BILINEAR PROGRAMMING APPROACH TO OPTIMIZATION OVER THE EFFICIENT SET OF A VECTOR AFFINE FRACTIONAL PROBLEM

LE DUNG MUU AND HOANG QUANG TUYEN

Abstract. We formulate the problem of optimizing a linear function over the weakly efficient set of a multicriteria affine fractional program as a special bilinear problem. For solving the latter problem, we propose a decomposition branch-and-bound algorithm taking into account the affine fractionality of the criterion function. The bounding uses only linear subprograms, the branching takes place over a simplex in the criteria space.

1. Introduction

Consider the multicriteria mathematical programming problem

\[(VP) \quad \text{min}\{F(x) = (f_1(x), \ldots, f_p(x)) \mid x \in X}\]

where \(X \subset \mathbb{R}^n\) is a bounded polyhedral convex set (polytope).

We recall that a vector \(x \in X\) is called efficient (resp. weakly efficient) point for Problem \((VP)\) if there does not exist \(y \in X\) such that \(F(y) \leq F(x), \ F(y) \neq F(x)\) (resp. \(F(y) < F(x)\)). By \(E(F,X)\) (resp. \(WE(F,X)\)) we will denote the set of all efficient (resp. weakly efficient) points. Here and subsequently for two vectors \(a = (a_1, \ldots, a_p)\) and \(b = (b_1, \ldots, b_p)\) the notion \(a < b\) (resp. \(a \leq b\)) mean that \(a_i < b_i\) (resp. \(a_i \leq b_i\)) for all \(i = 1, \ldots, p\). The inner product of two vectors \(a\) and \(b\) is written as \(\langle a, b \rangle\) or \(a^Tb, b^Ta\).

Recently the problems of optimizing a real valued function over the weakly efficient and the efficient sets of \((VP)\) have attracted much attention because of their important applications in decision making. A number of solution methods have been developed for solving these problems (see e.g.\([2, 3, 5, 6, 7, 12, 16, 19, 23]\) and the references therein). Most existing methods have been obtained assuming that Problem \((VP)\) is linear. These methods are based upon one or more of the following properties of efficiency:

- Both the efficient and weakly efficient sets are closed.

---

Received November 22, 2000; in revised form March 12, 2001.

Key words and phrases. Affine fractional, efficiency, optimization over the weakly efficient and efficient sets, bilinear programming, decomposition.

This paper is supported in part by the Basic Research Program in Natural Sciences.
Both these two sets are connected and consist of facets of the constrained polyhedron $X$.

They can be characterized as equalities defined by d.c. or saddle functions.

Unfortunately these properties in general are no longer valid when $(VP)$ is an affine fractional vector optimization problem, i.e., each function $F_i$ is affine fractional (see [8, 18]). Namely, for affine fractional case the efficient set may be neither closed nor open. Both the efficient and the weakly efficient sets do not necessarily consist of facets of constrained polyhedron. Up to now the question of how to formulate these sets as the solution-sets of inequalities or equalities defined by d.c. or saddle functions is, to our knowledge, still open. This explains why the developed methods for linear case could not be applied directly to affine fractional case.

Affine fractional functions are widely used as performance measures in some management situations, production planning and the analysis of financial enterprises. Thus the multicriteria programming problems with affine fractional criterion functions are important and have wide applications.

In this paper we consider the optimization problems over the efficient and weakly efficient sets of a multicriteria affine fractional program. These problems are given respectively as

$$(P) \quad \min \{d^T x \mid x \in E(F, X)\}$$

$$(WP) \quad \min \{d^T x \mid x \in WE(F, X)\}$$

where $E(F, X)$ and $WE(F, X)$ are the efficient and weakly efficient sets of Problem $(VP)$ with $F$ being a vector affine fractional function.

The main difficulty of these problems arises from the fact that both $E(F, X)$ and $WE(F, X)$, are in general neither convex nor given as a constrained set of an ordinary mathematical programming problem.

These problems have been studied in [9, 22]. In [9] Choo and Akin proposed a parametric algorithm for solving $(P)$ with $E(F, X)$ being the efficient set of a bicriteria affine fractional program. This parametric algorithm seems to work well for bicriteria case, but it cannot be extended for three or more criteria. Recently in [13], Malivert shows that Problems $(P)$ and $(WP)$ can be proceeded by solving a sequence of linearly constrained penalized problems of the form

$$\min \{f(x) + t_k p_w(x) \mid x \in X\}$$

where $t_k > 0$ and $p_w$ is a certain penalty function representing the efficient and weakly efficient sets, which are in general neither convex nor differentiable. So the penalized problems remain difficult global optimization ones. No solution method for solving the penalized problems has been discussed in [13].

In our recent paper [22] we used the necessary and sufficient condition for efficiency established by Malivert in [13] to characterize the weakly efficient set by an equality defined by a biconvex function. We proposed there an algorithm for approximating a minimal point of a convex function over the weakly efficient
set \( WE(F, X) \). From a computational point of view, the proposed algorithm of [22] has the disadvantage that it requires solving minimax subproblems rather than linear ones for computing lower bounds.

In this paper we continue our work by refining the algorithm of [22] for approximating a global optimal solution of a linear function over the efficient and weakly efficient sets of the multiple objective affine fractional Problem \((VP)\).

We equivalently formulate Problems \((P)\) and \((WP)\) as special cases of linear programs with additional bilinear constraints. We then propose a decomposition branch-and-bound method for solving the latter problems. The main difference between this algorithm and the algorithm of [22] is that here the subprograms needed to solve are linear whereas in [22] they are minimax problems.

The content of this article is as follows. In Section 2, we first illustrate our work by an example showing that both Problems \((P)\) and \((WP)\), unlike the linear case, do not necessarily attain their optimal solutions among the vertices of the constrained polyhedron \( X \). Next we show how to formulate \((P)\) and \((WP)\) as linear problems with additional bilinear constraints. Section 3 describes an algorithm solving Problem \((WP)\) when all vertices of \( X \) are known, particularly when \( X \) is a simplex. In Section 4 we describe a relaxation algorithm for solving Problem \((WP)\) without requiring that all vertices of \( X \) are known in advance. In Section 5 we show how to approximate a solution of Problem \((P)\) by solving problems of types \((WP)\). Finally, in Section 6, we illustrate the first algorithm by a small example and report some computational results.

2. Preliminaries

To be precise throughout the paper we suppose that the criterion function in the multicriteria programming problem \((VP)\) is given by

\[
F(x) = \left( \frac{A_1 x + s_1}{B_1 x + t_1}, ..., \frac{A_p x + s_p}{B_p x + t_p} \right),
\]

where \( A_i, B_i \) are \( n \)-dimensional vectors; \( s_i, t_i \) are real numbers for all \( i = 1, ..., p \). As usual we assume that \( B_i x + t_i > 0 \) for all \( x \in X \) and all \( i = 1, ..., p \). Thus \( F \) is continuous on \( X \). By definition, the efficient and weakly efficient sets of \((VP)\) can be given by

\[
E(F, X) = \{ x \in X \mid \exists y \in X : F(y) \leq F(x), F(y) \neq F(x) \},
\]

\[
WE(F, X) = \{ x \in X \mid \exists y \in X : F(y) < F(x) \}.
\]

Since \( X \) is compact, the weakly efficient set \( WE(F, X) \) is compact too [8] whereas the efficient set \( E(F, X) \) in general neither closed nor open. Since \( WE(F, X) \) is compact, Problem \((WP)\) has always a global optimal solution. Throughout the paper we assume that \((P)\) has also a global optimal solution.

It is worth pointing out that, in contrast to the linear case, both Problems \((P)\) and \((WP)\) do not necessarily attain their optimal solutions among the vertices of \( X \). To see this let us consider the following example.
\[
\min F(x) = (f_1(x), f_2(x)) = \left( \frac{-x_1}{x_1 + x_2}, \frac{3x_1 - 2x_2}{x_1 - x_2 + 3} \right)
\]
subject to
\[
(x_1, x_2) \in X = \begin{cases}
  x & \text{if } x_1 - 2x_2 \leq 2 \\
  -x_1 - 2x_2 \leq -2 \\
  -x_1 + x_2 \leq 1 \\
  x_1 \leq 6 \\
  x_1 \geq 0, \ x_2 \geq 0
\end{cases}
\]

\[
\min \{-x_1 - x_2 \mid (x_1, x_2) \in WE(F, X)\}
\]

Since the affine fractional function \( f_i(x) \) \((i = 1, 2)\) is monotone along the line segment joining any two points \( y^1 \) and \( y^2 \) in \( X \), it follows that if
\[
\lambda \in (0, 1), \quad f_i(y^1) < f_i(y^2)
\]
then
\[
f_i(y^1) < f_i(\lambda y^1 + (1 - \lambda)y^2) < f_i(y^2) \quad (i = 1, 2)
\]
for every \( \lambda \in (0, 1) \).

Take six points (Figure 1)
\[
y^1 = (0, 1); \ y^2 = (2, 0); \ y^3 = (6, 2),
\]
\[
y^4 = (6, 7); \ y^5 = (2, 3); \ y^6 = \left( \frac{1}{2}, \frac{3}{4} \right).
\]

A simple calculation shows that
\[
F(y^1) = (0, -1), \ F(y^2) = (1, \frac{6}{5}), \ F(y^3) = (\frac{3}{4}, 2),
\]
\[
F(y^4) = (\frac{6}{13}, 2), \ F(y^5) = (\frac{2}{5}, 0), \ F(y^6) = (\frac{2}{5}, 0).
\]

Using the monotonicity of \( f_i(x) \) we can see that the weakly efficient set \( WE(F, X) \) for this example consists of three segments \([y^1, y^5]\), \([y^5, y^6]\) and \([y^6, y^2]\). This can also be verified by using Corollary 1 in [13].

Thus
\[
\min \{f(x) := d^T x = -x_1 - x_2 \mid (x_1, x_2) \in WE(F, X)\}
\]
\[
= f(y^5) = -2 - 3 = -5 < f(y^i), \quad (i = 1, 2, 3, 4).
\]

Clearly, \( y^5 \not\in V(X) = \{y^1, y^2, y^3, y^4\}. \)

The following theorem due to Malivert [13] will be useful for our purpose.

**Theorem 2.1.** [13] A vector \( x \in X \) is efficient (resp. weakly efficient) if and only if there exist real numbers \( \lambda_i > 0 \) (resp. \( \lambda_i \geq 0 \) not all zero) for all \( i = 1, \ldots, p \) such that
\[
\sum_{i=1}^{p} \lambda_i [(B_i x + t_i) A_i - (A_i x + s_i) B_i](x - y) \leq 0 \quad \forall y \in X.
\]

Figure 1

By dividing by \(\sum_{i=1}^{p} \lambda_i > 0\) we can assume that \(\sum_{i=1}^{p} \lambda_i = 1\). So, if

\[
\Lambda_0 := \left\{ \lambda = (\lambda_1, ..., \lambda_p) \mid \lambda > 0, \sum_{i=1}^{p} \lambda_i = 1 \right\},
\]

\[
\Lambda := \left\{ \lambda = (\lambda_1, ..., \lambda_p) \mid \lambda \geq 0, \sum_{i=1}^{p} \lambda_i = 1 \right\}
\]

then

\[
E(F, X) = \left\{ x \in X \mid \exists \lambda \in \Lambda_0, \sum_{i=1}^{p} \lambda_i [(B_i x + t_i) A_i - (A_i x + s_i) B_i](x - y) \leq 0 \quad \forall y \in X \right\}.
\]

\[
WE(F, X) = \left\{ x \in X \mid \exists \lambda \in \Lambda, \sum_{i=1}^{p} \lambda_i [(B_i x + t_i) A_i - (A_i x + s_i) B_i](x - y) \leq 0 \quad \forall y \in X \right\}.
\]
Thus both Problems \((P)\) and \((WP)\) can take the form
\[
\begin{align*}
\text{(IP)} \\
\min \{ f(x) = d^T x \} \\
\text{s.t.} \\
x \in X, \lambda \in \bar{\Lambda}, \\
\sum_{i=1}^{p} \lambda_i [(B_i x + t_i) A_i - (A_i x + s_i) B_i] (x - y) \leq 0 \quad \forall y \in X
\end{align*}
\]
where \(\bar{\Lambda} = \Lambda_0\) for \((P)\) and \(\bar{\Lambda} = \Lambda\) for \((WP)\).

Let \(v^1, \ldots, v^q\) denote the vertices of \(X\). Problem \((IP)\) then can be further reduced due to the following proposition.

**Proposition 2.1.** We have
\[
\sum_{i=1}^{p} \lambda_i [(B_i x + t_i) A_i - (A_i x + s_i) B_i] (x - y) \leq 0 \quad \forall y \in X
\]
if and only if
\[
\sum_{i=1}^{p} \lambda_i [(B_i x + t_i) A_i - (A_i x + s_i) B_i] (x - v^k) \leq 0 \quad \forall k = 1, \ldots, q.
\]

**Proof.** Since \(v^1, \ldots, v^q \in X\), we only need to prove the “only if” part. Since every \(y \in X\) can be expressed as
\[
y = \sum_{k=1}^{q} \gamma_k v^k, \quad 0 \leq \gamma_k \leq 1 \quad \text{and} \quad \sum_{k=1}^{q} \gamma_k = 1,
\]
we have
\[
\sum_{k=1}^{q} \gamma_k \sum_{i=1}^{p} \lambda_i [(B_i x + t_i) A_i - (A_i x + s_i) B_i] (x - v^k) \leq 0 \quad \forall k = 1, \ldots, q
\]

\[
\iff \sum_{i=1}^{p} \lambda_i [(B_i x + t_i) A_i - (A_i x + s_i) B_i] \left( \sum_{k=1}^{q} \gamma_k \right) x - \sum_{k=1}^{q} \gamma_k v^k \right) \leq 0
\]

\[
\iff \sum_{i=1}^{p} \lambda_i [(B_i x + t_i) A_i - (A_i x + s_i) B_i] (x - y) \leq 0 \quad \forall y \in X.
\]

\[\square\]

For simplicity we define
\[
M(\lambda, x, y) := \sum_{i=1}^{p} \lambda_i [(B_i x + t_i) A_i - (A_i x + s_i) B_i] (x - y),
\]
\[
M_k(\lambda, x) := M(\lambda, x, v^k)
\]
\[
= \sum_{i=1}^{p} \lambda_i [(B_i x + t_i) A_i - (A_i x + s_i) B_i] (x - v^k), \quad k = 1, \ldots, q.
\]
By Theorem 2.1, if \((\lambda, x) \in \Lambda \times X\) (resp. \((\Lambda_0 \times X)\)) then \(x\) is weakly efficient (resp. efficient) if and only if
\[
M(\lambda, x, y) \leq 0 \quad \forall y \in X
\]
or
\[
M_k(\lambda, x) \leq 0 \quad \forall k = 1, \ldots, q.
\]
Clearly,
(i) \(M(\lambda, x,)\) is affine on \(X\) for each fixed \((\lambda, x)\);
(ii) For each \(k\), the function \(M_k(\lambda, x)\) is bilinear on \(\Lambda \times X\).

For each \(v^k\) we define
\[
G_k(\lambda) = \sum_{i=1}^{p} \lambda_i [(t_i + B_i v^k)A_i - (s_i + A_i v^k)B_i],
\]
\[
b_k(\lambda) = \sum_{i=1}^{p} \lambda_i [t_i A_i - s_i B_i]v^k.
\]
Let \(G(\lambda)\) denote the \((q \times n)\)-matrix whose \(k\)th row is \(G_k(\lambda)\) \((k = 1, \ldots, q)\), and \(b(\lambda)\) denote the \(q\)-dimensional vector whose \(k\)th entry is \(b_k(\lambda)\). Suppose that
\[
X = \{x \geq 0 \mid Gx \leq b\}.
\]
Then by using Proposition 2.1 and the definitions of \(G(\lambda)\) and \(b(\lambda)\) we can rewrite Problems \((IP)\) as
\[
(P\bar{\Lambda}) \quad \begin{cases} 
\min \{f(x) := d^T x\} \\
\text{s.t.} \\
Gx - b \leq 0, x \geq 0,
\end{cases}
G(\lambda)x - b(\lambda) \leq 0, \lambda \in \bar{\Lambda}.
\]

**Remark 2.1.** To determine Problem \((P\bar{\Lambda})\) we require that all vertices of the polyhedron \(X\) are known in advance. So this formulation is recommended to use when the vertices of \(X\) can be easily calculated. A special case appeared already in some applications (see e.g. [17, 18]) where each component \(x_i\) of the decision variable \(x\) represents the ratio of \(i\)th quantity to be defined. In this case \(X\) is a simplex given by
\[
X := \left\{x = (x_1, \ldots, x_n) \mid \sum_{i=1}^{n} x_i = 1, \ x_i \geq 0 \quad \forall i = 1, \ldots, n \right\}.
\]
Clearly \(X\) has exactly \(n\) vertices which are the unit vectors of \(R^n\).

3. Solution method

From the preceeding section we see that the problems of optimizing a function \(f\) over the efficient and weakly efficient sets of a multicriteria affine fractional program \((VP)\) can be formulated as bilinearly constrained problems of the form \((P\bar{\Lambda})\) with \(\bar{\Lambda} = \Lambda_0\) and \(\bar{\Lambda} = \Lambda\), respectively. The main difference between these
two problems is that $\Lambda$ is a unit simplex where $\Lambda_0 = \text{int} \Lambda$. Problem ($P\Lambda$) is thus easier to be handled.

Bilinear programming is an important topic of mathematical programming. A lot of methods have been developed for solving bilinear programming problems (see e.g. [1, 11, 14, 15, 20, 21] and the references therein). Most of the developed methods have been obtained by assuming that the bilinear term appears in the objective function.

In this section we first describe a decomposition method for solving Problem ($P\bar{\Lambda}$) with $\bar{\Lambda} = \Lambda$ by using its specific structure. Then we will use the fact that $\Lambda_0$ is the interior of $\Lambda$ to approximate a solution of ($P\Lambda_0$).

As usual, for a given $\varepsilon \geq 0$, we call a point $x$ an $\varepsilon$-optimal solution to Problem ($P$) if $x$ is feasible and $f(x) - f_* \leq \varepsilon(\|f(x)\| + 1)$, where $f_*$ denotes the optimal value of ($P$).

The proposed algorithm is a branch-and-bound procedure using Lagrangian bounding operation and a simplicial subdivision. Unlike the algorithm in [22], this algorithm uses only linear programming for computing lower and upper bounds.

Let

$$H(\lambda) := \begin{bmatrix} G \\ G(\lambda) \end{bmatrix}, \quad h(\lambda) = \begin{bmatrix} b \\ b(\lambda) \end{bmatrix}$$

Then Problem ($P\bar{\Lambda}$) can be rewritten as follows

$$\begin{cases} \min \{ f(x) = d^T x \} \\
\text{s.t.} \\
H(\lambda)x - h(\lambda) \leq 0, \\
x \geq 0, \quad \lambda \in \Lambda.
\end{cases}$$

(Lagrangian Bounding Operation)

Define the function $\phi : \Lambda \rightarrow R$ by setting

$$(P_{\lambda}) \quad \phi(\lambda) = \min \{ d^T x \mid H(\lambda)x - h(\lambda) \leq 0, \quad x \geq 0 \}.$$ 

Then the problem

$$(MP) \quad \min \{ \phi(\lambda) \mid \lambda \in \Lambda \}$$

is equivalent to Problems ($B\Lambda$) and ($WP$) in the sense of the following proposition whose proof is obvious from the results of the preceeding section. Let $f_*$ and $w_*$ denote the optimal values of Problem ($P$) and ($WP$) respectively.

**Proposition 3.1.** A point $(\lambda^*, x^*)$ is optimal to Problem ($B\Lambda$) if and only if $x^*$ is optimal to ($WP$) and $\lambda^*$ is optimal to ($MP$) and $w_* = f(x^*) = \phi(\lambda^*)$.

Note that a feasible point of Problem ($B\Lambda$) can be obtained by solving a standard linear program. In fact, if $\lambda \in \Lambda$ fixed and $x^\lambda$ is an optimal solution of the linear problem ($P_{\lambda}$) then $(\lambda, x^\lambda)$ is feasible for ($B\Lambda$). Hence $x^\lambda$ is feasible for ($WP$). So upper bounds for $w_*$ are computed by solving a standard linear
program. As the algorithm executes more and more feasible points can be found, and thereby upper bounds for \( w^* \) can be iteratively improved.

We now compute lower bound for \( w^* \) by using Lagrangian duality. Let \( S \) be a fully dimensional subsimplex of \( \Lambda \). Let \( V(S) \) denote the vertex set of \( S \). Consider Problem (BA) restricted on \( S \), i.e.,

\[
\begin{aligned}
  w^*(S) := \min d^T x \\
  \text{subject to} \\
  H(\lambda)x - h(\lambda) \leq 0, \\
  x \geq 0, \lambda \in S.
\end{aligned}
\]  

(\( BS \))

Let \( L(u, \lambda, x) \) be the Lagrangian function with respected to the constraint \( H(\lambda)x - b(\lambda) \) of this problem. That is

\[
L(u, \lambda, x) = d^T x + u^T (H(\lambda)x - h(\lambda)).
\]

From the Lagrangian duality theorem we have

\[
m(u, \lambda) \leq \phi(\lambda) \quad \forall u \geq 0, \forall \lambda \in S.
\]

(2)

Since for each fixed \( \lambda \) the function \( H(\lambda)x - h(\lambda) \) is affine, we have

\[
\sup_{u \geq 0} m(u, \lambda) = \phi(\lambda).
\]

(3)

Let

\[
\mu_S(u) = \min_{\lambda \in S} m(u, \lambda).
\]

(4)

From (2) it follows that

\[
\mu_S(u) = \min_{\lambda \in S} m(u, \lambda) \leq \min_{\lambda \in S} \phi(\lambda) = w^*(S) \quad \forall u \geq 0.
\]

Hence

\[
\sup_{u \geq 0} \mu_S(u) \leq w^*(S).
\]

Therefore, taking

\[
\beta(S) = \sup_{u \geq 0} \mu_S(u)
\]

(5)

we obtain a lower bound \( \beta(S) \) for \( w^*(S) \). The following lemma states that this lower bound can be computed by solving linear programs one for each vertex of \( S \).

**Lemma 3.1.** Let \( \lambda^i \ (i = 1, \ldots, p) \) be the vertices of \( S \). Then

\[
\beta(S) := \min_{\lambda \in V(S)} \left\{ \sup_{u \geq 0} -h^T(\lambda)u, \text{ s.t.} \right. \\
\left. H^T(\lambda^i)u + d \geq 0, \quad u \geq 0, \quad i = 1, \ldots, p \right\}
\]

(\( LS \))
Proof. From (1) and (5) it follows that
\[ \beta(S) = \sup_{u \geq 0} \mu_S(u) = \sup_{u \geq 0} \min_{\lambda \in S} (u, \lambda). \]
Hence
\[ \beta(S) = \sup_{u \geq 0} \min_{\lambda \in S} \{d^T x + u^T (H(\lambda)x - h(\lambda))\}. \]
Then
\[ \beta(S) = \sup_{u \geq 0} \min_{\lambda \in S} \{ \min_{x \geq 0} (d^T + u^T H(\lambda))x - u^T h(\lambda) \}. \]
If \( \lambda \in S \) is such that \( d^T + u^T H(\lambda) \not\geq 0 \) for all \( u \geq 0 \), then
\[ \min_{x \geq 0} (d^T + u^T H(\lambda))x = -\infty. \]
So, the supremum in (6) can be taken over all \( u \geq 0 \) satisfying
\[ H^T(\lambda)u + d^T \geq 0 \quad \forall \lambda \in S \]
which implies
\[ \min_{x \geq 0} (u^T H(\lambda) + d^T)x = 0. \]
Since
\[ (H^T(\lambda)u + d \geq 0 \quad \forall \lambda \in S) \iff (H^T(\lambda)u + d \geq 0 \quad \forall \lambda \in V(S)), \]
it follows from (6) and (7) that
\[ \beta(S) = \sup_{u \geq 0} \min_{\lambda \in S} \left\{ -u^T h(\lambda) \mid H^T(\lambda)u + d^T \geq 0 \quad i = 1, \ldots, p \right\} \]
with \( \lambda^i \in V(S), i = 1, \ldots, p. \) By the minimax theorem, we can interchange the supremum and the infimum in (8) to obtain
\[ \beta(S) := \min_{\lambda \in V(S)} \left\{ \sup_{-h^T(\lambda)u, \text{ s.t.}} H^T(\lambda^i)u + d \geq 0, \quad i = 1, \ldots, p, \quad u \geq 0, \right\} \]
which proves the lemma. \( \square \)

Simplicial Bisection

At each iteration \( k \) of the algorithm to be described below, a subsimplex of the simplex \( \Lambda \) will be bisected into subsimplices such a way so that as the algorithm executes the obtained lower and upper bounds tend to the same limit. This can be done by using an exhaustive simplicial bisection that is commonly known in global optimization [11]. This simplicial bisection can be described as follows. Let \( S_k \) be a subsimplex of full dimension of \( \Lambda \) that we want to bisect at iteration \( k \). Let \( v^k, w^k \) be two vertices of \( S_k \) such that the edge joining these vertices is longest. Let \( u^k = t_k v^k + (1 - t_k) w^k \) with \( 0 < t_k < 1 \). Bisect \( S_k \) into two subsimplices \( S_{k1} \) and \( S_{k2} \), where \( S_{k1} \) and \( S_{k2} \) are obtained from \( S_k \) by replacing \( v^k \) and \( w^k \) respectively by \( u^k \). It is well known [11] that \( S_k = S_{k1} \cup S_{k2} \), and that if \( \{S_k\} \) is an infinite sequence of nested simplices generated by this simplicial
bisection process such that \( 0 < \delta_0 < t_k < \delta_1 < 1 \) for every \( k \), then the sequence \( \{S_k\} \) shrinks to a singleton.

Now we are in a position to describe the algorithm for solving \((P\tilde{\Lambda})\) with \( \tilde{\Lambda} = \Lambda \).

**Algorithm LB**

*Initialization.* Choose a tolerance \( \varepsilon \geq 0 \) and set \( S_0 := \Lambda \). For each \( v \in V(S_0) \) solve the linear program

\[
(L_v) \quad \beta(v) := \begin{cases} 
\max -h^T(v)u, \text{ s.t.} \\
H^T(\lambda^i)u + d \geq 0, \quad u \geq 0, \quad \forall \lambda^i \in V(S_0).
\end{cases}
\]

Let \( \beta(S_0) := \min_{v \in V(S_0)} \beta(v) \) and \( \lambda^0 \in S_0, u^0 \geq 0 \) such that \( \beta(S_0) = -h^T(\lambda^0)u^0 \).

Solve the linear program

\[
\begin{aligned}
\min \left\{ f(x) := d^T x \right\}, & \text{ s.t.} \\
H(\lambda^0)x - h(\lambda^0) & \leq 0, \quad x \geq 0
\end{aligned}
\]

to obtain \( x^0 \). Let \( \alpha_0 := \phi(\lambda^0) := d^T x^0 \). Set \( \beta_0 := \beta(S_0) \) and

\[
\Gamma_0 := \begin{cases} 
\{S_0\} & \text{if } \alpha_0 - \beta_0 > \varepsilon(|\alpha_0| + 1), \\
\emptyset & \text{otherwise}
\end{cases}
\]

Let \( k \leftarrow 0 \) and go to Iteration \( k \).

*Iteration \( k \) (\( k = 0, 1, ... \))*

**Step \( k1 \) (selection):**

a) If \( \Gamma_k = \emptyset \), then terminate: \( x^k \) is an \( \varepsilon \)-optimal solution and \( \alpha_k \) is the \( \varepsilon \)-optimal value to Problem \((WP)\).

b) If \( \Gamma_k \neq \emptyset \) then select \( S_k \in \Gamma_k \) such that

\[
\beta_k = \beta(S_k) = \min \{ \beta(S) \mid S \in \Gamma_k \}.
\]

**Step \( k2 \) (bisection):** Bisect \( S_k \) into two simplices \( S_{kj}^1 \) and \( S_{kj}^2 \) by the exhaustive simplicial bisection described above.

**Step \( k3 \) (bounding):** Compute, for each \( v \in V(S_{kj}) \),

\[
(L_{S_{kj}}) \quad \beta(v) := \begin{cases} 
\max -h^T(v)u, \text{ s.t.} \\
H^T(\lambda^i)u + d \geq 0, \quad u \geq 0, \quad \forall \lambda^i \in V(S_{kj})
\end{cases}
\]

Set \( \beta(S_{kj}) = \min_{v \in V(S_{kj})} \beta(v) \). Let \( u^{kj} \) be the obtained optimal solution and \( \lambda^{kj} \in V(S_{kj}) \) such that

\[
\beta(S_{kj}) = -h^T(\lambda^{kj})u^{kj}, \quad (j = 1, 2)
\]

**Step \( k4 \) (updating):** Solving the linear programs, one for each \( \lambda^{kj} \)

\[
\begin{aligned}
\min \left\{ f(x) := d^T x \right\}, & \text{ s.t.} \\
H(\lambda^{kj})x - h(\lambda^{kj}) & \leq 0, \quad x \geq 0
\end{aligned}
\]
we obtain new feasible points. Using these feasible points to update the upper bound. Let $x^{k+1}$ be the best feasible point among $x^k$ and the newly generated feasible points. Set $\alpha_{k+1} := d^T x^{k+1}$ and 

$$
\Gamma_{k+1} \gets \{ S \in (\Gamma_k \setminus \{S_k\}) \cup \{S_{k1}, S_{k2}\} \mid \alpha_{k+1} - \beta(S) > \varepsilon(|\alpha_{k+1}| + 1) \}
$$

Increase $k$ by 1 and go to iteration $k$.

**Theorem 3.1.** (i) If Algorithm LB terminates at iteration $k$, then $x^k$ is an $\varepsilon$-global optimal solution to Problem (WP).

(ii) If the algorithm does not terminate then $\beta_k \not\rightarrow w_*$, $\alpha_k \not\rightarrow w_*$ as $k \rightarrow +\infty$, and any cluster point of the sequence $\{x^k\}$ is a globally solution to Problem (WP).

**Proof.** (i) If the algorithm LB terminates at iteration $k$ then $\Gamma_k = \emptyset$. This implies that $\alpha_k - \beta_k \leq \varepsilon(|\alpha_k| + 1)$. Since $\beta_k \leq w_*$, and $\alpha_k = f(x^k)$ it follows that $f(x^k) - w_* \leq \varepsilon(|f(x^k)| + 1)$. Hence $x^k$ is an $\varepsilon$-optimal solution.

(ii) Since $S_k = S_{k1} \cup S_{k2}$, by the rule for computing lower bound $\beta(S_k)$ we have 

$$
\beta_k = \beta(S_k) \leq \beta(S_{k+1}) = \beta_{k+1} \forall k.
$$

Also, since $\alpha_{k+1}$ is the currently smallest upper bound determined at Step $k4$ we have $\alpha_{k+1} \leq \alpha_k \forall k$. Thus, both $\beta_* = \lim \beta_k$ and $\alpha_* = \lim \alpha_k$ exist and satisfy 

$$
\beta_* \leq w_* \leq \alpha_*
$$

Suppose that the algorithm does not terminate. Then it generates an infinite sequence of nested simplices that, for simplicity of notation, we also denote by $\{S_k\}$. Since the subdivision is exhaustive, this sequence strinks to a singleton, say $\lambda^* \in \Lambda$. By the rule for computing lower bound $\beta_k$ we have 

$$
\beta_k = \sup_{u \geq 0} \min_{\lambda \in S_k} m(u, \lambda) \geq \min_{\lambda \in S_k} m(u, \lambda) \forall u \geq 0.
$$

Since the sequence $\{S_k\}$ tends to $\lambda^*$ as $k \rightarrow +\infty$ we obtain 

$$
\beta_* = \lim \beta_k \geq m(u, \lambda^*) \forall u \geq 0.
$$

Since $\phi(\lambda^k)$ is an upper bound determined at Step $k1$ and $\alpha_{k+1}$ is the currently smallest upper bound obtained at Step $k4$ we have 

$$
\alpha_{k+1} \leq \phi(\lambda^k) \forall k.
$$

Since $\lambda^k \rightarrow \lambda^*$, it follows from the continuity of $\phi$ [4] that 

$$
\alpha_* = \lim \alpha_k = \lim \alpha_{k+1} \leq \lim \phi(\lambda^k) = \phi(\lambda^*).
$$

On the other hand, by Lagrangian duality theorem for the linear program determining $\phi(\lambda^*)$ we have 

$$
\sup_{u \geq 0} m(u, \lambda^*) = \phi(\lambda^*).
$$

Then from (10) and (11) it follows that 

$$
\alpha_* \leq \phi(\lambda^*) \leq \beta_*
$$
which together with (9) implies
\[ \beta_* = w_* = \alpha_* = \phi(\lambda^*). \]
Let \( x^* \) be any cluster point of the sequence \( \{x^k\} \). By the definition we have \( \alpha_k = f(x^k) \). Since \( \alpha_k \downarrow w_* \), it follows from continuity of \( f \) that \( w_* = f(x^*) \). Since \( x^k \in WE(F,X) \) for all \( k \) and \( WE(F,X) \) is closed [8], \( x^* \in WE(F,X) \). Hence \( x^* \) is a globally optimal solution to Problem (WP).

Remark 3.1. (i) When \( \varepsilon > 0 \) the algorithm must terminate after a finite number of iterations. Indeed, if the algorithm does not terminate at iteration \( k \), then \( \alpha_k - \beta_k > \varepsilon(|\alpha_k| + 1) \). Since \( \alpha_k - \beta_k \to 0 \) as \( k \to +\infty \), it follows that when \( \varepsilon > 0 \) the inequality \( \alpha_k - \beta_k > \varepsilon(|\alpha_k| + 1) \) cannot happen indefinitely.

(ii) The subdivision takes place on the simplex \( \Lambda \) whose dimension is just equal to the number of the criteria of Problem (VP) that is just equal to the number of linear programs needed to solve for computing lower bounds. So the algorithm is expected to be efficient only when the number of the criteria in (VP) is small.

4. Relaxation approach

The algorithm described in the previous section requires that all vertices of the constrained polyhedron \( X \) are known in advance. So the algorithm can be used only when the vertices of \( X \) are easy to compute. In the case where computing all vertices of \( X \) is expensive, we propose a relaxation algorithm that computes vertices of \( X \) iteratively in a branch-and-bound procedure. It may expect that the algorithm finds a globally optimal solution without computing all of vertices of \( X \). To this end, we suppose that some vertices \( v^1, ..., v^r \) of \( X \) are known. As before, for each \( v^k \) we define
\[
G_k(\lambda) = \sum_{i=1}^{p} \lambda_i[(t_i + B_i v^k)A_i - (s_i + A_i v^k)B_i],
\]
\[
b_k(\lambda) = \sum_{i=1}^{p} \lambda_i[t_iA_i - s_iB_i]v^k.
\]
Denote by \( G^r(\lambda) \) the \((r \times p)\) matrix whose \( k \)th row is \( G_k(\lambda) \) \( (k = 1, ..., r) \), and by \( b^r(\lambda) \) the \( r \)-dimensional vector whose \( k \)th entry is \( b_k(\lambda), (k = 1, ..., r) \) and
\[
H^r(\lambda) := \begin{bmatrix} G \\ G^r(\lambda) \end{bmatrix}, \quad h^r(\lambda) = \begin{bmatrix} b \\ b^r(\lambda) \end{bmatrix}.
\]

Consider the following problem
\[
(B_r\Lambda)
\]
\[
\begin{aligned}
\min \{f(x) := d^T x\}, \text{ s.t. } \\
H^r(\lambda)x - h^r(\lambda) \leq 0, \ x \geq 0, \ \lambda \in \Lambda.
\end{aligned}
\]
Let \( (\lambda^r, x^r) \) be an optimal solution of this problem. If \( x^r \) is weakly efficient, then \( x^r \) is an optimal solution to (WP). Otherwise, the optimal value of \( (B_r\Lambda) \) is a lower bound for that of \( (BA) \), because the feasible set of \( (BA) \) is contained in
the feasible domain of \((B_r\Lambda)\). Then one can increase lower bound by adding a new vertex of \(X\) to obtain Problem \((B_{r+1}\Lambda)\) and so on. Since the number of the vertices of \(X\) is finite, this procedure must terminate yielding an optimal solution to \((B\Lambda)\).

This procedure can be described in details as follows. Let the tolerance \(\epsilon > 0\) be given.

**Algorithm RLB**

**Step 0.** Choose one or more distinct vertices \(v^1, ..., v^r\) of \(X\).

**Step 1.** Solve \((B_r\Lambda)\) by using Algorithm LB. Let \((\lambda^r, x^r)\) be an \(\epsilon\)-optimal solution to \((B_r\Lambda)\).

**Step 2.** Solve the linear program

\[
\max \{ M(\lambda^r, x^r, y) \mid y \in X \}
\]

\((L_r)\)

to obtain an optimal solution \(v^{r+1} \in V(X)\).

a) If \(M(\lambda^r, x^r, v^{r+1}) \leq 0\), then terminate: \((\lambda^r, x^r)\) is an \(\epsilon\)-optimal solution to \((BA)\) (hence \(x^r\) is weakly efficient and therefore it is an \(\epsilon\)-optimal solution to \((WP)\)).

b) If \(M(\lambda^r, x^r, v^{r+1}) > 0\), then use \(v^{r+1}\) to define Problem \((B_{r+1}\Lambda)\). Increase \(r\) by one and go back to Step 1.

**Theorem 4.1.** Algorithm RLB terminates after a finite number of Step 1 yielding an \(\epsilon\)-optimal solution to Problem \((B\Lambda)\).

**Proof.** By the definition of \(M(\lambda, x, y)\) we see that Problem \((PA)\) is equivalent to the problem

\[
\begin{align*}
\min \{ f(x) := d^T x \}, & \text{ s.t.} \\
M(\lambda, x, v^i) & \leq 0, \ \forall v^i \in V(X) \\
\lambda & \in \Lambda, \ x \in X,
\end{align*}
\]

and the relaxed problem \((B_r\Lambda)\) is equivalent to

\[
\begin{align*}
\min \{ f(x) := d^T x \}, & \text{ s.t.} \\
M(\lambda, x, v^i) & \leq 0, \ i = 1, ..., r \\
\lambda & \in \Lambda, \ x \in X.
\end{align*}
\]

Since \((\lambda^r, x^r)\) is feasible for \((B_r\Lambda)\), we have \(M(\lambda^r, x^r, v^i) \leq 0\) for every \(i = 1, ..., r\). Since

\[
M(\lambda^r, x^r, v^{r+1}) = \max_{x \in X} M(\lambda^r, x^r, x),
\]

it follows from Theorem 1.2 that if \(M(\lambda^r, x^r, v^{r+1}) \leq 0\) then \(x^r\) is weakly efficient, and therefore \((\lambda^r, x^r)\) solves \((BA)\). If \(M(\lambda^r, x^r, v^{r+1}) > 0\), then \(v^{r+1} \neq v^j\) for every \(j = 1, ..., r\), because \(M(\lambda^r, x^r, v^j) \leq 0\) for every \(j = 1, ..., r\). Thus if the algorithm does not terminate, it generates, at each Step 2, a new vertex of \(X\).
that distinct from all previous ones. Hence the algorithm must terminate after a finite number of Step 1, since the number of the vertices of $X$ is finite.

5. THE EFFICIENT SET CASE

Because of nonclosedness of $\Lambda_0$, Problem $(P\Lambda_0)$ is more difficult to handle than $(PA)$. To avoid nonclosedness of $\Lambda_0$, for $\delta > 0$ we define

$$\Lambda_\delta := \{ \lambda = (\lambda_1, ..., \lambda_p) | \lambda_j \geq \delta > 0, \ \forall j, \ \sum_{j=1}^p \lambda_j = 1 \}.$$ 

Clearly, for every $\delta > 0$ sufficiently small, $\Lambda_\delta \neq \emptyset$ and $\Lambda_\delta \subset \Lambda_0 \subset \Lambda$.

Let $\delta > 0$ be fixed such that $\Lambda_\delta \neq \emptyset$. Consider the problem

$$(PA_\delta) \quad \begin{cases} \min \{ f(x) = d^T x \}, & \text{s.t.} \\ H(\lambda)x - h(\lambda) \leq 0, & x \geq 0, \ \lambda \in \Lambda_\delta. \end{cases}$$

Since the feasible domain of this problem is compact, it admits an optimal solution. Corresponding to $(PA_\delta)$ we define the problem

$$(MP_\delta) \quad \min \{ \phi(\lambda) | \ \lambda \in \Lambda_\delta \}.$$ 

Since $\Lambda_\delta$ is compact, Problems $(PA_\delta)$ and $(MP_\delta)$ are equivalent in the sense that if $(\lambda^*, x^*)$ is optimal for $(PA_\delta)$ then $\lambda^*$ is optimal for $(MP_\delta)$. Conversely, if $\lambda^*$ is optimal for $(MP_\delta)$ and $x^*$ optimal solution of problem defining $\phi(\lambda^*)$, then $(\lambda^*, x^*)$ solves $(PA_\delta)$.

Clearly, if $(\lambda^*, x^*)$ is an optimal solution to $(PA)$ and $\lambda^* \in \Lambda_0$ then $(\lambda^*, x^*)$ is optimal solution to $(PA_0)$ as well. Otherwise an optimal solution of $(PA_0)$ can be approximated by solving Problem $(PA_\delta)$ with $\delta > 0$ sufficiently small, as shown in the following proposition.

**Proposition 5.1.** Suppose that Problem $(P)$ has an optimal solution. Let $\{\delta_k\}$ be a decreasing sequence of positive numbers tending to zero. Suppose that $\delta_k$ is small enough such that $\Lambda_{\delta_k} \neq \emptyset$ for every $k$. Let $\lambda^k$ be an optimal solution of Problem $(MP_{\delta_k})$ and $x^k$ be a solution of the problem defining $\phi(\lambda^k)$, i.e., $\phi(\lambda^k) = f(x^k)$. Then $f(x^k) \to f_*$ as $k \to \infty$, and $x^k \in E(F, X)$ for all $k$.

**Proof.** Since $\delta_k > \delta_{k+1} > 0$, we have $\Lambda_{\delta_k} \subset \Lambda_{\delta_{k+1}}$. Let $f_k := \phi(\lambda^k)$. Then

$$f_k = \min \{ \phi(\lambda) | \lambda \in \Lambda_{\delta_k} \} \geq \min \{ \phi(\lambda) | \lambda \in \Lambda_{\delta_{k+1}} \} = \phi(\lambda^{k+1}) = f_{k+1} \geq \min_{\lambda \in \Lambda_0} \phi(\lambda) = f_*.$$ 

Since $\{f_k\}$ is a decreasing sequence bounded from below, there exists $f_0$ such that $f_0 = \lim_{\delta_k \to 0} f_k$.
As \( f_k \leq f \) for all \( k \), it follows that \( f_k \leq f_0 \).

Since \( x^* \in E(X,F) \), there exists \( \lambda^* \in \Lambda_0 \) such that \( \phi(\lambda^*) = f^* \). Thus \( \phi(\lambda^*) \leq f_0 \). We show that \( \phi(\lambda^*) = f_0 \). Indeed, otherwise if \( \phi(\lambda^*) < f_0 \), then \( \lambda^* \not\in \Lambda_{\delta_k} \) for all \( k \) sufficiently large. However, since \( \lambda^* \in \Lambda_0 \), and \( \delta_k \downarrow 0 \), one can easily see that \( \lambda^* \in \Lambda_{\delta_k} \) for all \( \delta_k < \min\{\lambda_1^*, \ldots, \lambda_p^*\} \). We arrive at a contradiction. Thus \( f_0 = \phi(\lambda^*) = f^* \), which implies that \( f(x^k) \to f^* \).

Note that

\[
\lambda^k \in \Lambda_{\delta_k} \subset \Lambda_0 \ \forall \ k.
\]

Thus, since \( x^k \) is an optimal solution of Problem \((P_{\lambda^k})\) defining \( \phi(\lambda^k) \), we have \( x^k \in E(F,X) \) for every \( k \).

Remark 5.1. Since the set \( E(F,X) \) may be not closed, a cluster point of the sequence \( \{x^k\} \) may not be efficient, but it must be weakly efficient because \( E(F,X) \subset WE(F,X) \) and \( WE(F,X) \) is closed.

6. Example and computational results

We now illustrate the LB algorithm by the following example which has been introduced in Section 2.

\[(VP) \quad V \min_{x \in X} F(x) = (f_1(x), f_2(x)) = \left(\frac{-x_1}{x_1 + x_2}, \frac{3x_1 - 2x_2}{x_1 - x_2 + 3}\right)\]

where

\[
X = \{x | Gx \leq b, x \geq 0\},
\]

\[
G = \begin{bmatrix}
1 & -2 \\
-1 & -2 \\
-1 & 1 \\
1 & 0
\end{bmatrix}, \quad b = \begin{bmatrix}
2 \\
-2 \\
1 \\
6
\end{bmatrix}.
\]

Let \( \Lambda = \{\lambda = (\lambda_1, \lambda_2) | \lambda_1 \geq 0, \lambda_2 \geq 0, \lambda_1 + \lambda_2 = 1\} \) and \( d = [-1, -1] \). The problem to be solved is

\[
\min\{d^T x = -x_1 - x_2 | x \in WE(F,X)\}.
\]

Compute

\[
G(\lambda) = \begin{bmatrix}
-\lambda_1 + 8\lambda_2 & -6\lambda_2 \\
9\lambda_2 & -2\lambda_1 - 4\lambda_2 \\
-2\lambda_1 + 7\lambda_2 & 6\lambda_1 \\
-7\lambda_1 + 2\lambda_2 & 6\lambda_1
\end{bmatrix}, \quad b(\lambda) = \begin{bmatrix}
-6\lambda_2 \\
18\lambda_2 \\
42\lambda_2 \\
12\lambda_2
\end{bmatrix}.
\]
Hence
\[
H(\lambda) = \begin{bmatrix}
1 & -2 \\
-1 & -2 \\
-1 & 1 \\
1 & 0 \\
\lambda_1 + 8\lambda_2 & -6\lambda_2 \\
9\lambda_2 & -2\lambda_1 - 4\lambda_2 \\
-2\lambda_1 + 7\lambda_2 & 6\lambda_1 \\
-7\lambda_1 + 2\lambda_2 & 6\lambda_1 \\
\end{bmatrix}, \quad h(\lambda) = \begin{bmatrix}
2 \\
-2 \\
1 \\
6 \\
-6\lambda_2 \\
18\lambda_2 \\
42\lambda_2 \\
12\lambda_2 \\
\end{bmatrix},
\]

\[
H^T(\lambda^1) =
\begin{bmatrix}
+1 & -1 & -1 & -1 & (-\lambda_1^1 + 8\lambda_2^1) & 9\lambda_2^1 & (-2\lambda_1^1 + 7\lambda_2^1) & (-7\lambda_1^1 + 2\lambda_2^1) \\
-2 & -2 & +1 & 0 & -6\lambda_2^1 & (2\lambda_1^1 - 4\lambda_2^1) & 6\lambda_1^1 & 6\lambda_1^1 \\
\end{bmatrix},
\]

\[
H^T(\lambda^2) =
\begin{bmatrix}
+1 & -1 & -1 & +1 & (-\lambda_1^2 + 8\lambda_2^2) & 9\lambda_2^2 & (-2\lambda_1^2 + 7\lambda_2^2) & (-7\lambda_1^2 + 2\lambda_2^2) \\
-2 & -2 & +1 & 0 & -6\lambda_2^2 & (2\lambda_1^2 - 4\lambda_2^2) & 6\lambda_1^2 & 6\lambda_1^2 \\
\end{bmatrix}.
\]

For a fixed \(\lambda = (\lambda_1, \lambda_2) \in V(S_k)\), the problems of determining upper and lower bounds are
\[
\alpha = \begin{bmatrix}
\min\{-x_1 - x_2\}, \text{ s.t.} \\
1 & -2 \\
-1 & -2 \\
-1 & 1 \\
1 & 0 \\
-\lambda_1 + 8\lambda_2 & -6\lambda_2 \\
9\lambda_2 & +2\lambda_1 - 4\lambda_2 \\
-2\lambda_1 + 7\lambda_2 & 6\lambda_1 \\
-7\lambda_1 + 2\lambda_2 & 6\lambda_1 \\
\end{bmatrix} \begin{bmatrix}
x_1 \\
x_2 \\
\end{bmatrix} \leq \begin{bmatrix}
2 \\
1 \\
6 \\
18\lambda_2 \\
42\lambda_2 \\
12\lambda_2 \\
\end{bmatrix},
\]

\[
\beta(S_k) =
\begin{align*}
\beta(\lambda^1) := & \max \left\{ -4u_1 - 2u_2 + u_3 + 6u_4 - 6\lambda_1^1 u_5 + 18\lambda_1^1 u_6 + 42\lambda_1^1 u_7 + 12\lambda_1^1 u_8 \right\} \\
\beta(\lambda^2) := & \max \left\{ -4u_1 - 2u_2 + u_3 + 6u_4 - 6\lambda_2^2 u_5 + 18\lambda_2^2 u_6 + 42\lambda_2^2 u_7 + 12\lambda_2^2 u_8 \right\}
\end{align*}
\]
subject to
\[
\begin{align*}
2u_1 - u_2 - u_3 + u_4 + (-\lambda_1^1 + 8\lambda_2^1)u_5 & + 9\lambda_2^1 u_6 + \\
(-2\lambda_1^1 + 7\lambda_2^1)u_7 + (-7\lambda_1^1 + 2\lambda_2^1)u_8 & - 1 \geq 0 \\
-4u_1 - 2u_2 + u_3 + 0u_4 - 6\lambda_1^1 u_5 + (2\lambda_1^1 - 4\lambda_2^1)u_6 & + 6\lambda_1^1 u_7 + 6\lambda_1^1 u_8 - 1 \geq 0 \\
2u_1 - u_2 - u_3 + u_4 + (-\lambda_1^1 + 8\lambda_2^2)u_5 & + 9\lambda_2^2 u_6 + \\
(-2\lambda_1^2 + 7\lambda_2^2)u_7 + (-7\lambda_1^2 + 2\lambda_2^2)u_8 & - 1 \geq 0 \\
-4u_1 - 2u_2 + u_3 + 0u_4 - 6\lambda_2^1 u_5 + (2\lambda_1^2 - 4\lambda_2^2)u_6 & + 6\lambda_2^2 u_7 + 6\lambda_2^2 u_8 - 1 \geq 0 \\
u_1 \geq 0, u_2 \geq 0, u_3 \geq 0, u_4 \geq 0, & u_5 \geq 0, u_6 \geq 0, u_7 \geq 0, u_8 \geq 0.
\end{align*}
\]
**Initialization.** Choose $\varepsilon = 0.05$ and set $S_0 = [(0, 1), (1, 0)]$. Then

$$\beta(S_0) = \min \{ \beta((0, 1)), \beta((1, 0)) \} = \min \{-13, -13\} = -13.$$

Take $\lambda = (1, 0) \in S_0$, $\alpha_0 = [-1, -1][2, 0]^T = -2$, $\beta_0 = \beta(S_0) = -13$. Since $\alpha_0 - \beta_0 = -2 + 13 = 11 \geq 0.05(-2 + 1) = 0.15$, we have $\Gamma_0 = \{S_0\}$. Set $k := 0$ and go to Iteration $k$.

**Iteration $k$.** The following table presents the result attained by the maple V release 4 package after 27 Iterations.

In order to obtain a preliminary evaluation of the performance of Algorithm RLB, we have written a computer code by Maple V that implements this algorithm. We have used the code to solve thirty one randomly generated problems on a Pentium II personal computer. The computed results are reported in Table 2. In this table, $m_1$ stands for the number of newly added constraints.

<table>
<thead>
<tr>
<th>Iteration $k$</th>
<th>$\beta_k$</th>
<th>Incum-bent $\alpha k + 1$</th>
<th>$x_{k+1}$</th>
<th>Selected partition set</th>
<th>Number of stored partition sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-13</td>
<td>-2</td>
<td>(2, 0)</td>
<td>$[(0, 1), (1, 0)]$</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>-13</td>
<td>&quot;</td>
<td>&quot;</td>
<td>$[(\frac{1}{2}, \frac{1}{2}), (1, 0)]$</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>-13</td>
<td>-3.75</td>
<td>$\frac{3}{2}, \frac{9}{4}$</td>
<td>$[(0, 1), (\frac{1}{2}, \frac{3}{2})]$</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>-13</td>
<td>&quot;</td>
<td>&quot;</td>
<td>$[(0, 1), (\frac{1}{2}, \frac{3}{2})]$</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>-13</td>
<td>&quot;</td>
<td>&quot;</td>
<td>$[\frac{1}{4}, \frac{3}{4}, \frac{1}{2}, \frac{1}{2}]$</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>-12.5</td>
<td>&quot;</td>
<td>&quot;</td>
<td>$[\frac{1}{2}, \frac{1}{2}, \frac{3}{4}, \frac{1}{4}]$</td>
<td>4</td>
</tr>
<tr>
<td>6</td>
<td>-11</td>
<td>&quot;</td>
<td>&quot;</td>
<td>$[\frac{3}{4}, \frac{1}{4}, (1, 0)]$</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>-10.45</td>
<td>&quot;</td>
<td>&quot;</td>
<td>$[\frac{3}{8}, \frac{5}{8}, \frac{1}{2}, \frac{1}{2}]$</td>
<td>6</td>
</tr>
<tr>
<td>8</td>
<td>-9.8</td>
<td>&quot;</td>
<td>&quot;</td>
<td>$[\frac{1}{4}, \frac{3}{4}, \frac{3}{8}, \frac{5}{8}]$</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>-9.26</td>
<td>&quot;</td>
<td>&quot;</td>
<td>$[\frac{1}{2}, \frac{1}{2}, \frac{5}{8}, \frac{3}{8}]$</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>-7.8</td>
<td>&quot;</td>
<td>&quot;</td>
<td>$[\frac{1}{2}, \frac{1}{2}, \frac{9}{16}, \frac{7}{16}]$</td>
<td>5</td>
</tr>
</tbody>
</table>
Table 1 (continue)

<table>
<thead>
<tr>
<th>Iteration $k$</th>
<th>$\beta_k$</th>
<th>Incumbent $\alpha_k + 1$</th>
<th>$x^{k+1}$</th>
<th>Selected partition set</th>
<th>Number of stored partition sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>-7.4</td>
<td>$''$</td>
<td>$''$</td>
<td>$[[\frac{5}{8}, \frac{3}{8}], [\frac{3}{4}, \frac{1}{4}]]$</td>
<td>4</td>
</tr>
<tr>
<td>12</td>
<td>-7.4</td>
<td>-4.69</td>
<td>$\left(\frac{15}{8}, \frac{45}{16}\right)$</td>
<td>$[[\frac{9}{16}, \frac{7}{8}], [\frac{5}{8}, \frac{1}{8}]]$</td>
<td>5</td>
</tr>
<tr>
<td>13</td>
<td>-7.18</td>
<td>$''$</td>
<td>$''$</td>
<td>$[[\frac{19}{32}, \frac{13}{32}], [\frac{5}{8}, \frac{3}{8}]]$</td>
<td>4</td>
</tr>
<tr>
<td>14</td>
<td>-6.27</td>
<td>$''$</td>
<td>$''$</td>
<td>$[[\frac{5}{8}, \frac{13}{32}], [\frac{5}{8}, \frac{3}{8}]]$</td>
<td>5</td>
</tr>
<tr>
<td>15</td>
<td>-6.23</td>
<td>$''$</td>
<td>$''$</td>
<td>$[[\frac{9}{16}, \frac{7}{8}], [\frac{19}{32}, \frac{13}{32}]]$</td>
<td>5</td>
</tr>
<tr>
<td>16</td>
<td>-5.91</td>
<td>$''$</td>
<td>$''$</td>
<td>$[[\frac{5}{8}, \frac{3}{8}], [\frac{11}{32}, \frac{21}{32}]]$</td>
<td>3</td>
</tr>
<tr>
<td>17</td>
<td>-5.68</td>
<td>$''$</td>
<td>$''$</td>
<td>$[[\frac{5}{8}, \frac{3}{8}], [\frac{41}{64}, \frac{23}{64}]]$</td>
<td>3</td>
</tr>
<tr>
<td>18</td>
<td>-5.68</td>
<td>$''$</td>
<td>$''$</td>
<td>$[[\frac{39}{64}, \frac{25}{64}], [\frac{5}{8}, \frac{3}{8}]]$</td>
<td>4</td>
</tr>
<tr>
<td>19</td>
<td>-5.59</td>
<td>$''$</td>
<td>$''$</td>
<td>$[[\frac{3}{4}, \frac{1}{4}], [\frac{7}{8}, \frac{1}{8}]]$</td>
<td>3</td>
</tr>
<tr>
<td>20</td>
<td>-5.55</td>
<td>$''$</td>
<td>$''$</td>
<td>$[[\frac{11}{16}, \frac{5}{16}], [\frac{3}{4}, \frac{1}{4}]]$</td>
<td>3</td>
</tr>
<tr>
<td>21</td>
<td>-5.46</td>
<td>$''$</td>
<td>$''$</td>
<td>$[[\frac{21}{32}, \frac{11}{32}], [\frac{11}{16}, \frac{5}{16}]]$</td>
<td>4</td>
</tr>
<tr>
<td>22</td>
<td>-5.33</td>
<td>$''$</td>
<td>$''$</td>
<td>$[[\frac{5}{8}, \frac{3}{8}], [\frac{11}{16}, \frac{5}{16}]]$</td>
<td>2</td>
</tr>
<tr>
<td>23</td>
<td>-5.31</td>
<td>-4.92</td>
<td>$\left(\frac{63}{32}, \frac{189}{64}\right)$</td>
<td>$[[\frac{83}{128}, \frac{45}{128}], [\frac{21}{32}, \frac{11}{32}]]$</td>
<td>3</td>
</tr>
<tr>
<td>24</td>
<td>-5.30</td>
<td>$''$</td>
<td>$''$</td>
<td>$[[\frac{41}{64}, \frac{23}{64}], [\frac{83}{128}, \frac{45}{128}]]$</td>
<td>2</td>
</tr>
<tr>
<td>25</td>
<td>-5.30</td>
<td>$''$</td>
<td>$''$</td>
<td>$[[\frac{21}{32}, \frac{11}{32}], [\frac{43}{64}, \frac{21}{64}]]$</td>
<td>1</td>
</tr>
<tr>
<td>26</td>
<td>-5.22</td>
<td>$''$</td>
<td>$''$</td>
<td>$[[\frac{21}{32}, \frac{11}{32}], [\frac{85}{128}, \frac{43}{128}]]$</td>
<td>1</td>
</tr>
<tr>
<td>27</td>
<td>-5.17</td>
<td>$''$</td>
<td>$''$</td>
<td>$[[\frac{21}{32}, \frac{11}{32}], [\frac{169}{256}, \frac{87}{256}]]$</td>
<td>0</td>
</tr>
</tbody>
</table>

An $\varepsilon$-global optimal solution is $x = (63/32, 189/64)$.
The $\varepsilon$-optimal value is $f_{\varepsilon} = -4.92$. 
Table 2

<table>
<thead>
<tr>
<th>Pron-lems</th>
<th>p</th>
<th>n</th>
<th>m</th>
<th>ε</th>
<th>$f_\ast$</th>
<th>$m_1$</th>
<th>Iter.</th>
<th>CPU time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>6</td>
<td>0.005</td>
<td>-4.995</td>
<td>4</td>
<td>101</td>
<td>189</td>
</tr>
<tr>
<td>2</td>
<td>9</td>
<td>0.01</td>
<td>-99.5</td>
<td>3</td>
<td>56.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>7</td>
<td>0.01</td>
<td>-120</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>7</td>
<td>0.01</td>
<td>-5.4</td>
<td>2</td>
<td>16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>10</td>
<td>0.01</td>
<td>-179</td>
<td>4</td>
<td>12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>10</td>
<td>-110</td>
<td>4</td>
<td>45</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>76</td>
<td>862</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>10</td>
<td>-90</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>9</td>
<td>10</td>
<td>-180</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>12</td>
<td>13</td>
<td>0.01</td>
<td>-59.3</td>
<td>5</td>
<td>84</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>4</td>
<td>10</td>
<td>0.01</td>
<td>-144</td>
<td>6</td>
<td>164</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>4</td>
<td>10</td>
<td>0.01</td>
<td>-141</td>
<td>6</td>
<td>76</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>4</td>
<td>10</td>
<td>0.01</td>
<td>-144</td>
<td>3</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>10</td>
<td>0.01</td>
<td>-22.5</td>
<td>5</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>4</td>
<td>10</td>
<td>0.01</td>
<td>-66</td>
<td>5</td>
<td>25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>4</td>
<td>10</td>
<td>0.01</td>
<td>0</td>
<td>5</td>
<td>763</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>4</td>
<td>10</td>
<td>0.01</td>
<td>-255</td>
<td>4</td>
<td>20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>4</td>
<td>10</td>
<td>0.01</td>
<td>-249</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>4</td>
<td>10</td>
<td>0.01</td>
<td>-246</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>4</td>
<td>10</td>
<td>0.01</td>
<td>-32</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

References


HANOI INSTITUTE OF MATHEMATICS,
P.O. Box 631 BOHO, 10000 HANOI, VIETNAM