IIASA

International Institute for Applied Systems Analysis • A-2361 Laxenburg • Austria
Tel: +43 2236 807 • Fax: +43 2236 71313 • E-mail: info@iiasa.ac.at • Web: www.iiasa.ac.at

INTERIM REPORT IR-98-011 /March

An Algorithm for Projecting a Reference Direction onto the Nondominated Set of Given Points

Pekka Korhonen(korhonen@iiasa.ac.at)
Jasmina Karaivanova

Approved by
Gordon MacDonald(macdon@iiasa.ac.at)
Director, *IIASA*

Contents

Abstract

In this paper, we consider the problem of searching nondominated alternatives in a discrete multiple criteria problem. The search procedure is based on the use of a reference direction. A reference direction reflects the desire of the decision maker (DM) to specify a search direction. To find a set of given alternatives related somehow to the reference direction specified by the DM, the reference direction has to be projected onto the set of nondominated alternatives. Our purpose is to develop an efficient algorithm for making this projection. The projection of each given reference direction determines a nondominated ordered subset. The set is provided to a decision maker for evaluation. The decision maker will choose the most preferred alternative from this subset and continues the search from this alternative with a new reference direction. The search will end when no direction of improvement is found. A critical point in the procedure is the efficiency of the projection operation. This efficiency of our algorithm is considered theoretically and numerically.

The projection is made by parametrizing an achievement scalarizing function originally proposed by Wierzbicki (1980) to project any single point onto the nondominated set.

Keywords: Multiple Criteria, Discrete, Evaluation, Reference Point, Reference Direction.

Acknowledgments

The research was supported, in part, by grants from the Academy of Finland, the Foundation for Economic Education, and the Foundation of the Helsinki School of Economics, Finland.

About the Authors

Pekka Korhonen is Project Leader of the Decision Analysis and Support Project at IIASA, and also Professor of Statistics at the Department of Economics and Management Science, Helsinki School of Economics and Business Administration.

More information about the author can be found at the Web site:

http://www.iiasa.ac.at/~korhonen

An Algorithm for Projecting a Reference Direction onto the Nondominated Set of Given Points

Pekka Korhonen
Jasmina Karaivanova

1. Introduction

A multiple objective optimization problem in the criterion space can be defined as follows:

$$max\ \mathbf{q} \tag{1.1}$$

$$s.t. \quad \mathbf{q} \in Q,$$

where $Q \subset \Re^p$ is a feasible region in a p-dimensional criterion space. When Q consists of a finite number of elements which are explicitly known in the beginning of the solution process, we have an important class of problems which may be called e.g.. *(Multiple Criteria) Evaluation Problems.* Those problems are sometimes referred to as *Discrete Multiple Criteria Problems* or *Selection Problems* (for survey, see, e.g. Olson 1996).

Which kind of approach is most suitable to help the decision maker (DM) to solve an Evaluation Problem is heavily dependent on the characteristics of the problem. The outranking approach (Roy 1973), the multiattribute utility theory (Keeney and Raiffa 1976), the analytic hierarchy process (Saaty 1980), the regime method (Hinloopen, Nijkamp and Rietveld 1983), the interactive programming approach (Korhonen, Wallenius and Zionts 1984), the hierarchical interactive approach (Korhonen 1986), the visual reference direction approach (Korhonen 1988), and the aspiration-level interactive method (AIM) (Lotfi, Stewart, and Zionts 1992) are typical examples of the approaches developed to solve evaluation problems, but not the same kind of problems. A simple cross-tabulation can be made on the basis of the number of criteria and alternatives (**Table 1**). However, there are many other ways to classify the problems as

well. For instance, in the outranking approach by Roy [1973], there is no need to explicitly specify the criteria at all, the multiattribute utility theory by Keeney and Raiffa [1976] is able to deal with uncertainty in criterion values, and the analytic hierarchy process by Saaty [1980] is developed to handle the hierarchical structure of criteria.

Table 1: **A Classification of Multiple Criteria Evaluation Methods**

	# of Criteria	
# of Alternatives	Small	Large
Small	Roy [1973], Keeney and Raiffa [1976], Hinloopen, Nijkamp and Rietveld [1983]	Saaty [1980], Korhonen [1985]
Large	Korhonen, Wallenius and Zionts [1984], Korhonen [1988], Lotfi, Stewart, and Zionts [1992]	

A key issue in *Multiple Criteria Decision Support* (MCDS) is to generate nondominated solutions to the DM's evaluation in such a way that the DM's preferences are taken into account. In interactive methods, the system will communicate with the DM and information the system will get through this communication process is used to model his/her preferences. The preference information is further used to control the search process. The ultimate goal is to help the DM to find the most preferred alternatives from among the set of nondominated alternatives. During the last twenty-five years, a lot of various approaches to solve multiple criteria problems have been developed. For an excellent introduction to multiple objective programming methods, see Steuer [1986]. Correspondingly, Olson [1996] provides a good introduction to various multiple criteria evaluation methods.

In early methods, the generation of nondominated solutions was based on the use of criterion weights, limiting consideration to nondominated extreme points (see, e.g., Zionts and Wallenius [1976]) in multiple objective linear program and to nondominated alternatives which can be obtained as an optimum of a linear function in multiple criteria evaluation problems. Today, many systems are based on the use of aspiration level (or reference point) projections, where the projection is performed using the Chebyshev-type achievement scalarizing functions. These functions can be controlled either by varying weights (keeping aspiration levels fixed) or by varying the aspiration levels (keeping weights fixed). The same idea was originally proposed in a somewhat different form by Steuer and Choo [1983] and Wierzbicki [1980].

The *Achievement Scalarizing Function* is also the main theoretical basis in the *Reference Direction Approach* proposed by Korhonen and Laakso [1986]. Instead of projecting one point at a time onto the nondominated frontier - like in the original reference point approach by Wierzbicki [1980] - Korhonen and Laakso [1986] proposed the parametrization of an achievement scalarizing function making it possible to project the

whole direction onto the nondominated frontier. The problem can be easily solved by using linear parametric programming. When a reference direction is projected onto the nondominated frontier in multiple objective linear programming, a curve traversing across the frontier is obtained. Then an interactive line search is performed along this curve. The idea enables the DM to make a continuous search on the nondominated frontier.

When the reference direction approach is extended to evaluation problems, the projection of a reference direction onto the set of nondominated alternatives requires the solving of an integer parametric programming problem. Because the problem has a special structure, it is possible to develop an efficient algorithm for the problem. The first algorithm proposed by Korhonen [1988]. The aim of this paper is to present an improved version. The algorithm presented in this paper is much more sophisticated than the previous one. The algorithm is capable of solving the problems of many hundred alternatives in a few seconds, whereas the old version required even minutes.

The plan of this paper is as follows. In section 2 some preliminary considerations are given. The algorithm for solving discrete Multiple Criteria Problems is presented in section 3. We conclude the paper in section 4.

2. Preliminary considerations

2.1. Problem Formulation and Basic Definitions

We assume that a single decision maker (DM) has to choose one alternative from among a set of n (n>0) explicitly defined deterministic decision alternatives using p (p>1) (quantitative) criteria, which the DM would like to maximize. The basic data set can be given in an n×p matrix $A \in \Re^{n\times p}$ whose elements a_{ij}, $i \in I = \{1,2, \ldots, n\}$ and $j \in J= \{1,2, \ldots, p\}$ represent the criterion values on alternatives. We use notation $\mathbf{a}_i$ or simply index i to refer to the alternative in row i. Thus each decision alternative is a point in the criterion space $\Re^p$. When we refer to the index of the element, for which $f_k = \max_{i\in I} f_i$, we use notation $\arg\max_{i\in I} f_i$. (Note that k is not necessarily uniquely defined.)

Let $\mathbf{x} \in \Re^p$ and $\mathbf{y} \in \Re^p$ be two vectors. We will use notation $\mathbf{x} \geq \mathbf{y}$ to denote that $x_i \geq y_i$ for all i = 1,2, …, n. Correspondingly, notation $\mathbf{x} > \mathbf{y}$ means that $x_i > y_i$ for all i = 1,2, …, n.

Definition 1. An alternative $\mathbf{a}_i$*, $i \in I$, is nondominated iff $\nexists\, \mathbf{a}_j$, $j \in J$, $i \neq j$ such that $\mathbf{a}_j \geq \mathbf{a}_i$ and $\mathbf{a}_j \neq \mathbf{a}_i$.

Definition 2. An alternative $\mathbf{a}_i^*$, $i \in I$, is weakly-nondominated iff $\nexists\, \mathbf{a}_j$, $j \in J$, $i \neq j$ such that $\mathbf{a}_j > \mathbf{a}_i$.

If an alternative is not **nondominated** it is dominated.

The Multiple Criteria Problem considered in this paper can be defined as follows:

$$\text{'}max\text{'} \underset{i \in I}{\mathbf{a}_i}. \qquad (2.1)$$

Because no single alternative exists with the maximum values for all criteria, the rational DM will choose one nondominated alternative. Which alternative is chosen depends on the DM's preference structure. To be able to make a decision, he/she needs a tool to help him/her to find his/her 'best' alternatives. Inmulticriteria decision making (MCDM), the 'best' alternative is usually called the *most preferred alternative*. Conceptually, the most preferred alternative is the alternative maximizing the DM's utility at the moment, when he/she makes the final choice. In practice, we cannot guarantee that the final choice is the DM's most preferred alternative in the sense that it is really preferred to all other alternatives. How good the final choice is depends on the algorithm used to support the DM to search for the most preferred alternative.

2.2. Solution principle

Our proposal is an interactive approach, which makes it possible for the DM to make a free search among the set of nondominated alternatives until he/she is convinced that no better alternative than the current choice can be found. We do not need to assume anything about the DM's choice behavior. The DM will control the search by varying aspiration levels for criterion values. The point characterized by aspiration levels is also called a *reference point* by Wierzbicki [1980]. At each iteration, he/she will choose the most preferred alternative from among the set of alternatives available, and specify the new aspiration levels describing his/her desire to change the current criterion values. The reference direction is specified as a vector starting from the current alternative and passing through the reference point. The reference direction expresses the DM's desire to move from the current alternative, but it does not provide realistic suggestions to move.

To find concrete alternatives somehow related to the reference direction, that direction has to be projected onto the set of nondominated alternatives. The projection can be done by using an **achievement scalarizing function** developed by Wierzbicki [1980]. Originally, Wierzbicki developed the achievement scalarizing function to project any single (reference) point onto the nondominated frontier in multiple objective linear programming. Korhonen and Laakso [1986] extended the original idea by parametrizing the achievement scalarizing function. The extension made it possible to project any direction instead of a single point onto the nondominated frontier. When the method is applied to a discrete multiple criteria problem, a new (ordered) finite subset is produced

for the DM's evaluation. A visual representation (see, Korhonen 1988) can then be used to present the subset in an illustrative form for the DM's evaluation.

The simplest form of the achievement scalarizing function for each alternative is defined as follows:

$$s_i(\mathbf{g}, \mathbf{w}) = \max_{j \in J} (g_j - a_{ij})/w_j,\ i = 1, 2, \ldots, n \qquad (2.2)$$

where $\mathbf{g} \in \Re^p$ is a given reference point, and $\mathbf{w} \in \Re^p$, $\mathbf{w} > \mathbf{0}$, is a given weighting vector.

By solving the following minimizing problem

$$\min_{i \in I} s_i(\mathbf{g}, \mathbf{w}) \qquad (2.3)$$

the solution is one of the (weakly)-nondominated alternatives. The weakly-nondominated alternatives are not a problem in an evaluation problem, because dominated alternatives could be easily eliminated from a set of a finite number of alternatives.

To generate an ordered set of nondominated alternatives for the DM's evaluation, we parametrize the achievement scalarizing function:

$$S_{hi}(t, \mathbf{r}, \mathbf{w}) = s_i(\mathbf{a}_h + t\mathbf{r}, \mathbf{w}) = \max_{j \in J} (a_{hj} + tr_j - a_{ij})/w_j\ , \qquad (2.4)$$

where $\mathbf{a}_h$, $h \in I$, is a current alternative and $\mathbf{r}$, $\mathbf{r} \in \Re^p$, is a given reference direction. The alternatives solving the problem

$$\min_{i \in I} S_{hi}(t, \mathbf{r}, \mathbf{w}), \qquad (2.5)$$

when $t: 0 \rightarrow \infty$ define an ordered set of nondominated decision alternatives. The set containing the indices of those alternatives is denoted by $M = \{m_1, m_2, \ldots, m_k\}$, $k \leq n$.

Definition 3. Alternative $\mathbf{a}_\alpha \in$ M iff $\alpha = \underset{i \in I}{arg\ min}\ S_{hi}(t, \mathbf{r}, \mathbf{w})$ for some $t \geq 0$ and if $\mathbf{a}_\alpha$ precedes alternative $\mathbf{a}_\beta$, $\alpha \neq \beta$, in set M then there exists $t_\alpha \geq 0$ for which $\alpha = \underset{i \in I}{arg\ min}\ S_{hi}(t_\alpha, \mathbf{r}, \mathbf{w})$ such that $t_\alpha \leq t_\beta$ for all $t_\beta \geq 0$ for which $\beta = \underset{i \in I}{arg\ min}\ S_{hi}(t_\beta, \mathbf{r}, \mathbf{w})$.

It is easy to see that the solutions of problem (2.5) are solutions of the following parametric programming problems:

$$
\begin{aligned}
&\textit{min} && \varepsilon \\
&\text{s.t.} && \\
&&& A^T\mathbf{x} + \varepsilon\mathbf{w} - \mathbf{z} = \mathbf{a}_h + t\mathbf{r}, \\
&&& \mathbf{1}^T\mathbf{x} = 1 \\
&&& x_i = 0 \text{ or } 1, \text{ for } i = 1,2, ..., n \\
&&& \mathbf{x}, \mathbf{z} \geq 0,
\end{aligned}
\tag{2.6}
$$

when $t: 0 \rightarrow \infty$, where $\mathbf{1} = [1, 1, ..., 1]^T \in \Re^n$. Like in parametric programming, in general our task is to identify a) the parameter values with which the bases change and b) the entering and leaving variable. However, the special structure of problem (2.6) makes it possible to develop algorithms, which are much more efficient than general integer parametric programming.

Our algorithm is based on the idea to operate with functions $V_i(t)$, i = 1,2, …, n, $t \geq 0$:

$$V_i(t) = S_{hi}(t, \mathbf{r}, \mathbf{w}) = \underset{j \in J}{max}\ (a_{hj} + tr_j - a_{ij})/w_j\ , \tag{2.7}$$

where $\mathbf{a}_h$, $\mathbf{r}$, $\mathbf{w}$ are taken as given. If we denote $b_{ij} = (a_{hj} - a_{ij})/w_j$, i = 1,2, …, n and j = 1,2, …, p, and $d_j = r_j/w_j$, j = 1,2, …, p, so we may simply write

$$V_i(t) = \underset{j \in J}{max}\ (b_{ij} + td_j). \tag{2.8}$$

Thus our problem is to identify all indices $k \in I$ for which

$$V_k(t) = \underset{i \in I}{min}\, V_i(t) \tag{2.9a}$$

or in other words

$$k = \underset{i \in I}{arg\ min}\, V_i(t) \text{ for some } t \geq 0. \tag{2.9b}$$

The problem is illustrated in Figure 1. The shaded area describes the values higher than the values of function $\underset{i \in I}{min}\, V_i(t)$. Our idea is to study the values of that function piece by piece and to recognize the points, where the function with a minimum value will change.

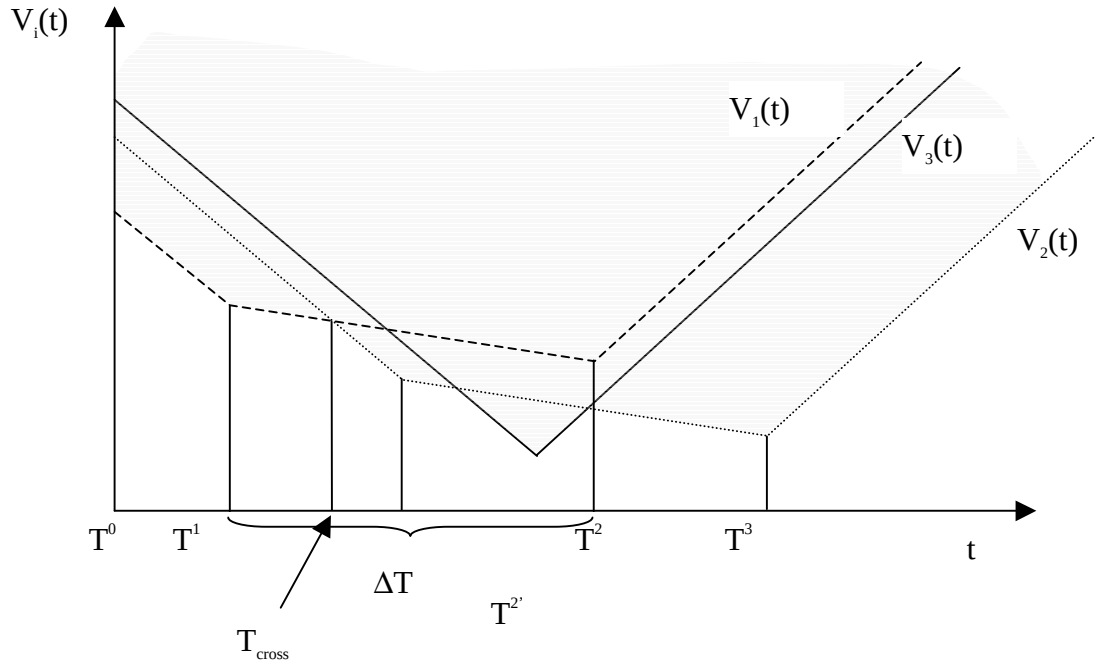

Figure 1: Illustration of function $\underset{i \in I}{min}\, V_i(t)$ and the principles of the algorithm

2.3. Some Theory

First, we consider the properties of the functions $V_i(t)$, i = 1, 2, …, n, which we can utilize in our algorithm:

Lemma 1. Consider the function $V_i(t)$, $i \in I$, and let $t_0 \geq 0$ be an arbitrary value of parameter t. If $b_{ij} + t_0 d_j \geq b_{ik} + t_0 d_k$ and $d_j \geq d_k$ for some j, $k \in J$, then $V_i(t) \geq b_{ij} + td_j \geq b_{ik} + td_k$, when $t \geq t_0$.

Proof. We see immediately that $V_i(t) = \underset{m \in J}{max} (b_{im} + td_m) \geq b_{ij} + td_j \geq b_{ij} + t_0 d_j + (t - t_0)d_j \geq b_{ik} + t_0 d_k + (t - t_0)d_j \geq b_{ik} + t_0 d_k + (t - t_0)d_k = b_{ik} + td_k$.

Corollary 1.1 If $b_{ij} + t_0 d_j > b_{ik} + t_0 d_k$ and $d_j > d_k$, then $b_{ij} + td_j > b_{ik} + td_k$, when $t \geq t_0$.

Corollary 1.2 If $V_i(t_1) = b_{ij} + t_1 d_j$ for some $j \in J$ and $t_1 \geq 0$, and $V_i(t_2) = b_{ik} + t_2 d_k$ for some, $k \in J$, $k \neq j$, $t_2 \geq 0$, and $t_2 > t_1$, then necessarily $d_k \geq d_j$. Furthermore, if $b_{ij} + t_1 d_j > b_{ik} + t_1 d_k$, then necessarily $d_k > d_j$.

Proof. Assume $d_j > d_k$. Because $b_{ij} + t_1 d_j \geq b_{ik} + t_1 d_k$ by definition of $V_i(t)$, then from Lemma 1 it follows that $b_{ij} + td_j > b_{ik} + td_k$, for all $t \geq t_1 \Rightarrow$ the contradiction with the assumption that $V_i(t_2) = b_{ik} + t_2 d_k$. The proof of the latter part is straightforward.

Lemma 2. Each function $V_i(t)$, $i \in I$ and $t \geq 0$, has the following properties:

a) functions $V_i(t)$ are piecewise linear and the number of linear pieces are at most p,
b) functions $V_i(t)$ are convex, and
c) $V_i(t) \geq b_{ij} + td_j$ for all j = 1,2, …, p.

Proof.

a) For each $t \geq 0$, there exists k, such that $V_i(t) = b_{ik} + td_k = \underset{j \in J}{max} (b_{ij} + td_j) \Rightarrow$ function $V_i(t)$ is piecewise linear. By Corollary 1.2, if for t_1 and t_2, $t_2 > t_1$, $V_i(t_1) = b_{ij} + t_1 d_j$ and $V_i(t_2) = b_{ik} + t_2 d_k$, $b_{ij} + t_1 d_j > b_{ik} + t_1 d_k$, then $d_k > d_j$. There are at most p elements for which this is true.

b) Let $t_1 \geq 0$ and $t_2 \geq 0$ be two arbitrary values of parameter t, and let $\lambda \in [0, 1]$. Then

$$V_i((1-\lambda)t_1 + \lambda t_2) = \underset{j\in J}{max}\,[b_{ij} + ((1-\lambda)t_1 + \lambda t_2)d_j] = \underset{j\in J}{max}\,[(1-\lambda)b_{ij} + \lambda b_{ij} + (1-\lambda)t_1 d_j + \lambda t_2 d_j]$$

$$= \underset{j\in J}{max}\,[(1-\lambda)(b_{ij} + t_1 d_j) + \lambda(b_{ij} + t_2 dj)] \leq \underset{j\in J}{max}\,(1-\lambda)(b_{ij} + t_1 d_j) + \underset{j\in J}{max}\,\lambda(b_{ij} + t_2 d_j)$$

$$\leq (1-\lambda)V_i(t_1) + \lambda V_i(t_2) \Rightarrow \text{the convexity of } V_i(t).$$

c) The result immediately follows from the definition of $V_i(t)$.

A key task in our algorithm is to recognize the indices $k \in I$ which cannot be the solution of (2.9b), when $t \in T$, where T is a specific interval. For this purpose we will prove the following Lemmas.

Lemma 3. Consider functions $V_m(t)$ and $V_i(t)$, m, $i \in I$, for which $V_m(t_0) = b_{mk} + t_0 d_k \geq V_i(t_0) = b_{ij} + t_0 d_j$, k, $j \in J$, and $d_k \geq d_j$, then $V_m(t) \geq V_i(t)$ for all $t \geq t_0$ for which $d_k \geq d_h$, $V_i(t) = b_{ih} + td_h$.

Proof. Let $t^* > t_0$ be arbitrarily chosen and assume $V_i(t^*) = b_{ih} + t^* d_h$, and $d_k \geq d_{\mathbf{h}}$, $h \in J$. By Lemma 2 c), $V_m(t^*) \geq b_{mk} + t^* d_k$ and by definition of $V_i(t)$, $V_i(t) = b_{ij} + t_0 d_j \geq b_{ih} + t_0 d_h$ for all $h \in J$. Thus $V_m(t^*) \geq b_{mk} + t^* d_k = b_{mk} + t_0 d_k + (t^*-t_0)d_k \geq b_{mk} + t_0 d_k + (t^*-t_0)d_h \geq b_{ij} + t_0 d_j + (t^*-t_0)d_h \geq b_{ih} + t_0 d_h + (t^*-t_0)d_h = b_{ih} + t^* d_h = V_i(t^*)$. The result concerning the strict inequality follows immediately.

Corollary 3.1 If $V_m(t_0) > V_i(t_0)$, then $V_m(t) > V_i(t)$, for all $t \geq t_0$ for which $d_k \geq d_h$.

Corollary 3.2 If $d_k = \underset{j \in J}{max}\, d_j$ when $V_m(t_0) \geq V_i(t_0)$, then $V_m(t) \geq V_i(t)$ ($V_m(t) > V_i(t)$) for all $t \geq t_0$.

Proof. Because $d_k \geq d_{\mathbf{h}}$ for all $h \in J$, the result follows immediately from Lemma 3.

Lemma 4. Consider functions $V_m(t)$ and $V_i(t)$, m, $i \in I$, and assume $V_i(t) = b_{ij} + td_j$, $j \in J$, when $t \in [t_1, t_2]$. If there exists k such that $b_{mk} + t_1 d_k \geq b_{ij} + t_1 d_j$ and $d_k \geq d_j$, then $V_m(t) \geq V_i(t)$ for all $t \in [t_1, t_2]$.

Proof. The result follows directly from Lemma 1 and Lemma 3.

Corollary 4.1 If $b_{mk} + t_1 d_k > b_{ij} + t_1 d_j$, then $V_m(t) > V_i(t)$ for all $t \in [t_1, t_2]$.

Corollary 4.2 If $d_k > d_j$, then $V_m(t) > V_i(t)$ for all $t \in (t_1, t_2]$.

3. Outline of the proposed algorithm

In the following, we describe our algorithm step by step and intersperse comments between the steps to clarify the reading of the algorithm. Without loss of generality we assume that all alternatives are nondominated, and there are no identical alternatives.

When a linear function $b_{ij} + td_j$ determining the maximum of function $V_i(t)$ with the certain value of t is not unique, we use one with a maximal d_j. To do this we define

$\beta(i,t) = \underset{j \in \Theta(i,t)}{arg\ max}\ d_j$, where $\Theta(i,t) = \{\ \theta \mid \theta = \underset{\theta \in J}{max}\ [b_{i\theta} + td_\theta]$, and let π: J $\rightarrow$ J be a function permutes the index set J such that $d_{\pi(1)} \le d_{\pi(2)} \le \ldots \le d_{\pi(p)}$. Accordingly, π^{-1} is standing for the inverse function.

For example, if we have $d_4 \le d_1 \le d_2 \le d_3$, *then* $d_{\pi(1)} \le d_{\pi(2)} \le d_{\pi(3)} \le d_{\pi(4)} \Rightarrow \pi(1) = 4$, $\pi(2) = 1$, $\pi(3) = 2$, *and* $\pi(4) = 3$. *Correspondingly,* $\pi^{-1}(4) = 1$, $\pi^{-1}(1) = 2$, $\pi^{-1}(2) = 3$, *and* $\pi^{-1}(3) = 4$.

Step 1: ***Initialization***

Assume the index of the current alternative is r, and let the reference direction be ρ. Set m := 1, M[m] := r, K := I - {r}, $T_{min} := 0$, $b_{ij} = (a_{rj} - a_{ij})/w_j$, and $d_j = \rho_j/w_j$, where i = 1,2, …, n and j = 1,2, …, p.

Because we assumed that all alternatives were nondominated, any alternative is a feasible choice.

Step 2: ***Determining Range of Inspection ΔT***

Determine the upper bound for the range $\Delta T = [T_{min}, T_{max}]$ of inspection as follows:

$$T_{max}: = \begin{cases} min\ \{t \mid \beta(r, t) \neq \beta(r, T_{min})\}, \text{ if } \{t \mid \beta(r, t) \neq \beta(r, T_{min})\} \neq \varnothing \\ \infty\ , \text{ otherwise} \end{cases}$$

and set $T_{cross} := T_{max}$. Set $j := \beta(r, T_{min})$, i := 0, and $i_{new} := 0$.

We determine the range ΔT such that the function $V_r(t)$ is linear, when $t \in \Delta T$.
We will determine parameter T_{cross} such that

$$T_{cross} = \begin{cases} \min\{t \mid t \in \Gamma\}, \text{if } \Gamma \neq \varnothing, \Gamma = \{t \mid V_i(t) \leq V_r(t), i \in J, i \neq r, \text{and } t \in [T_{min}, T_{max}]\} \\ T_{max}, \text{ otherwise} \end{cases}$$

Step 3: ***Finding the Next Potential Element of Vector M***

Repeat

i:= i + 1

If i > n then go to **Step 5.**

if (i∈ K) then

NeverCross := $\begin{cases} \text{True, if } b_{i\pi(p)} + T_{min}d_{\pi(p)} \geq b_{rj} + T_{min}d_j \\ \text{False, Otherwise} \end{cases}$

NoCross := NeverCross

See Corollary 3.2 for explanation

If NoCross

then

K := K - {i}

If K = {r} then **Stop.**

else

NoCross := $\begin{cases} \text{True, if } \exists\, k \in J \text{ for which } d_k \geq d_j \text{ and } b_{ik} + T_{min}d_k \geq b_{rj} + T_{min}d_j \\ \text{False, Otherwise} \end{cases}$

See Lemma 4 for explanation

end

until (i ∈ K) and not NoCross

Step 4: ***Finding the First Possible Crossing Point within Range $[T_{min}, T_{max}]$***

If (i∈ K) and not NoCross then

begin

$\xi := \pi^{-1}(j)$

$\tau_{cross} := T_{min}$

$j_{cross} := 0$

$h := 1$

While $(h < \xi)$ and $(d_j > d_{\pi(h)})$ do

begin

if $b_{i\pi(h)} + T_{min}d_{\pi(h)} > b_{rj} + T_{min}d_j$ then

begin

$$t = \frac{b_{i\pi(h)} - b_{rj}}{d_j - d_{\pi(h)}}$$

if $t \geq T_{cross}$ then go to **Step 3.**

By Lemma 2 c) $V_r(t) < V_i(t)$, when $t \in [T_{min}, T_{cross}]$

if $t > \tau_{cross}$

then

$\tau_{cross} := t$

$i_{cross} := i$

$j_{cross} := \pi(h)$

end

$h := h + 1$

end

$h := \xi$

Repeat $h := h + 1$ until $(h>p)$ or $(d_j \neq d_{\pi(h)})$

While $(h \leq p)$ do

begin

$$t = \frac{b_{rj} - b_{i\pi(h)}}{d_{\pi(h)} - d_j}$$

if $t < \tau_{cross}$ then go to **Step 3.**

$h := h + 1$

end

$T_{cross} := \tau_{cross}$

$i_{new} := i_{cross}$

$j_{new} := j_{cross}$

end

Go to **Step 3.**

Step 5: ***Updating Vector M***

If $T_{cross} < T_{max}$

then

set

$m := m + 1$

$M[m] := M[m] + i_{new}$

$T_{min} := T_{cross}$

$r := i_{new}$

$j := j_{new}$

Go to **Step 2**

else

$T_{min} := T_{max}$

Go to **Step 2**

4. Consideration of Computational Efficiency of the Algorithm

The algorithm is implemented in Turbo Pascal and the computational results are run with a 386 processor. It is well known that the algorithmic complexity is connected with the computational resources (so called elementary operations) for the worst case behavior of a given algorithm. In our case we found that the number of the elementary operations of the type {+, - , x , < , > , =} when determining the set M for a given reference direction is restricted by the following expression:

$$H = p^2 + 5n^2 + 5pn^2 - 3pn + 8p - 4. \qquad (4.1)$$

As we can see the highest degree in the above polynomial is pn^2. This expression, however, can not give enough information about the real behavior of our algorithm. It is supposed to work much better in practice. That is why we have also computed some experimental results.

The data sets are randomly generated in such a way that all alternatives are efficient. The results are given in Table 2

Table 2: Average Computing Times (in seconds) for Finding Set M for a Reference Direction

(The results for each cell are computed 5 times)

		p		
N	3	5	7	9
50	0.22	0.13	0.18	0.18
100	0.57	0.36	0.35	0.44
150	1.11	0.60	0.66	0.52
200	1.21	0.89	0.86	0.97
250	1.64	1.45	1.15	1.27
300	2.19	0.98	1.73	1.00
350	2.40	1.89	1.62	1.46
400	2.45	2.30	2.34	1.77

As we can see from Table 2, the polynomial describing the worst case behavior does not explain the resultsvery well. To search a proper model we started with the polynomial (4.1). The results are given in Table 3 (Model IT). The corresponding $R^2 = 0.923$. We also constructed a so-called unrestricted model (Model IIT) by adding variable n. After making some experiments, we ended up with the model called a restricted model (Model IIIT). Our purpose was to choose as simple a model as possible. For this model $R^2 = 0.920$.

To test the sufficiency of this model, we tested the hypothesis:

H_0: $\beta_1 = \beta_3 = \beta_4 = \beta_5 = 0$

H_1: $\beta_i \neq 0$ for at least one i = 1, 3, 4, and 5

Using the F-test

$$F(f_r\text{-}f_u, f_u) = \frac{\frac{R_u^2\text{-}R_r^2}{f_r - f_u}}{\frac{1\text{-}R_u^2}{f_u}} = F(30\text{-}25, 25) = \frac{\frac{0.936\text{-}0.920}{30 - 25}}{\frac{1\text{-}0.936}{25}} = 1.223,$$

where f_r and f_u are the degrees of freedom of the restricted and unrestricted model, correspondingly.

To choose the risk level α=0.05, we conclude H_0, because

$P\{F_{(5,25)} \geq 1.223\} = 0.328.$

In addition, we test the hypotheses H_0: $\beta i = 0$ against the alternative hypotheses H_1: $\beta_i \neq 0$, i = 2,5 with risk level $\alpha = 0.05$, we conclude H_1 in the both cases.

Thus, we think that the model $T = \beta_2 \frac{pn}{100} + \beta_6 n + \varepsilon$ explains quite well the computing times of our algorithm in practice. At first glance, it sounds strange that $\overline{\beta}_2 < 0$, but it is quite understandable. When the number of criteria increases, the number of elements in set M decreases as shown in Table 3, and the efficiency of the algorithm to recognize the alternatives not belonging to set M improves computing times.

To explain the results in Table 4, we started with a simple model (Model IS) by using p and n as a independent variables. We got $R^2 = 0.778$. The results in Table 5 indicate that the size of set M may be proportional to log n instead of n. That's why we used the model (Models IIS). Because for β_0, we conclude H_0: $\beta_0 = 0$ (H_1: $\beta_0 \neq 0$) $\alpha = 0.05$, we choose as a final model (Model IIS):

$$S_0 = \beta_1 p + \beta_2 \log n + \varepsilon.$$

For this model $R^2 = 0.817$ provides a quite good fitting.

Table 3: Searching the Model for Fitting Numerical Results in Table 2

	Model IT: Basic Model $T = \beta_0 + \beta_1 p + \beta_2 \frac{pn}{100} + \beta_3 \frac{pn^2}{1000} + \beta_4 \frac{n^2}{1000} + \beta_5 p^2 + \varepsilon$						
R_0^2= 0.923	$\bar{\beta}_0$	$\bar{\beta}_1$	$\bar{\beta}_2$	$\bar{\beta}_3$	$\bar{\beta}_4$	$\bar{\beta}_5$	$\bar{\beta}_6$
Coeff.	1.01	-0.27	0.065	-0.0021	0.017	0.015	
Dev.	0.35	0.12	0.024	0.0006	0.002	0.01	
t-Stat	2.94	-2.21	2.68	-3.49	7.96	1.55	
P-Value	0.007	0.036	0.012	0.002	0	0.134	
	Model IIT: Unrestricted Model $T = \beta_0 + \beta_1 p + \beta_2 \frac{pn}{100} + \beta_3 \frac{pn^2}{1000} + \beta_4 \frac{n^2}{1000} + \beta_5 p^2 + \beta_6 n + \varepsilon$						
R_u^2=0.936	$\bar{\beta}_0$	$\bar{\beta}_1$	$\bar{\beta}_2$	$\bar{\beta}_3$	$\bar{\beta}_4$	$\bar{\beta}_5$	$\bar{\beta}_6$
Coeff.	0.18	-0.14	-0.071	0.0008	-0.003	0.015	0.0093
Dev.	0.49	0.12	0.064	0.0014	0.009	0.009	0.0041
t-Stat	0.36	-1.16	-1.1	0.55	-0.34	1.67	2.25
P-Value	0.721	0.259	0.28	0.59	0.733	0.108	0.033
	Model IIIT: Restricted Model $T = \beta_2 \frac{pn}{100} + \beta_6 n + \varepsilon$						
R_r^2=0.920	$\bar{\beta}_0$	$\bar{\beta}_1$	$\bar{\beta}_2$	$\bar{\beta}_3$	$\bar{\beta}_4$	$\bar{\beta}_5$	$\bar{\beta}_6$
Coeff.			-0.035				0.0073
Dev.			0.006				0.0004
t-Stat			-5.35				17.69
P-Value			0				0

Table 4: The Average Size of Set M (The results for each cell are computed 5 times)

		p		
N	3	5	7	9
50	16.4	9.6	9.6	8.6
100	23.6	14.6	13.2	14
150	30	18.6	14.8	13
200	27.4	17.2	17.4	15.6
250	31.4	22	18	18.6
300	33	18.2	20.6	13.4
350	34	22	18.8	15.6
400	32.8	26.2	19	18.9

Table 5: Searching the Model for Fitting Numerical Results in Table 4

	Model IS: Basic Model $S_0 = \beta_0 + \beta_1 p + \beta_2 n + \varepsilon$		
	Intercept	p	n
R_r^2=0.778	$\bar{\beta}_0$	$\bar{\beta}_1$	$\bar{\beta}_2$
Coeff.	25.55	-2.19	0.032
Dev.	2.1	0.27	0.005
t-Stat	12.17	-8.09	6.01
P-Value	0	0	0
	Models IIS: log n -Model $S_1 = \beta_0 + \beta_1 p + \beta_2 \log n + \varepsilon$		
	Intercept	p	log n
R_r^2=0.818	$\bar{\beta}_0$	$\bar{\beta}_1$	$\bar{\beta}_2$
Coeff.	1.75	-2.19	13.6
Dev.	4.63	0.24	1.92
t-Stat	0.38	-8.93	7.1
P-Value	0.71	0	0
	Model IIIS: log n - Model $S_2 = \beta_1 p + \beta_2 \log n + \varepsilon$		
	Intercept	p	log n
R_r^2=0.817	$\bar{\beta}_0$	$\bar{\beta}_1$	$\bar{\beta}_2$
Coeff.		-2.16	14.28
Dev.		0.23	0.64
t-Stat		-9.42	22.34
P-Value		0	0

5. Conclusion

We have presented an algorithm to solve discrete multiple criteria problems. The algorithm is based on the reference direction approach. In order to minimize the solution time we use some heuristic tests eliminating those alternatives that have no chance to be a desired projection onto the nondominated set of given points. Our computational tests show that our algorithm is working in practice much better than the worst case behavior predicts.

REFERENCES

Hinloopen, E., Nijkamp, P., and Rietveld, P. (1983), "The regime method: A new multicriteria technique", in: P. Hansen (ed.), Essays and Surveys on Multiple Criteria Decision Making, Springer, 146-155.

Keeney, R., and Raiffa, H. (1976), Decisions with Multiple Objectives, Wiley, New York.

Korhonen, P., Wallenius, J., and Zionts, S. (1984), "Solving the discrete multiple criteria problem using convex cones", Management Science 30, 1336-1345.

Korhonen, P. (1986), "A hierarchical interactive method for ranking alternatives with multiple qualitative criteria", European Journal of Operational Research 24, 265-276.

Korhonen, P., and Laakso, J. (1986), "A visual interactive method for solving the multiple criteria problem", European Journal of Operational Research 24, 277-287.

Korhonen, P. (1988), "A visual reference direction approach to solving discrete multiple criteria problems", European Journal of Operational Research 34, 152-159.

Lotfi, V., T. J. Stewart, and S. Zionts (1992), "An Aspiration-Level Interactive Model for Multiple Criteria Decision Making." Computers and Operations Research 19, 671-681.

Marcotte, O., and Soland, R. (1986), "An interactive branch-and-bound algorithm for multiple criteria optimization", Management Science 32, 61-75.

Olson, D. (1996), Decision Aids for Selection Problems, Springer Series in Operations Research, New York.

Roy, B. (1973), "How outranking relation helps multiple criteria decision making", in: J. Cochrane and M. Zeleny (eds.), Multiple Criteria Decision Making, University of South Carolina Press, Columbia, SC, 179-201.

Saaty, T. (1980), The Analytic Hierarchy Process, McGraw-Hill, New York.

Steuer, R.E. (1986), Multiple Criteria Optimization: Theory, Computation, and Application, Wiley, New York.

Steuer, R. and Choo, E.-U. (1983), "An Interactive Weighted Tchebycheff Procedure for Multiple Objective Programming", Mathematical Programming 26, 326-344.

Wierzbicki, A. (1980), "The use of reference objectives in multiobjective optimization", in: G. Fandel and T. Gal (eds.), Multiple Criteria Decision Making, Theory and Application, Springer, New York.

Wierzbicki, A. (1986), "On the completeness and constructiveness of parametric characterizations to vector optimization problems", OR Spectrum 8, 73-87.

Zionts, S. and Wallenius, J. (1976). "An Interactive Programming Method for Solving the Multiple Criteria Problem", Management Science 22, 652-663.