begin
  vstar.tab := vstarptr^data.tab;
  vstar.vector := vstarptr^data.vector;
  addtreenode(vstar.vector);
  removevstar;
end;
end;

begin {main}
  assign (f,'a:molph.pas');
  reset (f);
  writeln;
  write('          HIT THE RETURN KEY TO START ');
  read(ch);
  writeln;
  writeln('           THIS IS A RUN FOR A TEST PROBLEM');
  writeln;
  inittabl(tablp);
  getmodel(tablp);
  close(f);
  lpsimplex(tablp);

      { we need to branch and project form vstar, bound and test till
termination.  }

  getvertex (tablp,vstar);
  initlists;
  done := false;
  iter := 1;
  while (not done) do
begin
  writeln; writeln;
  writeln(' iteration # ',iter);
  writeln('----------');
  branch_project(vstar.tab);
  test_bound;
  iter := iter + 1;
end
end.  {main}
Appendix IV: Test Problems for the Quasiconvex Utility-based MOLP Algorithm.

Test Problem # 1
# of Objectives: 2
# of Decision variables: 2
# of Constraints: 3
Objectives
\[ f_1 = x_1 \]
\[ f_2 = x_2 \]
Constraints
\[ x_1 + x_2 \leq 2 \]
\[ x_i \geq 0, \quad i = 1, 2. \]

Test Problem # 2
# of Objectives: 2
# of Decision variables: 3
# of Constraints: 4
Objectives
\[ f_1 = 4x_1 + 2x_2 + x_3 \]
\[ f_2 = -4x_1 -3x_2 + -2x_3 \]
Constraints
\[ x_1 + x_2 + x_3 \leq 1 \]
\[ x_i \geq 0, \quad i = 1, \ldots, 3. \]

Test Problem # 3
# of Objectives: 2
# of Decision variables: 2
# of Constraints: 4
Objectives

\[ f_1 = 0.4x_1 + 0.3x_2 \]
\[ f_2 = x_1 \]

Constraints

\[ x_1 + x_2 \leq 400 \]
\[ 2x_1 + x_2 \leq 500 \]
\[ x_i \geq 0, \quad i = 1, 2. \]

Test Problem # 4

# of Objectives: 3
# of Decision variables: 4
# of Constraints: 6

Objectives

\[ f_1 = 3x_1 + x_2 + 2x_3 + x_4 \]
\[ f_2 = x_1 - x_2 + 2x_3 + 4x_4 \]
\[ f_3 = -x_1 + 5x_2 + x_3 + 2x_4 \]

Constraints

\[ 2x_1 + x_2 + 4x_3 + 3x_4 \leq 60 \]
\[ 3x_1 + 4x_2 + x_3 + 2x_4 \leq 60 \]
\[ x_i \geq 0, \quad i = 1, \ldots, 4. \]

Test Problem # 5

# of Objectives: 3
# of Decision variables: 3
# of Constraints: 6

Objectives

\[ f_1 = 4x_1 + x_2 + 2x_3 \]
\[ f_2 = x_1 + 3x_2 - x_3 \]
\[ f_3 = -x_1 + x_2 + 4x_3 \]

Constraints

\[ x_1 + x_2 + x_3 \leq 3 \]
\[ x_1 + 2x_2 + x_3 \leq 4 \]
\[ x_1 - x_2 \leq 0 \]
\[ x_i \geq 0, \quad i = 1, \ldots, 3. \]

Test Problem # 6

# of Objectives: 3
# of Decision variables: 4
# of Constraints: 7

Objectives
\[ f_1 = 10x_1 + 30x_2 + 50x_3 + 100x_4 \]
\[ f_2 = x_1 + x_2 \]
\[ f_3 = x_1 + 4x_2 + 6x_3 + 2x_4 \]

Constraints
\[ 5x_1 + 3x_2 + 2x_3 + x_4 \leq 420 \]
\[ 3x_1 + 8x_4 \leq 320 \]
\[ 2x_1 + 3x_2 + 4x_3 + 6x_4 \leq 180 \]
\[ x_i \geq 0, \quad i = 1, \ldots, 4. \]

Test Problem # 7

# of Objectives: 3
# of Decision variables: 3
# of Constraints: 7

Objectives
\[ f_1 = x_1 \]
\[ f_2 = x_2 \]
\[ f_3 = x_3 \]

Constraints
\[ 3x_1 + 2x_2 + 3x_3 \leq 18 \]
\[ x_1 + 2x_2 + x_3 \leq 10 \]
\[ 9x_1 + 20x_2 + 7x_3 \leq 96 \]
\[ 7x_1 + 20x_2 + 9x_3 \leq 96 \]
\[ x_i \geq 0, \quad i = 1, \ldots, 3. \]
Test Problem # 8
# of Objectives: 3
# of Decision variables: 7
# of Constraints: 11

Objectives
\[ f_1 = x_2 + x_3 + x_4 + 3x_5 + x_6 \]
\[ f_2 = x_1 + x_3 - x_4 - x_6 \]
\[ f_3 = x_1 + 2x_2 - x_3 + 3x_4 + 2x_5 + x_7 \]

Constraints
\[ x_1 + 2x_2 + x_3 + x_4 + 2x_5 + x_6 + 2x_7 \leq 16 \]
\[ -2x_1 - x_2 + x_4 + 2x_5 + 2x_7 \leq 16 \]
\[ -x_1 + x_3 + 2x_5 - 2x_7 \leq 16 \]
\[ x_1 + 2x_2 - x_3 + x_5 - 2x_6 - x_7 \leq 16 \]
\[ x_i \geq 0, \quad i = 1, \ldots, 7. \]

Test Problem # 9
# of Objectives: 3
# of Decision variables: 6
# of Constraints: 16

Objectives
\[ f_1 = -0.225x_1 - 2.2x_2 - 0.8x_3 - 0.1x_4 - 0.05x_5 - 0.26x_6 \]
\[ f_2 = -10x_1 - 20x_2 - 120x_3 \]
\[ f_3 = -24x_1 - 27x_2 - 15x_4 - 1.1x_5 - 52x_6 \]

Constraints
\[ 720x_1 + 107x_2 + 7080x_3 + 134x_4 + 1000x_5 \geq 5000 \]
\[ 0.2x_1 + 10.1x_2 + 13.2x_3 + 0.75x_4 + 0.15x_5 + 1.2x_7 \geq 12.5 \]
\[ 344x_1 + 460x_2 + 1040x_3 + 75x_4 + 17.4x_5 + 240x_6 \geq 2500 \]
\[ 18x_1 + 151x_2 + 78x_3 + 2.5x_4 + 0.2x_5 + 4x_6 \geq 63 \]
\[ x_1 \leq 6, \quad x_2 \leq 1, \quad x_3 \leq 0.25, \quad x_4 \leq 10, \quad x_5 \leq 10, \quad x_6 \leq 4 \]
\[ x_i \geq 0, \quad i = 1, \ldots, 6. \]
Test Problem # 10

# of Objectives: 5
# of Decision variables: 4
# of Constraints: 11

Objectives
\[ f_1 = x_3 \]
\[ f_2 = x_4 \]
\[ f_3 = x_3 + x_4 \]
\[ f_4 = 2x_3 + x_4 \]
\[ f_5 = x_3 + 2x_4 \]

Constraints
\[ 2x_1 + x_2 \geq 4 \]
\[ 3x_1 + 4x_2 \leq 24 \]
\[ -x_1 + x_2 \leq 10 \]
\[ x_1 + 2x_2 \leq 40 \]
\[ 5x_1 + x_2 \leq 60 \]
\[ -x_1 - 80x_2 + x_3 \leq 100 \]
\[ x_1 + 6x_2 + x_4 \leq 100 \]
\[ x_i \geq 0, \ i = 1, \ldots, 4. \]