Discussion Paper

A Bisection-Extreme Point Search Algorithm for Optimizing over the Efficient Set in the Linear Dependence Case

HAROLD P. BENSON
Department of Decision and Information Sciences, University of Florida, Gainesville, FL 32611-2017, U.S.A.

(Received: 12 June 1991; accepted: 28 February 1992)

Abstract. The algorithms and algorithmic ideas currently available for globally optimizing linear functions over the efficient sets of multiple objective linear programs either use nonstandard subroutines or cannot yet be implemented for lack of sufficient development. In this paper a Bisection-Extreme Point Search Algorithm is presented for globally solving a large class of such problems. The algorithm finds an exact, globally-optimal solution after a finite number of iterations. It can be implemented by using only well-known pivoting and optimization subroutines, and it is adaptable to large scale problems or to problems with many local optima.

Key words. Efficient set, multiple criteria decision making, bisection, extreme point search.

1. Introduction

Assume that $k \geqslant 2$ is an integer, and that $c_1^T, c_2^T, \ldots, c_k^T \in R^n$ are row vectors. Let C be the $k \times n$ matrix whose ith row is given by c_i^T, $i = 1, 2, \ldots, k$, and let X be a nonempty, compact polyhedron in R^n. We will assume without loss of generality that $X = \{x \in R^n | Ax \leqslant b, x \geqslant 0\}$, where A is an $m \times n$ matrix of real numbers and $b \in R^m$. Then the *multiple objective linear programming* problem (MOLP), written as

$$\text{VMAX: } Cx\,, \quad \text{subject to } x \in X\,,$$

can be viewed as the problem of finding all solutions that are efficient in the sense of the following definition.

DEFINITION 1.1. A point x^0 is said to be an *efficient* solution of problem (MOLP) when $x^0 \in X$, and whenever $Cx \geqslant Cx^0$ for some $x \in X$, then $Cx = Cx^0$.

An efficient solution is also often called a *nondominated* or *Pareto-optimal* solution. The functions $f_i(x) = \langle c_i, x \rangle$, $i = 1, 2, \ldots, k$, are called the *objective functions* or *criterion functions* for problem (MOLP). Let X_E denote the set of all efficient solutions of problem (MOLP).

Problem (MOLP) has been extensively studied and increasingly used as a decision aid for multiple criteria decision making (see, for instance, books and reviews by Evans [13], Rosenthal [25], Steuer [26], Yu [28], and Zeleny [29]). Many of the approaches for using problem (MOLP) to analyze situations with multiple objectives involve generating points in X_E. Two of these, in particular, are quite commonly used. They are the *vector maximization* approach and *interactive* approaches. In the vector maximization approach, either all of X_E or all of the extreme points of X contained in X_E is mathematically generated. The decision maker then examines the generated set and chooses a most preferred efficient solution. In the interactive approaches, the decision maker, with the aid of a computerized routine, iteratively generates selected points in X_E until he finds one that he most prefers (for details concerning these two approaches, see [13], [26], [28]).

The problem of main concern in this paper is the problem of optimizing a linear function over the efficient set X_E of problem (MOLP) in the case where the vector of coefficients of the linear function is linearly dependent upon the rows of C. This problem, denoted henceforth as problem (PD), may be written

$$\max\langle d, x\rangle\ , \quad \text{subject to } x \in X_E\ ,$$

where $d \in R^n$ and, for some $w \in R^k$, $d^T = w^T C$. Let θ denote the optimal objective function value for problem (PD).

While research on problem (PD) has been sparse, the motivation for studying it stems from recent interest in two problems closely related to it. To define these problems, let $g \in R^n$ and $i \in \{1, 2, \ldots, k\}$. The first problem, denoted problem (PI), may be written

$$\max\langle g, x\rangle\ , \quad \text{subject to } x \in X_E\ .$$

Since g can be linearly independent of the rows of C, problem (PI) subsumes problem (PD) as a special case. The second problem, denoted problem (PC), may be written

$$\min\langle c_i, x\rangle\ , \quad \text{subject to } x \in X_E\ .$$

Problem (PC) is a special case of problem (PD) since, with $d^T = (w^i)^T C$, where $(w^i)^T = (0, 0, \ldots, 0, -1, 0, \ldots, 0) \in R^k$ has -1 in its ith entry, problem (PD) yields problem (PC).

Interest in problems (PI) and (PC) arose in the past two decades and has recently been intensifying. At least part of this interest is in response to some of the difficulties in using problem (MOLP) as a decision aid. For instance, Philip [23] and Benson [7] argued that in certain multiple criteria decision making situations, models of the form of problem (PI) are simply more appropriate than models of the form (MOLP). As an instance, Benson [7] gave a production planning example. In this example, a firm seeks a maximum-profit production plan under the constraint that the production plan yield a vector of employment levels at the firm's $k = 10$ factories which is efficient. Benson [2] also explained how solving problem (PI) helps to avoid certain computational and practical burdens of using the generation method for problem (MOLP).

Other authors, including Weistroffer [27], Dessouky, Ghiassi, and Davis [10], Isermann and Steuer [18], and Reeves and Reid [24], have pointed out several important uses of problem (PC) in multiple criteria decision making. They have shown, for instance, how solving problem (PC) can improve the performance of several of the interactive algorithms for problem (MOLP), aid the decision maker in setting goals and evaluating decisions, and help the decision maker to decide how to rank the objective functions of problem (MOLP) according to their relative importance.

The motivation for solving problem (PD) comes from the same sources as the motivations for solving problems (PI) and (PC). This is because many of the multiple criteria decision making situations which can be represented as instances of problem (PI) are also instances of problem (PD), and any algorithm which can successfully solve problem (PD) can clearly solve problem (PC) as well.

Mathematically, problem (PD) can be classified as a *global optimization* problem (also called a *nonconvex programming* problem), since its feasible region X_E is, in general, a nonconvex set [15], [16], [21]. For the same reason, problems (PI) and (PC) are also global optimization problems. Such problems possess local optima, frequently large in number, which need not be globally optimal.

A few algorithms and algorithmic ideas have been suggested for finding globally optimal solutions for problems (PI) and (PC). Philip [23] and Bolintineanu [8] have described procedures using local search and cutting planes which could potentially solve problem (PI). Isermann and Steuer [18] independently suggested using the same approach as Philip, but for problem (PC). However, in all three cases, certain mathematical details necessary for implementing the procedures are not explained. More recently, Benson [3] [5] has proposed two implementable relaxation algorithms for solving problem (PI). The main computational burden in these algorithms involves solving either one linear program with additional bilinear constraints or one system of linear and bilinear equations for an improved solution at each iteration. Other studies of problems (PI) and (PC) or of generalizations of them have been either theoretical in nature or have searched for solutions which are not necessarily globally optimal (see, for instance, [7], [9], [10]).

When $d^T = w^T C$ for some $w \in R^k$ such that $w > 0$, it is well-known that problem (PD) can be solved by solving the linear program which maximizes $\langle d, x \rangle$ over X. But in applications where no such strictly positive vector w exists (e.g. problem (PC)), this approach fails.

No algorithm designed to globally solve problem (PD) has to date been developed. Study of problem (PD) has been sparse, and confined to theoretical results only [7].

In this paper, a Bisection-Extreme Point Search Algorithm is developed for finding a globally optimal solution for problem (PD). The algorithm has the following features:

(a) it is a finite procedure;
(b) it finds an exact, globally optimal solution;
(c) it is implementable;
(d) it uses only well-known pivoting and optimization subroutines;
(e) its main computational burden involves maximizations of convex functions over the compact polyhedron X;
(f) it frequently does not require carrying out the convex maximizations to optimality;
(g) it is adaptable to large-scale problems.

Crucial to the development of the algorithm is the fact that d is linearly dependent upon the rows of C.

The next section presents the theoretical prerequisites for the algorithm. Section 3 gives a statement of the algorithm and proves its convergence properties. In Section 4 guidelines and suggestions for implementing the algorithm are discussed. Some concluding remarks are given in Section 5.

2. Theoretical Background

Let X_{ex} denote the set of extreme points of the polyhedron X. The following result follows immediately from Theorem 4.5 in Benson [7].

THEOREM 2.1. *Problem (PD) has an optimal solution which belongs to* X_{ex}.

The Bisection-Extreme Point Search Algorithm to be developed will exploit the property given in Theorem 2.1 to find an optimal solution to problem (PD) belonging to $X_E \cap X_{\text{ex}}$ in a finite number of iterations.

An important property of problem (PD) that is crucial to the algorithm is given in the next result. Let $e \in R^k$ denote the vector whose entries are each unity.

THEOREM 2.2. *There exists a positive real number* $\hat{M}$ *such that for any* $M \geqslant \hat{M}$, $\theta = t^*$, *where* t^* *is the smallest value of the parameter* $t \in R$ *in the problem* (W_t) *given by*

$$\pi_t = \max \langle \lambda, Cx \rangle - \langle b, u \rangle - tv$$

subject to

$$\begin{aligned} Ax &\leqslant b \\ u^T A + v d^T - \lambda^T C &\geqslant 0 \\ \langle e, \lambda \rangle &= M \\ \lambda &\geqslant e \\ x, u, v &\geqslant 0 \end{aligned}$$

such that $\pi_t = 0$.

Proof. From Theorem 4.4 of Benson [7], there exists an $\hat{M} > 0$ such that with $M = \hat{M}$ in problem (W_t), $\theta = t^*$, where t^* is as described in the theorem. It is easy to see from the proof of Theorem 4.4 of [7], which is given in [6], that if the result holds for some particular positive value of M, then it also holds for all values of M exceeding that value. ■

Theorem 2.2 is not valid when d is not linearly dependent upon the rows of C. In such cases, the value t^* described in the theorem could strictly underestimate the optimal value θ of problem (PD), rather than equal it, as shown by the following example.

EXAMPLE 2.1. Let $X = \{x \in R^3 | Ax \leq b, x \geq 0\}$, where

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 2 \\ 4 \end{bmatrix},$$

and let C and d be given by

$$C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad d = \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix}.$$

Then it is easy to show that $x^{*T} = (0, 2, 4) \in X_E \cap X_{\text{ex}}$ is an optimal solution for problem (PD) with $\langle d, x^* \rangle = \theta = 2$. Algebraic reasoning, on the other hand, can be used to conclude that for every $M \geq 20$, for instance, if $t \geq -2$, then the optimal value π_t of problem (W_t) equals 0, while $\pi_t = +\infty$ for all $t < -2$. Thus, the application of Theorem 2.2 to this example yields the erroneous conclusion that $\theta = -2$. This false conclusion is due to the fact that d is not linearly dependent upon the rows of C. Therefore, Theorem 2.2 is not valid unless d is linearly dependent upon the rows of C.

For each value of $t \in R$, problem (W_t) given in the statement of Theorem 2.2 belongs to a class of global optimization problems called *bilinear programming problems* (see [16], [21], and references therein). By further studying this bilinear program, one can begin to describe the behavior of its optimal value π_t as t varies over R. The next result accomplishes this. Let t_m denote the optimal value of the linear program

$$\min \langle d, x \rangle, \quad \text{subject to } x \in X.$$

COROLLARY 2.1. *Let M be chosen as in Theorem 2.2, and consider the function $\pi: R \to R$ defined by $\pi(t) = \pi_t$ for each $t \in R$, where π_t is defined in Theorem 2.2. Then:*

(a) *π is nonnegative on R;*
(b) *$\pi(t) = +\infty$ for all $t < t_m$;*
(c) *$\pi(t)$ is finite for all $t \geq t_m$;*
(d) *$t^* \geq t_m$, and $\pi(t) = 0$ for all $t \geq t^*$;*
(e) *$\pi(t) > 0$ for all $t < t^*$;*

and

(f) π *is nonincreasing on R.*

Proof. Using the framework established in the proof given in [6] of Theorem 4.4 of [7], for any $t \in R$, let

$$X_t = \{x \in X | \langle d, x \rangle \leq t\},$$

and let

$$\Lambda = \{\lambda \in R^k \mid \lambda \geq e, \quad \langle e, \lambda \rangle = M\}.$$

Now define $\emptyset: \Lambda \to R$ and $\hat{\emptyset}: \Lambda x R \to R$ by

$$\emptyset(\lambda) = \max\langle \lambda, Cx \rangle, \text{ subject to } x \in X,$$

and

$$\hat{\emptyset}(\lambda, t) = \max\langle \lambda, Cx \rangle, \text{ subject to } x \in X_t$$

for each $\lambda \in \Lambda$ and $t \in R$, respectively. Then define a function $\bar{\pi}: R \to R$ by

$$\bar{\pi}(t) = \sup_{\lambda \in \Lambda} [\emptyset(\lambda) - \hat{\emptyset}(\lambda, t)]$$

for each $t \in R$.

Notice that for any $t \in R$, $X_t \neq \emptyset$ if and only if $t \geq t_m$. Also, it was shown in [6] that for each $t \in R$, $\pi(t)$ is finite if and only if $\bar{\pi}(t)$ is finite, and $\pi(t) = \bar{\pi}(t)$ whenever these numbers are finite.

Choose any $t \in R$. Then $X_t \subseteq X$, so that for each $\lambda \in \Lambda$, $\emptyset(\lambda) \geq \hat{\emptyset}(\lambda, t)$. From the definition of $\bar{\pi}$, this implies that $\bar{\pi}(t) \geq 0$. Hence, if $\pi(t)$ is finite, $\pi(t) \geq 0$. From Theorem 2.2, $\pi(t^*) = 0$. This implies that problem (W_{t^*}), and hence problem (W_t), has a feasible solution. Therefore, if $\pi(t)$ is not finite, $\pi(t) = +\infty$, and part (a) is established.

Suppose now that $t < t_m$. Then $X_t = \emptyset$, so that by definition, $\hat{\emptyset}(\lambda, t) = -\infty$ for each $\lambda \in \Lambda$. Since X is nonempty and compact, $-\infty < \emptyset(\lambda) < \infty$ for each $\lambda \in \Lambda$. The latter two statements imply that $\bar{\pi}(t) = +\infty$. Hence $\pi(t)$ is not finite. By part (a), $\pi(t) = +\infty$ must hold, thus establishing (b).

Now suppose that $t \geq t_m$. Then X and X_t are nonempty, compact polyhedral sets. Therefore, from linear programming theory (see, for instance, Murty [20]), for each $\lambda \in R^k$,

$$\emptyset(\lambda) = \max\{\langle \lambda, Cx^1 \rangle, \langle \lambda, Cx^2 \rangle, \ldots, \langle \lambda, Cx^q \rangle\}$$

and

$$\hat{\emptyset}(\lambda, t) = \max\{\langle \lambda, Cx_t^1 \rangle, \langle \lambda, Cx_t^2 \rangle, \ldots, \langle \lambda, Cx_t^r \rangle\}$$

where x^i, $i = 1, 2, \ldots, q$, and x_t^j, $j = 1, 2, \ldots r$, are the extreme points of X and X_t, respectively. From Theorem 8.7 in [20], this implies that the functions $\emptyset(\cdot)$ and $\hat{\emptyset}(\cdot, t)$ are continuous on R^k. Therefore, since Λ is a nonempty compact set, the supremum in the definition of $\bar{\pi}(t)$ is achieved and $\bar{\pi}(t)$ is a finite number.

Hence $\pi(t)$ is a finite number and (c) is established.

Suppose now that $t \geq t^*$. By Theorem 2.2, $\pi(t^*) = 0$. From parts (b) and (c), this implies that $t^* \geq t_m$. Therefore $t \geq t_m$, so that $X_t \neq \emptyset$ and, by part (c), $\pi(t)$ is finite. Since $\pi(t^*)$ and $\pi(t)$ are finite numbers, $\pi(t^*) = \bar{\pi}(t^*) = 0$, and $\pi(t) = \bar{\pi}(t)$.

Since $t \geq t^*$, $X_{t^*} \subseteq X_t$. Therefore, for each $\lambda \in \Lambda$, by definition of $\hat{\emptyset}$, $\hat{\emptyset}(\lambda, t^*) \leq \hat{\emptyset}(\lambda, t)$. This implies that for each $\lambda \in \Lambda$, $\emptyset(\lambda) - \hat{\emptyset}(\lambda, t) \leq \emptyset(\lambda) - \hat{\emptyset}(\lambda, t^*)$. From the definition of $\bar{\pi}$, this implies that $\bar{\pi}(t) \leq \bar{\pi}(t^*) = 0$. Since $\pi(t) = \bar{\pi}(t)$, it follows that $\pi(t) \leq 0$. By part (a) $\pi(t) \geq 0$. Therefore, $\pi(t) = 0$ and part (d) is established.

Assume now that $t < t^*$. By Theorem 2.2, $\pi(t) \neq 0$. From part (a) this implies that $\pi(t) > 0$, so that part (e) follows.

Part (f) will follow from the previous parts once it is shown that π is nonincreasing on $[t_m, t^*]$. Towards this end, choose $t_1, t_2 \in R$ such that $t_m \leq t_1 < t_2 \leq t^*$. From part (c), $\pi(t_1)$ and $\pi(t_2)$ are finite numbers. Therefore, $\pi(t_1) = \bar{\pi}(t_1)$ and $\pi(t_2) = \bar{\pi}(t_2)$. Since $t_1 < t_2$, $X_{t_1} \subseteq X_{t_2}$. Therefore, for each $\lambda \in \Lambda$, $\hat{\emptyset}(\lambda, t_1) \leq \hat{\emptyset}(\lambda, t_2)$. This implies that for each $\lambda \in \Lambda$, $\emptyset(\lambda) - \hat{\emptyset}(\lambda, t_1) \geq \emptyset(\lambda) - \hat{\emptyset}(\lambda, t_2)$. From the definition of $\bar{\pi}$, it follows that $\bar{\pi}(t_1) \geq \bar{\pi}(t_2)$. Since $\bar{\pi}(t_i) = \pi(t_i)$, $i = 1, 2$, part (f) is established. ∎

In the Bisection-Extreme Point Search Algorithm to be presented for problem (PD), the question of whether $\pi_t = 0$ or $\pi_t > 0$ must be answered for various values of t satisfying $t \geq t_m$. The following result gives another definition of π_t when $t \geq t_m$.

THEOREM 2.3. *Let $t \in R$ satisfy $t \geq t_m$, and assume that $M \geq \max\{k, \hat{M}\}$, where $\hat{M}$ is chosen as in Theorem 2.2. Then the value of π_t in problem (W_t) equals the optimal value of the problem (V_t) given by*

$$\max h_t(x), \quad \text{subject to } Ax \leq b, \quad x \geq 0,$$

where $h_t\colon R^n \to R$ is the continuous piecewise linear convex function defined by

$$h_t(x) = \max \langle \lambda, Cx \rangle - \langle b, u \rangle - tv$$

subject to

$$\begin{aligned} u^T A + v d^T - \lambda^T C &\geq 0 \\ \langle e, \lambda \rangle &= M \\ \lambda &\geq e \\ u, v &\geq 0 \end{aligned}$$

for each $x \in R^n$.

Proof. Let $x \in R^n$. We first show that since $t \geq t_m$ and $M \geq k$, $h_t(x)$ is a finite number. Since t and x are fixed, the problem defining $h_t(x)$ is a linear programming problem (LP). By duality theory of linear programming [20], $h_t(x)$ equals the optimal value of the dual linear program to problem (LP), i.e.,

$$h_t(x) = \min M\gamma - \langle e, s \rangle$$

$$\text{subject to} \quad Cy + e\gamma - s \geq Cx \tag{1}$$

$$Ay \leq b \tag{2}$$

$$d^T y \leq t \tag{3}$$

$$y, s \geq 0 . \tag{4}$$

Since $t \geq t_m$, $X_t \neq \emptyset$. By choosing any fixed $y \in X_t$, $s = 0$, and $\gamma \in R$ sufficiently large, the constraints (1)–(4) are satisfied. Therefore $h_t(x) < +\infty$. Now let y, γ, and s be any points which together satisfy (1)–(4). Then from (1), for each $i = 1, 2, \ldots, k$,

$$\gamma - s_i \geq (Cx - Cy)_i . \tag{5}$$

From (2)–(4), $y \in X_t$. Since X_t is compact, this together with (5) implies that there exist real numbers α_i, $i = 1, 2, \ldots, k$ such that $(\gamma - s_i) \geq \alpha_i$, $i = 1, 2, \ldots, k$. Summing these k inequalities yields

$$k\gamma - \langle e, s \rangle \geq \alpha ,$$

where $\alpha = \Sigma_{i=1}^{k} \alpha_i$. By rearranging the last inequality, it follows that

$$\gamma \geq (\alpha + \langle e, s \rangle)/k .$$

Since $M > 0$, this implies that

$$M\gamma \geq (M/k)(\alpha + \langle e, s \rangle) . \tag{6}$$

From (6), it follows that

$$\begin{aligned} M\gamma - \langle e, s \rangle &\geq (M/k)(\alpha + \langle e, s \rangle) - \langle e, s \rangle \\ &= (M/k)\alpha + [(M/k) - 1]\langle e, s \rangle \\ &\geq (M/k)\alpha \\ &> -\infty , \end{aligned}$$

where the second-to-last inequality follows from $M \geq k > 0$ and $s \geq 0$. Since y, γ, and s were arbitrary points satisfying (1)–(4), we conclude that $h_t(x) > -\infty$. Together with $h_t(x) < +\infty$, this implies that $h_t(x)$ is a finite number.

From linear programming theory [20], since $h_t(x)$ is a finite number defined as the optimal value of linear program (LP),

$$h_t(x) = \max\{\langle (\lambda^i)^T C, x \rangle - \langle b, u^i \rangle - tv^i \mid i = 1, 2, \ldots, p\} ,$$

where (λ^i, u^i, v^i), $i = 1, 2, \ldots, p$ are the extreme points of the polyhedron defining the feasible region of linear program (LP). This implies that $h_t \colon R^n \to R$ is a piecewise linear convex function [20, Theorem 8.7]. Therefore, h_t is continuous on R^n.

Since X is nonempty and compact and h_t is continuous on R^n, the optimal value β_t of problem (V_t) is finite. By the definitions of problem (V_t), of h_t, of X, and of π_t, since $M \geq \hat{M}$,

$$\begin{aligned}\beta_t &= \max h_t(x)\,, \quad \text{subject to } x \in X \\ &= \max_{x \in X} \{\max \langle \lambda, Cx \rangle - \langle b, u \rangle - tv\} \\ &\qquad\qquad \text{subject to } u^tA + vd^T - \lambda^T C \geq 0 \\ &\qquad\qquad\qquad\qquad \langle e, \lambda \rangle = M \\ &\qquad\qquad\qquad\qquad \lambda \geq e \\ &\qquad\qquad\qquad\qquad u, v \geq 0 \\ &= \pi_t\,,\end{aligned}$$

and the proof is complete. ■

From Corollary 2.1 and Theorem 2.3, if $t \geq t_m$, π_t is nonnegative and finite, and it can be calculated by maximizing the continuous convex function h_t: $R^n \rightarrow R$ over the nonempty compact polyhedron X. Although such a calculation involves solving a global optimization problem, it can be accomplished by any of a number of algorithms now available for convex maximization [16], [21], [22]. Crucial to the applicability of these algorithms is the fact that h_t is continuous over R^n.

To motivate the final background result, it is necessary to consider one of the procedures used in the Bisection-Extreme Point Search Algorithm for problem (PD). The algorithm needs to answer the question of whether $\pi_t = 0$ or $\pi_t > 0$ for various values of $t \geq t_m$. From Theorem 2.2 and Corollary 2.1, if $\pi_t = 0$, then $t \geq \theta$, while if $\pi_t > 0$, $t < \theta$. In the latter case, by Theorem 2.1 and the definition of θ, there exists at least one point $\bar{x} \in X_E \cap X_{ex}$ such that $\langle d, \bar{x} \rangle > t$. The algorithm will use the next result to find such a point $\bar{x}$ when $\pi_t > 0$.

THEOREM 2.4. *Let $t \in R$ satisfy $t \geq t_m$, and assume that $M \geq \max\{k, \hat{M}\rangle$, where $\hat{M}$ is chosen as in Theorem 2.2. Then:*

(a) *Problem (V_t) defined in Theorem 2.3 has an optimal solution which belongs to X_{ex};*

(b) *If $h_t(\hat{x}) > 0$ for some feasible solution $\hat{x}$ for problem (V_t), then*

  (i) $\pi_t > 0$,

  (ii) $t < \theta$,

  and

  (iii) *if $(\hat{\lambda}, \hat{u}, \hat{v})$ is any optimal solution to the linear program given in Theorem 2.3 whose optimal value defines $h_t(\hat{x})$, then for any optimal extreme point solution $\bar{x}$ to the linear program $(P_{\hat{\lambda}})$ given by*

$$\max \langle \hat{\lambda}^T C, x \rangle\,, \quad \textit{subject to } x \in X\,,$$

$\bar{x} \in X_E \cap X_{ex}$ *and* $\langle d, \bar{x} \rangle > t$.

Proof. From Theorem 2.3, for t and M as chosen in Theorem 2.4, problem (V_t) involves maximizing the continuous convex function h_t over the nonempty compact polyhedron X. From Martos [19], this implies that problem (V_t) has an optimal solution in X_{ex}, and part (a) is established. Part (b)(i) follows directly from Theorem 2.3 and the definition of problem (V_t). Part (b)(ii) then follows from part (b)(i), Theorem 2.2, and Corollary 2.1.

To establish part (b)(iii), let $(\hat{\lambda}, \hat{u}, \hat{v})$ be any optimal solution to the linear program given in Theorem 2.3 whose optimal value defines $h_t(\hat{x})$. Since $h_t(\hat{x}) > 0$, $\langle \hat{\lambda}, C\hat{x} \rangle - \langle b, \hat{u} \rangle - t\hat{v} \rangle > 0$, and the optimal value γ_1 of the linear program

$$\begin{aligned} &\max \langle \hat{\lambda}, C\hat{x} \rangle - \langle b, u \rangle - tv \\ &\text{subject to } u^T A + v d^T \geq \hat{\lambda}^T C \\ &\qquad\qquad u, v \geq 0 \end{aligned}$$

is positive. Notice that $\gamma_1 = \langle \hat{\lambda}, C\hat{x} \rangle - \gamma_2$, where γ_2 is the optimal value of the linear program

$$\begin{aligned} &\min \langle b, u \rangle + tv \\ &\text{subject to } u^T A + v d^T \geq \hat{\lambda}^T C \\ &\qquad\qquad u, v \geq 0 . \end{aligned}$$

By duality theory of linear programming [20], γ_2 equals the optimal value of the linear program

$$\begin{aligned} &\max \langle \hat{\lambda}, Cy \rangle \\ &\text{subject to } Ay \leq b \\ &\qquad \langle d, y \rangle \leq t \\ &\qquad\qquad y \geq 0 . \end{aligned}$$

Since $\gamma_1 > 0$ and the feasible region of the latter problem is X_t, $\langle \hat{\lambda}, C\hat{x} \rangle$ exceeds the optimal value γ_2 of the linear program (P_t) given by

$$\max \langle \hat{\lambda}, Cy \rangle , \quad \text{subject to } y \in X_t .$$

Now assume that $\bar{x}$ is any optimal extreme point solution to the linear program $(P_{\hat{\lambda}})$ Then $\bar{x} \in X_{ex}$ and, since $\hat{\lambda} \geq e > 0$, $\bar{x} \in X_E$ [23]. In addition, since $\hat{x}$ is a feasible solution for problem $(P_{\hat{\lambda}})$, the optimal value γ_3 of problem $(P_{\hat{\lambda}})$ satisfies $\gamma_3 \geq \langle \hat{\lambda}, C\hat{x} \rangle$. Since $\langle \hat{\lambda}, C\hat{x} \rangle > \gamma_2$, this implies that $\gamma_3 > \gamma_2$. It follows from this inequality that $\langle d, \bar{x} \rangle > t$ must hold, for if it did not, then $\bar{x}$ would be a feasible solution to problem (P_t), so that $\gamma_2 \geq \langle \hat{\lambda}, C\bar{x} \rangle = \gamma_3$ would hold. ■

Theorem 2.4 is instrumental in guaranteeing that the algorithm to be presented finds an exact, globally optimal solution for problem (PD) after a finite number of iterations.

3. The Bisection-Extreme Point Search Algorithm

Assume henceforth in this paper that M is chosen as in Theorems 2.3 and 2.4. The goal of the Bisection-Extreme Point Search Algorithm is to find an optimal extreme point solution for problem (PD) in a finite number of iterations. A bisection search is used in the algorithm to find progressively smaller subintervals of R containing θ. For each $t \in (L, U)$ chosen during the bisection search, where $[L, U]$ is the current subinterval of R known to contain θ, the convex maximization problem (V_t) defined in Theorem 2.3 is used to determine whether $\pi_t = 0$ or $\pi_t > 0$. When $\pi_t = 0$, U is decreased to t. When $\pi_t > 0$, an extreme point search procedure is invoked to update L. This procedure finds a new point x^{INC}, called an *incumbent solution*, which belongs to $X_E \cap X_{\mathrm{ex}}$ and satisfies $\langle d, x^{\mathrm{INC}} \rangle > t$. The value of L is then increased to $\langle d, x^{\mathrm{INC}} \rangle$. A test is performed, at times partially determined by the user of the algorithm, to determine whether $\pi_L = 0$ or $\pi_L > 0$. When $\pi_L = 0$, x^{INC} is an (exact) optimal solution for problem (PD), and the algorithm terminates.

One of the procedures repeatedly used by the algorithm involves finding a local optimum for problem (PD) by a standard search procedure similar to those used previously, for instance, in [8] and in [23]. To facilitate the presentation of the algorithm, this procedure will be described first. Towards this end, consider the following definition.

DEFINITION 3.1. A point $x^0 \in X_E \cap X_{\mathrm{ex}}$ is said to be a *locally optimal solution* for problem (PD) when

$$\langle d, x^0 \rangle \geq \langle d, x^i \rangle \quad \text{for all } i \in Q_E ,$$

where x^i, $i = 1, 2, \ldots, q$, are the elements of $X_E \cap X_{\mathrm{ex}}$ adjacent to x^0 in the polyhedron X, and $Q_E \subseteq \{1, 2 \ldots, q\}$ indexes those elements x^i for which the edge connecting x^i and x^0 lies in X_E.

We can now state the Local Optimum Search Procedure. In this procedure, it is assumed that $y^0 \in X_E \cap X_{\mathrm{ex}}$.

Local Optimum Search Procedure

Step 1. Set $w = 0$, $i = 0$, and $E_0^w = \emptyset$.

Step 2. If possible, find an edge E_{i+1}^w of X emanating from y^w such that $E_{i+1}^w \subseteq X_E$ and $E_{i+1}^w \neq E_j^w$, $j = 0, 1, \ldots, i$. If no such edge exists, then STOP: y^w is a locally optimal solution for problem (PD). Otherwise, find the extreme point $y_{i+1}^w \neq y^w$ of X on the edge E_{i+1}^w and continue.

Step 3. If $\langle d, y_{i+1}^w \rangle > \langle d, y^w \rangle$, set $y^{w+1} = y_{i+1}^w$, $w = w + 1$, $i = 0$, and $E_0^w = \emptyset$ and go to Step 2. Otherwise, set $i = i + 1$ and go to Step 2.

The algorithm may now be stated as follows.

Bisection-Extreme Point Search Algorithm for Problem (PD)

Initialization Step. Find any point $x^0 \in X_E \cap X_{ex}$. With x^0 as a starting point, use the Local Optimum Search Procedure to find a locally optimal solution $\hat{x}^0 \in X_E \cap X_{ex}$ for problem (PD). Set $\underline{M} = \langle d, \hat{x}^0 \rangle$ and calculate

$$\bar{M} = \max \langle d, x \rangle \ , \quad \text{subject to } x \in X \, .$$

Choose a parameter $\rho \in (0, 1]$, set FLAG = OFF, $L = \underline{M}$, $U = \bar{M}$, $x^{\text{INC}} = \hat{x}^0$, and $k = 0$, and go to Iteration k.

Iteration k, $k \geq 0$.

Step k.1.(i) If FLAG = ON, go to Step k.2. Otherwise, go to Step k.1(ii).

(ii) If $(U - L) > \rho(\bar{M} - \underline{M})$, go to Step k.3. Otherwise, set FLAG = ON and go to Step k.2.

Step k.2. Use an appropriate convex maximization algorithm for problem (V_t) with $t = L$ to determine whether $\pi_L = 0$ or $\pi_L > 0$. If $\pi_L = 0$, STOP: x^{INC} is an optimal solution for problem (PD), and $\theta = L$. If $\pi_L > 0$, continue.

Step k.3. Set $t = (L + U)/2$. Use an appropriate convex maximization algorithm for problem (V_t) to determine whether $\pi_t = 0$ or $\pi_t > 0$. If $\pi_t = 0$, go to Step k.4. If $\pi_t > 0$, find any feasible solution $\hat{x}$ for problem (V_t) such that $h_t(\hat{x}) > 0$ and go to Step k.5.

Step k.4. Set $U = t$ and $k = k + 1$. If FLAG = OFF, go to Step k.1.(ii). If FLAG = ON, go to Step k.3.

Step k.5. Find any optimal solution $(\hat{\lambda}, \hat{u}, \hat{v})$ to the linear program given in Theorem 2.3 whose optimal value defines $h_t(\hat{x})$.

Step k.6. Find any extreme point optimal solution x^k to the linear program $(P_{\hat{\lambda}})$ given by

$$\max \langle \hat{\lambda}^T C, x \rangle \ , \quad \text{subject to } x \in X \, .$$

With x^k as a starting point, use the Local Optimum Search Procedure to find a locally optimal solution $\hat{x}^k \in X_E \cap X_{ex}$ for problem (PD).

Step k.7. Set $x^{\text{INC}} = \hat{x}^k$, $L = \langle d, x^{\text{INC}} \rangle$, $k = k + 1$, and go to Iteration Step k.

The parameter ρ chosen by the user helps establish the iteration at which the termination test in step k.2 first is invoked. When FLAG = OFF, no termination tests are performed. As soon as U and L satisfy $(U - L) \leq \rho(\bar{M} - \underline{M})$, FLAG is set equal to ON. The termination Step k.2 is then immediately invoked for the first time. Subsequently, FLAG keeps the value ON, and each time that the value of L is changed, the termination Step k.2 is executed.

Steps k.3 – k.7 execute the bisection search procedure. Notice that when $\pi_t = 0$ in Step k.3, U is simply decreased to t in Step k.4, and the algorithm moves to the next iteration. But when $\pi_t > 0$ in Step k.3, the extreme point search procedure is used to update L. This procedure begins in Step k.3 by finding a feasible solution $\hat{x}$ for problem (V_t) such that $h_t(\hat{x}) > 0$. Next, Steps k.5 and k.6 use the results in Theorem 2.4 to find an efficient extreme point x^k which satisfies $\langle d, x^k \rangle > t$. The Local Optimum Search Procedure is then invoked to find an efficient extreme point $\hat{x}^k$ which satisfies $\langle d, \hat{x}^k \rangle \geq \langle d, x^k \rangle > t$. In Step k.7, L is increased to $\langle d, \hat{x}^k \rangle$, and $\hat{x}^k$ becomes the new incumbent solution.

The convergence properties of the Bisection-Extreme Point Search Algorithm are given in the next result.

THEOREM 3.1. *The Bisection-Extreme Point Search Algorithm finds an exact, globally optimal solution for problem (PD) after a finite number of iterations.*

Proof. From the definitions of $\underline{M}$ and $\bar{M}$, $\theta \in [\underline{M}, \bar{M}]$, and each value of $t = (L + U)/2$ in Step k.3 satisfies $t \geq t_m$. Therefore, by Theorem 2.2 and Corollary 2.1, for each such t, either $\pi_t = 0$ and $t \geq \theta$, or $\pi_t > 0$ and $t < \theta$. In the former case, Step k.4 is invoked and the new value of U is set equal to t. Thus, Step k.4 is valid and reduces the interval $[L, U]$ by one-half. In the latter case, Steps k.3 and k.5–k.6 are used to find a point $\hat{x}^k \in X_E \cap X_{ex}$, and the new value of L is set equal to $\langle d, \hat{x}^k \rangle$. Since $\hat{x}^k$ is a feasible solution for problem (PD), this new value is a valid lower bound for θ. Furthermore, from Theorem 2.4 and the Local Optimum Search Procedure, $\langle d, \hat{x}^k \rangle > t$, so that the interval $[L, U]$ is reduced by *at least* one half.

Since $0 < \rho \leq 1$ and each iteration of the algorithm reduces the interval $[L, U]$ containing θ by one-half or more, after some finite number of iterations, the inequality $(U - L) \leq \rho(\bar{M} - \underline{M})$ is satisfied. Therefore, the value of FLAG in Step k.1(ii) is set equal to ON, and the termination test in Step k.2 is performed for the first time. Subsequently, it is performed each time that the value of L is increased.

If $\pi_L = 0$ is eventually detected in the termination test in Step k.2 during some iteration k, then, from Theorem 2.2 and Corollary 2.1, $L \geq \theta$. Since $L \leq \theta$ always holds, this implies that $\theta = L$. This, together with $x^{INC} \in X_E$ and $\langle d, x^{INC} \rangle = L$, implies that x^{INC} is an optimal solution for problem (PD). Therefore, the algorithm, if it terminates, finds an exact, globally optimal solution for problem (PD).

We now show that the algorithm generates an L value satisfying $\pi_L = 0$ after some finite number of iterations. To show this suppose, to the contrary, that this is not the case. Then for every value of L generated by the algorithm, $\pi_L > 0$. The number of distinct values of L generated by the algorithm is either (a) finite or (b) infinite.

Case (a): The number of distinct values of L generated by the algorithm is finite. Let $\bar{L}$ be the largest of these values. Then from Theorem 2.2 and Corollary 2.1, since $\pi_{\bar{L}} > 0$, $\bar{L} < \theta$. Because no larger values than $\bar{L}$ for L are generated by the algorithm, for each t generated via Step k.3 in iterations subsequent to the one where $\bar{L}$ is generated, $\pi_t = 0$. This implies that in each of these subsequent iterations, by using Steps k.3 and k.4, the algorithm reduces the interval $[\bar{L}, \bar{U}]$ that existed when $\bar{L}$ was generated by one-half by decreasing $U = \bar{U}$. Therefore, since $\bar{L} < \theta$, the upper bound U for the interval $[\bar{L}, U]$ must eventually satisfy

$U < \theta$. But this contradicts the validity of the upper bounds. Therefore this case cannot hold.

Case (b): The number of distinct values of L generated by the algorithm is infinite. Let $L_1, L_2, L_3, \ldots$ represent these L values, where $L_i < L_{i+1}$, $i = 1, 2, \ldots$ For each L_i generated, an efficient point y^i in X_{ex} is generated in Step k.6 which satisfies $\langle d, y^i \rangle = L_i$. Therefore, $X_E \cap X_{ex}$ contains an infinite number of distinct points. But from linear programming theory [20], this is impossible. Therefore, this case cannot hold either.

Since neither Case (a) nor Case (b) can hold, it follows that the algorithm generates an L value which satisfies $\pi_L = 0$ after some finite number of iterations. Coupled with the facts that after some finite number of iterations, the termination test in Step. k.2 is performed, and that it is subsequently performed whenever a new value of L is found, this implies that the algorithm terminates after a finite number of iterations, and the proof is complete. ■

4. Implementation Issues

The Bisection-Extreme Point Search Algorithm can be implemented using only well-known pivoting and optimization subroutines. In this section, some suggestions and guidelines are briefly given for accomplishing this implementation. An examination of the algorithm reveals that it calls for the following:

(a) solving linear programs;
(b) finding an initial point $x^0 \in X_E \cap X_{ex}$;
(c) accomplishing the Local Optimum Search Procedure;
(d) deciding for various values of $t \geq t_m$ whether $\pi_t = 0$ or $\pi_t > 0$;
(e) finding, when $\pi_t > 0$, a feasible solution $\hat{x}$ for problem (V_t) such that $h_t(\hat{x}) > 0$.

Item (a) can be readily accomplished by using either the well-known simplex method or Karmarkar's method [1] [20]. Several simple methods involving the simplex method and/or simplex-type pivoting operations can be used to implement item (b) (see [4], [11], [26], and references therein).

To implement item (c), for any $y \in X_E \cap X_{ex}$, a means of identifying each edge of X emanating from y which lies in X_E must be available. In addition, for each such edge, the extreme point $\bar{y} \neq y$ of X on the edge must be identified. Various procedures have been developed which can accomplish these tasks. Most of them were developed in conjunction with algorithms for generating all points in $X_E \cap X_{ex}$ (see [12], [14], [17], [26] and references therein). These procedures use linear programming and simplex-type pivoting operations. Although any one of them will suffice in implementing item (c), computational experience with the Evans–Steuer method [14] [26] is favorable and seems particularly extensive.

To implement items (d) and (e), a means is needed for finding the optimal value π_t of the convex maximization problem (V_t) for various values of $t \geq t_m$. In addition, when $\pi_t > 0$, a feasible solution $\hat{x}$ for problem (V_t) with a positive objective function value $h_t(\hat{x})$ must be found. Since problem (V_t) is a global optimization problem, it is anticipated that the main computational burden in executing the algorithm will involve implementing items (d) and (e). Some care should therefore be exercised in choosing the most efficient means available for implementing these items. In addition, the number of different problems (V_t) that the Bisection-Extreme Point Search Algorithm must consider should be kept to a minimum. Following are some general suggestions and guidelines for accomplishing these tasks.

Assume that $t \geq t_m$. From Theorem 2.3, problem (V_t) involves the maximization of a continuous convex function $h_t: R^n \to R$ over a nonempty, compact polyhedron. Since h_t is continuous on R^n, a wide variety of general-purpose algorithms for solving this global optimization problem is available (see [15], [16], [21], [22] and references therein). Many of these algorithms are finite. While the particular algorithm chosen will depend upon many factors, two factors of particular computational importance in making this choice should be considered.

First, in choosing an algorithm for solving problem (V_t), one should consider how the algorithm detects that $\pi_t > 0$. For instance, many algorithms which use relaxation or extreme point ranking cannot detect that $\pi_t > 0$ until the algorithm has run to completion and has found an optimal solution. Other algorithms, such as those using branch and bound, can detect that $\pi_t > 0$ prior to finding an optimal solution by first finding a feasible solution $\hat{x}$ such that $h_t(\hat{x}) > 0$. Algorithms of the latter type may be preferred, since they often will accomplish items (d) and (e) when $\pi_t > 0$ without having to solve problem (V_t) to optimality.

A second consideration in choosing an algorithm for solving problem (V_t) concerns the potential for using information generated from solving a previous problem to help solve a later problem. The problems (V_t) differ only in the parameter $t \in R$. Therefore, for some algorithms, the potential may exist to reduce solution times for the problems (V_t) by using information generated from problems already solved. Such algorithms may be preferable to those for which no such potential exists.

The parameter $\rho \in (0, 1]$ chosen by the user of the algorithm is intended to keep to a minimum the number of different convex maximization problems (V_t) that need to be considered. Each time the termination test in Step k.2 is performed, a new problem (V_L) must be considered. For problems which have many local optima which are not global, it may be wise to choose a rather small value of ρ (e.g. $\rho < 1/4$) in order to avoid needless termination tests in earlier steps of the algorithm. For problems with fewer local optima, larger values of ρ seem preferable. Of course, the user has no guaranteed means of knowing in advance the number of local optima that exist in problem (PD). Indirect measures, such as the values of k, m, n, $\bar{M}$, and $\underline{M}$, however, may sometimes give some indication of this number.

5. Concluding Remarks

We have presented a Bisection-Extreme Point Search Algorithm for problem (PD) which finds an exact optimal solution after a finite number of iterations. The algorithm is the first to be suggested for globally solving problem (PD). In contrast to algorithms suggested for the more general problem (PI) or for the special case (PC), the Bisection-Extreme Point Search Algorithm can be implemented by using only well-known pivoting and optimization subroutines. Furthermore, the bisection search procedure used in the algorithm makes it adaptable to large-scale problems or to problems with many local optima. Finally, efficient implementation of the algorithm can yield important computational benefits. For these reasons, the algorithm is a potentially important practical tool for solving problem (PD) and, hence, for multiple criteria decision making.

References

1. Bazaraa, M. S., Jarvis, J. J., and Sherali, H. D. (1990), *Linear Programming and Network Flows*, Wiley, New York.
2. Benson, H. P. (1986), An Algorithm for Optimizing over the Weakly-Efficient Set, *European J. of Operational Research* **25**, 192–199.
3. Benson, H. P. (1991), An All-Linear Programming Relaxation Algorithm for Optimizing over the Efficient Set, *J. of Global Optimization* **1**, 83–104.
4. Benson, H. P. (1981), Finding an Initial Efficient Extreme Point for a Linear Multiple Objective Program, *J. of the Operational Research Society* **32**, 495–498.
5. Benson, H. P. (1992), A Finite, Non-Adjacent Extreme Point Search Algorithm for Optimization over the Efficient Set, *J. of Optimization Theory and Applications* **73**, 47–64.
6. Benson, H. P. (1981), Optimization over the Efficient Set, Discussion Paper No. 35, Center for Econometrics and Decision Sciences, University of Florida, Gainesville, Florida.
7. Benson, H. P. (1984), Optimization over the Efficient Set, *J. of Mathematical Analysis and Applications* **98**, 562–580.
8. Bolintineanu, S. (1990), Minimization of a Quasi-Concave Function over an Efficient Set, Mathematics Research Paper No. 90–15, Department of Mathematics, La Trobe University, Bundoora, Victoria, Australia.
9. Dauer, J. P. (1991), Optimization over the Efficient Set Using an Active Constraint Approach, *Zeitschrift für Operations Research* **35**, 185–195.
10. Dessouky, M. I., Ghiassi, M., and Davis, W. J. (1986), Estimates of the Minimum Nondominated Criterion Values in Multiple-Criteria Decision-Making, *Engineering Costs and Production Economics* **10**, 95–104.
11. Ecker, J. G. and Hegner, N. S. (1978), On Computing an Initial Efficient Extreme Point, *J. of the Operational Research Society* **29**, 1005–1007.
12. Ecker, J. G. and Kouada, I. A. (1978), Finding All Efficient Extreme Points for Multiple Objective Linear Programs, *Mathematical Programming* **14**, 249–261.
13. Evans, G. W. (1984), An Overview of Techniques for Solving Multiobjective Mathematical Programs, *Management Science* **30**, 1268–1282.
14. Evans, J. P. and Steuer, R. E. (1973), Generating Efficient Extreme Points in Linear Multiple Objective Programming: Two Algorithms and Computing Experience, in J. L. Cochrane and M. Zeleny (eds.), *Multiple Criteria Decision Making*, University of South Carolina Press, Columbia, South Carolina, pp. 349–365.
15. Forgo, F. (1988), *Nonconvex Programming*, Akademiai Kiado, Budapest.
16. Horst, R. and Tuy, H. (1990), *Global Optimization: Deterministic Approaches*, Springer-Verlag, Berlin.

17. Isermann, H. (1977), The Enumeration of the Set of All Efficient Solutions for a Linear Multiple Objective Program, *Operational Research Quarterly* **28**, 711–725.
18. Isermann, H. and Steuer, R. E. (1987), Computational Experience Concerning Payoff Tables and Minimum Criterion Values over the Efficient Set, *European J. of Operational Research* **33**, 91–97.
19. Martos, B. (1965), The Direct Power of Adjacent Vertex Programming Methods, *Management Science* **12**, 241–252.
20. Murty, K. G. (1983), *Linear Programming*, Wiley, New York.
21. Pardalos, P. M. and Rosen, J. B. (1987), *Constrained Global Optimization: Algorithms and Applications*, Springer-Verlag, Berlin.
22. Pardalos, P. M. and Rosen, J. B. (1986), Methods for Global Concave Minimization: A Bibliographic Survey, *SIAM Review* **28**, 367–379.
23. Philip, J. (1972), Algorithms for the Vector Maximization Problem, *Mathematical Programming* **2**, 207–229.
24. Reeves, G. R. and Reid, R. C. (1988), Minimum Values over the Efficient Set in Multiple Objective Decision Making, *European J. of Operational Research* **36**, 334–338.
25. Rosenthal, R. E. (1985), Principles of Multiobjective Optimization, *Decision Sciences* **16**, 133–152.
26. Steuer, R. E. (1986), *Multiple Criteria Optimization: Theory, Computation, and Application*, Wiley, New York.
27. Weistroffer, H. R. (1985), Careful Usage of Pessimistic Values Is Needed in Multiple Objectives Optimization, *Operations Research Letters* **4**, 23–25.
28. Yu, P. L. (1985), *Multiple-Criteria Decision Making*, Plenum, New York.
29. Zeleny, M. (1982), *Multiple Criteria Decision Making*, McGraw-Hill, New York.