A Polynomial-Space Finite Algorithm for Solving a Class of Reverse Convex Programs

Takahito Kuno[*] and Yoshiyuki Shiguro

Graduate School of Systems and Information Engineering

University of Tsukuba, Tsukuba, Ibaraki 305-8573, Japan

March 2007

Abstract

We develop a new kind of branch-and-bound algorithm to solve a linear program with an additional reverse convex constraint. The proposed algorithm is based on a polynomial-space pivoting algorithm for enumerating feasible bases of a linear program. We show that it generates a globally optimal solution after finitely many pivoting operations with polynomial space.

Key words: Global optimization, reverse convex program, branch-and-bound algorithm, simplex algorithm, Bland's pivoting rule.

1 Introduction

A reverse convex program is an optimization problem whose feasible set is the difference of two convex sets, and hence is also called a d.c. optimization problem. It is known that any continuous optimization problem over a compact set can be described within this class [7, 11, 12]. Included as an important subclass is a linear program with an additional reverse convex constraint (LPARC), in which a linear function $\mathbf{cx}$ is maximized on the difference of a polyhedron D and an open convex set C. This is certainly a special class but still contains various optimization problems. In fact, any linear 0-1 integer program belongs to LPARC because a binary constraint $x \in \{0, 1\}$ is equivalent to a system including a reverse convex constraint: $0 \leq x \leq 1$, $x^2 - x \geq 0$. This also implies that LPARC is NP-hard in the sense of computational complexity [4].

[*]The author was partially supported by the Grand-in-Aid for Scientific Research (C) 17560050 from the Japan Society for the Promotion of Sciences. E-mail: takahito@cs.tsukuba.ac.jp

Since the feasible set $D \setminus C$ is nonconvex, LPARC usually has multiple locally optimal solutions, many of which fail to be globally optimal. The algorithms commonly used to find a globally optimal solution can be classified mainly into two types. The first type repeatedly computes a locally optimal solution $\mathbf{x}^\circ$ and checks its global optimality: $D \cap \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{cx} \geq \mathbf{cx}^\circ\} \subset \overline{C}$ (or $D \cap \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{cx} = \mathbf{cx}^\circ\} \subset \overline{C}$) [5, 8, 10], where $\overline{C}$ is the closure of C. The second type is branch-and-bound, which recursively partitions a simpler polyhedron including D and computes lower and upper bounds for LPARC restricted in each partition set [6, 9, 13]. Both involve enumerating vertices of the intersection of D with a half space, a plane, or a polyhedral cone for example, and cannot be executed effectively without using branch-and-bound implicitly or explicitly. Even for an instance of modest scale, therefore, each of these approaches easily suffers from combinatorial explosion far before yielding an expected solution.

The branch-and-bound algorithm we propose in this paper is totally different from the above approaches. It simply enumerates vertices of D in some orderly way, as searching for an optimal solution to LPARC. The reason why such a naïve method has not been succeeded so far is naturally due to the combinatorial explosion in the number of vertices. However, we will show that it can be realized with polynomial space complexity, using Bland's pivoting rule [2] in the simplex algorithm. After some preliminaries in the next section, we explain in Section 3 a vertex enumeration algorithm proposed by Avis and Fukuda [1]. Using it as a procedure for generating a branching tree, we develop a branch-and-bound algorithm for solving LPARC exactly in a finite time with space complexity $O(mn)$, where (m, n) is the size of the linear system defining D. In Section 4, we close the paper by giving a detailed algorithm description and some remarks.

2 Some preliminaries

The class of problems, LPARC, is generally formulated as follows:

$$\left|\begin{array}{ll} \text{maximize} & \mathbf{cx} \\ \text{subject to} & \mathbf{Ax} \leq \mathbf{b}, \quad \mathbf{x} \geq \mathbf{0} \\ & f(\mathbf{x}) \geq 0, \end{array}\right. \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{b} \in \mathbb{R}^m$ and $\mathbf{c}^\mathsf{T} \in \mathbb{R}^n$. Let

$$D = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{Ax} \leq \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\}, \quad C = \{\mathbf{x} \in \mathbb{R}^n \mid f(\mathbf{x}) < 0\}.$$

We assume that D is nonempty and bounded, and f is a continuous quasiconvex function defined on an open set including D. Although both D and C are convex, the feasible set is their difference $D \setminus C$, and hence can be nonconvex and even disconnected. Therefore,

(1) is a typical multiextremal global optimization problem. If the last constraint $f(\mathbf{x}) \geq 0$ is removed, the problem (1) reduces to an ordinary linear program:

$$\left| \begin{array}{ll} \text{maximize} & \mathbf{cx} \\ \text{subject to} & \mathbf{Ax} \leq \mathbf{b}, \quad \mathbf{x} \geq 0, \end{array} \right. \tag{2}$$

which is obviously far easier to solve, and has an optimal solution $\overline{\mathbf{x}}$ in D by assumption. If $\overline{\mathbf{x}}$ happens to lie outside C, then $\overline{\mathbf{x}}$ is also an optimal solution to (1). To exclude such trivial cases, we assume throughout the paper that

$$D \setminus C \neq \emptyset, \tag{3}$$

$$\max \{\mathbf{cx} \mid \mathbf{x} \in D \setminus C\} < \max \{\mathbf{cx} \mid \mathbf{x} \in D\}. \tag{4}$$

Then we have the following well-known result (see e.g., [5, 7, 11] for proof):

Proposition 2.1. *Under assumptions (3) and (4), the boundary ∂C of C contains all optimal solutions to (1), at least one of which lies on some edge of D.*

We see immediately from this proposition that (1) can be solved in finite time if we search for the optimal solution, say $\mathbf{x}^*$, along edges of D in some orderly way, because there are only a finite number of edges. A simple but reliable way to realize it would be enumeration of the vertices of D. The edges can then be examined from their end points. To give a short sketch of our proposed method, let us recall *Bland's pivoting rule* [2], commonly used to avoid cycling in the simplex algorithm for solving linear programs like (2) [3]. Brand's rule enable us to reach an optimal solution to (2) in finitely many steps starting from any vertex of D. This implies that there exists a unique path consisting of some edges in D from each vertex to an optimal one. If the optimal solution $\overline{\mathbf{x}}$ to (2) is unique, we can therefore constitute a spanning tree T rooted at $\overline{\mathbf{x}}$ on a graph G associated with the vertices and edges of D (see Figure 1). To be more precise, the nodes of this graph G correspond one-to-one with the feasible bases for (2), not with the vertices of D. Anyway, we can visit all vertices of D along the *feasible bases tree* T. Our algorithm is a kind of combinatorial branch-and-bound using this spanning tree T as a branching tree.

3 Search for an optimal solution

Let us introduce a vector $\mathbf{y} \in \mathbb{R}^m$ of m slack variables $x_{n+1}, \ldots, x_{n+m}$ into the linear program (2). Then we have its standard form:

$$\left| \begin{array}{ll} \text{maximize} & \mathbf{cx} \\ \text{subject to} & \mathbf{Ax} + \mathbf{y} = \mathbf{b}, \quad \mathbf{x} \geq \mathbf{0}, \quad \mathbf{y} \geq \mathbf{0}, \end{array} \right. \tag{5}$$

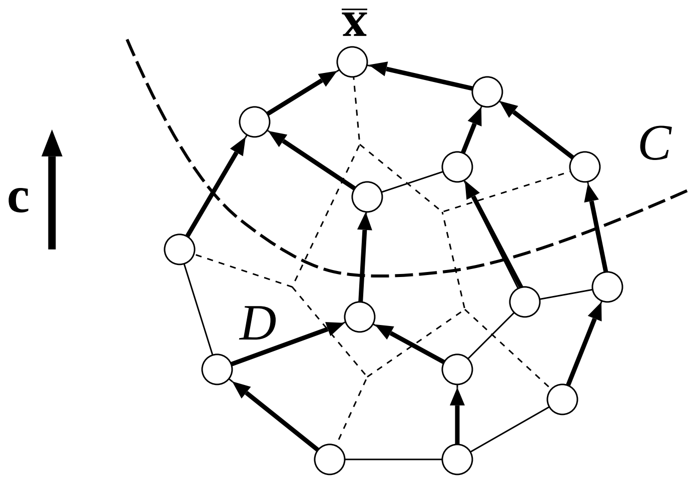

Figure 1: Feasible bases tree T on the polytope D.

whose optimal solution is denoted by $(\overline{\mathbf{x}}, \overline{\mathbf{y}})$. We assume for simplicity that (5) is both primal and dual nondegenerate at $(\overline{\mathbf{x}}, \overline{\mathbf{y}})$. This implies that $\overline{\mathbf{x}}$ is the unique optimal solution to (2). Let $(\mathbf{x}^1, \mathbf{y}^1)$ be a feasible basic solution to (5) and assume $\mathbf{x}^1 \in C$. Such a basic solution must be in D by assumption (4). Let B denote the basis, the index set of m basic variables in $(\mathbf{x}^1, \mathbf{y}^1)$, and let $N = \{1, \ldots, n+m\} \setminus B$. Also denote by $(\mathbf{A}_B, \mathbf{A}_N)$, $(\mathbf{c}_B, \mathbf{c}_N)$ and $(\mathbf{x}_B, \mathbf{x}_N)$ the partitions of columns of $[\mathbf{A}, \mathbf{I}]$, $[\mathbf{c}, \mathbf{0}]$ and $[\mathbf{x}^\mathsf{T}, \mathbf{y}^\mathsf{T}]$, respectively in accordance with the pair (B, N), where $\mathbf{I} \in \mathbb{R}^{m \times m}$ is the identity matrix. Then $\mathbf{A}_B \in \mathbb{R}^{m \times m}$ is nonsingular and we have a feasible dictionary of (5):

$$\left| \begin{array}{ll} \mathbf{x}_B & = \overline{\mathbf{b}} - \overline{\mathbf{A}}_N \mathbf{x}_N \\ z & = z^1 + \overline{c}_N \mathbf{x}_N, \end{array} \right. \tag{6}$$

where

$$\overline{\mathbf{b}} = \mathbf{A}_B^{-1}\mathbf{b}, \quad \overline{\mathbf{A}}_N = \mathbf{A}_B^{-1}\mathbf{A}_N, \quad z^1 = \mathbf{c}_B \mathbf{A}_B^{-1}\mathbf{b}, \quad \overline{\mathbf{c}}_N = \mathbf{c}_N - \mathbf{c}_B \mathbf{A}_B^{-1}\mathbf{A}_N.$$

Letting $\mathbf{x}_N = \mathbf{0}$ in (6), then we have the basic solution $(\mathbf{x}^1, \mathbf{y}^1)$, whose objective function value is given by z^1. To improve $(\mathbf{x}^1, \mathbf{y}^1)$ in the simplex algorithm, we have to select an index $r \in B$ and an $s \in N$ appropriate to change. Bland's pivoting rule specifies this pivot (r, s) uniquely in a very simple manner:

(a) $s \in N$ is the smallest index among all candidates to enter the basis, i.e.,

$$s = \min\{j \in N \mid \overline{c}_j > 0\};$$

(b) $r \in B$ is the smallest index among all candidates to leave the basis, i.e.,

$$r = \min\left\{i \in B \mid \overline{b}_i = \overline{a}_{is} d\right\},$$

where $d = \min\left\{\overline{b}_i/\overline{a}_{is} \mid \overline{a}_{is} > 0,\ i \in B\right\}$.

We refer to the pivot (r, s) satisfying (a) and (b) as the *Bland pivot* in dictionary (6). If (a) is not applicable in (6), then the current basis B is optimal for (5) and we have $(\overline{\mathbf{x}}, \overline{\mathbf{y}}) = (\mathbf{x}^1, \mathbf{y}^1)$. Even otherwise, we have the following (see [2, 3] for proof):

Proposition 3.1. *If the simplex algorithm is applied to (6), then it generates a finite sequence of feasible bases of (5) under Bland's pivoting rule. The last feasible basis in this sequence is optimal for (5).*

Reverse Bland pivot and the feasible bases tree

Let us discuss how to constitute the feasible bases tree T mentioned in Section 2. We think of each feasible basis of (5) as a node of the graph G, and assume an arc between any two bases sharing exactly $m-1$ indices among $\{1, \ldots, n+m\}$. By proposition 3.1, the simplex algorithm traces a unique path on G from each node to the one corresponding to the basis $\overline{B}$ of $(\overline{\mathbf{x}}, \overline{\mathbf{y}})$. Note that $\overline{B}$ is unique, because (5) is assumed to be primal nondegenerate at $(\overline{\mathbf{x}}, \overline{\mathbf{y}})$. Therefore, T consisting of those paths is connected and spans all nodes in G. Suppose $(\mathbf{x}^1, \mathbf{y}^1) \neq (\overline{\mathbf{x}}, \overline{\mathbf{y}})$ and the basis B is yielded from some feasible basis B^2 by applying a pivoting operation to the associated dictionary at its Bland pivot (r, s). Such B^2 might not be unique if exists. Avis and Fukuda [1] named (s, r) as a *reverse Bland pivot* in dictionary (6) and proposed to enumerate the feasible bases of (5) recursively in lexicographic order of reverse Bland pivots from $\overline{B}$. This is essentially equivalent to depth-first search along the spanning tree T from the root node $\overline{B}$.

To check if (s, r) is actually a reverse Brand pivot in (6), we have only to perform a pivoting operation at (s, r). Assuming $\overline{a}_{sr} > 0$, then we have

$$\left|\begin{array}{ll} x_r & = \overline{b}_s/\overline{a}_{sr} - 1/\overline{a}_{sr} x_s - \sum_{j \in N \setminus \{r\}} \overline{a}_{sj}/\overline{a}_{sr} \\ x_i & = (\overline{b}_i - \overline{a}_{ir}\overline{b}_s/\overline{a}_{sr}) + \overline{a}_{ir}/\overline{a}_{sr} x_s - \sum_{j \in N \setminus \{r\}} (\overline{a}_{ij} - \overline{a}_{ir}\overline{a}_{sj}/\overline{a}_{sr}) x_j, \quad i \in B \setminus \{s\} \\ z & = (z^1 + \overline{c}_r \overline{b}_s/\overline{a}_{sr}) - \overline{c}_r/\overline{a}_{sr} x_s + \sum_{j \in N \setminus \{r\}} (\overline{c}_j - \overline{c}_r \overline{a}_{sj}/\overline{a}_{sr}) x_j, \end{array}\right. \tag{7}$$

which is another dictionary of (5) adjacent to (6). We may simply check in (7) if (r, s) satisfies (a) and (b). This brutal method requires $O(mn)$ arithmetic operations. However, from the following lemma, we can see in time $O(m+n)$ whether it is worth carrying out or not.

Lemma 3.2. *If (s, r) is a reverse Bland pivot in (6), then*

$$\overline{c}_r < 0, \tag{8}$$

$$\overline{b}_s = \overline{a}_{sr} d, \tag{9}$$

where $d = \min\left\{\overline{b}_i/\overline{a}_{ir} \mid \overline{a}_{ir} > 0,\ i \in B\right\}$.

Proof. If (9) does not hold, then $\overline{b}_i - \overline{a}_{ir}\overline{b}_s/\overline{a}_{sr} < 0$ for some $i \in B \setminus \{s\}$; and hence (7) is infeasible. Also, r cannot be a candidate to enter the basis in (7) unless (8) holds. $\square$

For each $(i, j) \in B \times N$ in lexicographic order, we test conditions (8) and (9) with $(s, r) = (i, j)$, and perform a pivoting operation if necessary. If (i, j) turns out to be a valid reverse Bland pivot in (6), then we move to the node $B^2 = B \cup \{j\} \setminus \{i\}$ on G. We refer to B^2 as a child of B. If no reverse Bland pivots are found in (6), the current basis B is a leaf of T. In that case, we may backtrack along T according to Bland's usual rule. Thus, we can eventually constitute the feasible bases tree T of the linear program (5) with space complexity only $O(mn)$ for a dictionary.

Procedures necessary for branch-and-bound

Our purpose is not to enumerate the feasible bases of (5), but to find an optimal solution to the reverse convex program (1). We need to do some additional computation to locate the edge of D on which Proposition 2.1 suggests the optimal solution $\mathbf{x}^*$ lies. Suppose (s, r) has passed both tests (8) and (9) in Lemma 3.2, i.e., dictionary (7) is now feasible. Let $B^2 = B \cup \{r\} \setminus \{s\}$ and $(\mathbf{x}^2, \mathbf{y}^2)$ denote the corresponding basic solution. Since C is a convex set and $\mathbf{x}^1$ is assumed to be in C, the segment $[\mathbf{x}^1, \mathbf{x}^2]$ does not intersect ∂C if $\mathbf{x}^2 \in C$. However, if $\mathbf{x}^2 \notin C$, it might be the edge of D suggested by Lemma 2.1. We therefore compute an intersection with ∂C, regardless whether (s, r) is a valid reverse Bland pivot or not in (6).

Lemma 3.3. *The segment $[\mathbf{x}^1, \mathbf{x}^2]$ intersects ∂C at a unique point if $\mathbf{x}^1 \in C$ and $\mathbf{x}^2 \notin C$.*

Proof. Suppose $[\mathbf{x}^1, \mathbf{x}^2]$ intersects ∂C at different points $\mathbf{v}^1$ and $\mathbf{v}^2$ such that $\mathbf{c}^\mathsf{T}\mathbf{x}^1 \leq \mathbf{c}^\mathsf{T}\mathbf{v}^1 \leq \mathbf{c}^\mathsf{T}\mathbf{v}^2 \leq \mathbf{c}^\mathsf{T}\mathbf{x}^2$. Since $\mathbf{x}^1$ belongs to the open set C, there is some open sphere S of center $\mathbf{x}^1$ such that $S \subset C$. Let $\widetilde{S}$ denote the convex hull of S and $\mathbf{v}^2$. Then we have $\mathbf{v}^1 \in \widetilde{S} \setminus \{\mathbf{v}^2\} \subset C$, which contradicts $\mathbf{v}^1 \in \partial C$. $\square$

Let $\mathbf{v}$ denote the unique intersection point of $[\mathbf{x}^1, \mathbf{x}^2]$ with ∂C. If $\mathbf{cv} > \mathbf{cx}^\circ$ for the best feasible solution $\mathbf{x}^\circ$ to (1) found so far, we update the incumbent $\mathbf{x}^\circ$ with $\mathbf{v}$.

Even if (s, r) fails the tests in Lemma 3.2, the basic solution $(\mathbf{x}^2, \mathbf{y}^2)$ can be feasible and $[\mathbf{x}^1, \mathbf{x}^2]$ might intersect ∂C. However, we can neglect it without overlooking $\mathbf{x}^*$.

Lemma 3.4. *Suppose (s, r) fails (8), (9) or both. Then no optimal solutions lie on the segment $[\mathbf{x}^1, \mathbf{x}^2]$ unless $\mathbf{x}^2$ is optimal for (1).*

Proof. If (s, r) fails (9), then dictionary (7) is infeasible as shown in the proof of Lemma 3.2, and so $[\mathbf{x}^1, \mathbf{x}^2] \cap D = \{\mathbf{x}^1\} \subset C$. Suppose (s, r) fails only (8) and $[\mathbf{x}^1, \mathbf{x}^2]$ intersects ∂C at $\mathbf{v}$. Since $\overline{c}_r \geq 0$ and $\overline{a}_{sr} > 0$, we have

$$\mathbf{c}\mathbf{x}^1 = z^1 \leq \mathbf{c}\mathbf{v} \leq \mathbf{c}\mathbf{x}^2 = z^1 + \overline{c}_r \overline{b}_s / \overline{a}_{sr},$$

but $\mathbf{c}\mathbf{x}^2 < \mathbf{c}\mathbf{x}^*$ if $\mathbf{x}^2$ is not optimal for (1). □

Then what should we do if $\mathbf{x}^2$ is an optimal solution to (1) in this lemma? In that case, we may continue pivoting operations as usual at valid reverse Bland pivots. Then we must reach B^2 in finite steps from $\overline{B}$ via some different path than including B because $(\mathbf{x}^2, \mathbf{y}^2)$ is a feasible basic solution to (5) and $B^2 \in T$. In either case, we can ignore the segment $[\mathbf{x}^1, \mathbf{x}^2]$ completely unless (s, r) passes both tests (8) and (9).

In addition to the above observations, we have the following:

Lemma 3.5. *Suppose (s, r) passes both (8) and (9). If for the incumbent $\mathbf{x}^\circ$ we have*

$$z^1 + \overline{c}_r d \leq \mathbf{c}\mathbf{x}^\circ, \tag{10}$$

where $d = \min\left\{\overline{b}_i / \overline{a}_{ir} \mid \overline{a}_{ir} > 0,\ i \in B\right\}$, then B^2 and all its descendants in T have no solutions better than $\mathbf{x}^\circ$.

Proof. The value of $\mathbf{x}^2$ corresponding to B^2 is given by $z^1 + \overline{c}_r d$, as seen in dictionary (7). It is never improved as long as we follow (8) and (9). □

We also see from this lemma that if (10) holds, there are no solutions better than $\mathbf{x}^\circ$ on the segment connecting two points given by each adjacent descendants of B^2. Therefore, we need not move to B^2 from B on T even if (s, r) is a valid reverse Bland pivot in (6) and B^2 is a legitimate child of B. In other words, the value $z^1 + \overline{c}_r d$ serves as an upper bound on $\mathbf{c}\mathbf{x}^*$ to fathom the node B^2 of the branching tree T. It is worth noting that this bounding operation spends little time if dictionary (7) is in hand.

Treatments for degenerate case

Since Bland's pivoting rule terminates the simplex algorithm once an optimal basis is found, the above procedure might not work when the linear program (5) has multiple optimal bases. This can actually occur if (5) is primal or dual degenerate at $(\overline{\mathbf{x}}, \overline{\mathbf{y}})$. Even in that case, each path to an optimal basis on G is still unique under Bland's rule. However, the union T of those paths becomes a spanning forest, not a spanning tree, of G. Therefore, to search for the optimal solution $\mathbf{x}^*$ to (1), we have to collect all optimal bases for (5) and then start our algorithm from each. It is possible to generate all optimal bases for (5) if we enumerate feasible bases of a degenerate system:

$$\left|\begin{array}{l} \mathbf{A}\mathbf{x} + \mathbf{y} = \mathbf{b}, \quad \mathbf{x} \geq \mathbf{0}, \quad \mathbf{y} \geq \mathbf{0} \\ \mathbf{c}\mathbf{x} = \mathbf{c}\overline{\mathbf{x}}, \end{array}\right.$$

using a dual form of Bland's rule. The details are omitted, because they are technically somewhat involved, but introduced in [1]. A more practical method would be, however, to avoid degeneracy itself by giving a little perturbation to $\mathbf{b}$ and $\mathbf{c}$.

4 Algorithm and remarks

Finally, let us summarize the discussion of the previous sections into an algorithm.

```
algorithm TREE_SEARCH
begin
    solve the linear program (5) to obtain an optimal solution $(\overline{\mathbf{x}}, \overline{\mathbf{y}})$;
    let $\mathbf{x}^\circ$ denote any feasible solution to (1);
    for each feasible basis $B$ of value $\mathbf{c}\overline{\mathbf{x}}$ in (5) do begin
        let $N := \{1, \ldots, n\} \setminus B$;
        call procedure REVERSE_BLAND to update $\mathbf{x}^\circ$
    end;
    output $\mathbf{x}^* := \mathbf{x}^\circ$ as an optimal solution to (1)
end;

procedure REVERSE_BLAND
begin
    let $(\mathbf{x}^1, \mathbf{y}^1)$ denote the basic solution for dictionary (6) associated with $(B, N)$;
    let $z^1 := \mathbf{c}_B \mathbf{A}_B^{-1} \mathbf{b}$;
    for each $(s, r) \in B \times N$ in lexicographic order do begin
        if $\overline{c}_r < 0$ in (6) then begin
            let $d := \min\left\{\overline{b}_i / \overline{a}_{ir} \mid \overline{a}_{ir} > 0,\ i \in B\right\}$;
            if $\overline{b}_s / \overline{a}_{sr} = d$ then begin
                perform a pivoting operation at $(s, r)$ in (6) and obtain dictionary (7);
                let $(\mathbf{x}^2, \mathbf{y}^2)$ denote the basic solution for (7);
                if $\mathbf{x}^2 \notin C$ then begin
                    compute the intersection point $\mathbf{v}$ of $[\mathbf{x}^1, \mathbf{x}^2]$ with $\partial C$;
                    if $\mathbf{c}\mathbf{v} > \mathbf{c}\mathbf{x}^\circ$ then $\mathbf{x}^\circ := \mathbf{v}$;
                end;
                if $(r, s)$ is the Bland pivot in (7) and $z^1 + \overline{c}_r d > \mathbf{c}\mathbf{x}^\circ$ then begin
                    let $B := B \cup \{r\} \setminus \{s\}$ and $N := N \cup \{s\} \setminus \{r\}$;
                    call procedure REVERSE_BLAND to update $\mathbf{x}^\circ$;
                    let $B := B \cup \{s\} \setminus \{r\}$ and $N := N \cup \{r\} \setminus \{s\}$;
                end;
                perform a pivoting operation at $(r, s)$ in (7)
            end
        end
    end;
    return $\mathbf{x}^\circ$;
end;
```

In practice, it might not be so easy to compute the intersection point $\mathbf{v}$ of $[\mathbf{x}^1, \mathbf{x}^2]$ with ∂C, depending on the function f defining C, but we assume here that it can be done in finite time. Then finiteness of this recursive algorithm is guaranteed by Proposition 3.1 and Lemma 3.2, and optimality of the output $\mathbf{x}^*$ is by Proposition 2.1 and Lemmas 3.3–3.5. These, together with the fact that it is not necessary to keep anything other than a dictionary and the incumbent $\mathbf{x}^\circ$, lead to the following:

Theorem 4.1. *Algorithm* TREE_SEARCH *occupies $O(mn)$ space and yields a globally optimal solution $\mathbf{x}^*$ to problem (1) after finitely many pivoting operations.*

Since algorithm TREE_SEARCH has not yet been implemented nor compared with other existing algorithms, we cannot make a definitive conclusion regarding the practical efficiency. However, we should remark that TREE_SEARCH visits only vertices of D contained in C. Therefore, it is expected to work well on instances of (1) where C cuts off only a few vertices from D. In contrast to this, for instances where C contains a larger number of vertices of D, the performance of TREE_SEARCH could get worse, and in the worst case, it might visit all vertices but one. To put algorithm TREE_SEARCH into practical use, we need to devise some sort of procedure for reducing the volume of $D \cap C$. A promising one is to cut off a portion of $D \cap C$ using a plane which passes through intersection points of ∂C with n extreme rays of a full-dimensional simplicial cone vertexed in C and including D. The details will be reported elsewhere, with numerical results for comparison with some other algorithms.

References

[1] Avis, D. and K. Fukuda, "A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra", *Discrete and Computational Geometry* **8** (1992), 295–313.

[2] Bland, R.G., "New finite pivoting rules for the simplex method", *Mathematics of Operations Research* **2** (1977), 103–107.

[3] Chvátal, V., *Linear Programming*, Freeman (N.Y., 1983).

[4] Garey, M.R. and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman (San Francisco, 1979).

[5] Hillestad, R.J. and S.E. Jacobsen, "Linear programs with an additional reverse convex constraint", *Applied Mathematics and Optimization* **6** (1980), 257–269.

[6] Horst, R., "Deterministic global optimization with partition sets whose feasibility is not known: application to concave minimization, reverse convex constraints, dc-programming, and Lipschitzian optimization", *Journal of Optimization Theory and Applications* **58** (1988), 11–37.

[7] Horst, R. and H. Tuy, *Global Optimization: Deterministic Approaches*, 2nd ed., Springer-Verlag (Berlin, 1993).

[8] Kuno, T. and J. Shi, "Linear programs with an additional separable concave constraint", *Journal of Applied Mathematics and Decision Sciences* **8** (2004), 155–174.

[9] Nagai, H. and T. Kuno, "A conical branch-and-bound algorithm for a class of reverse convex programs", *Proceedings of the Fourth International Conference on Nonlinear Analysis and Convex Analysis* (2007), 417–426.

[10] Pferschy, U. and H. Tuy, "Linear programs with an additional rank two reverse convex constraint", *Journal of Global Optimization* **4** (1994), 441–454.

[11] Tuy, H., "D.c. optimization: theory, methods and algorithms", *Handbook of Global Optimization*, Kluwer Academic Publishers (Dordrecht, 1995), 149–216.

[12] Tuy, H., *Convex Analysis and Global Optimization*, Kluwer Academic Publishers (Dordrecht, 1998).

[13] Tuy, H. and R. Horst, "Convergence and restart in branch and bound algorithms for global optimization. Application to concave minimization and d.c. optimization problems", *Mathematical Programming* **41** (1988), 161–183.