A Parallel probabilistic algorithm to find "all" vertices of a polytope

Ivan Lavallee, Pham Canh Duong

INRIA

UNITÉ DE RECHERCHE
INRIA-ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
BP 105
78153 Le Chesnay Cedex
France
tél (1) 39 63 55 11

Rapports de Recherche

1 9 9 2

□ème anniversaire

N° 1813

Programme 1
Architectures parallèles, Bases de données,
Réseaux et Systèmes distribués

A PARALLEL PROBABILISTIC ALGORITHM TO FIND "ALL" VERTICES OF A POLYTOPE

Ivan LAVALLEE
Pham Canh DUONG

Décembre 1992

RR_1813

UN ALGORITHME PARALLÈLE PROBABILISTE POUR OBTENIR PRESQUE TOUS LES SOMMETS D'UN POLYTOPE

IVAN LAVALLÉE[1], PHAM CANH DUONG[2]

Résumé

Ce travail présente un algorithme parallèle probabiliste pour obtenir tous les sommets d'un polytope avec une certaine probabilité. L'argument probabiliste est de type Monte-Carlo et l'algorithme ne présuppose rien quant aux problèmes de dégénérescence. De même, la redondance de constraintes dans les données n'influe pas sur l'algorithme.

La partie de l'algorithme qui concerne la localisation d'un sommet utilise un nouvel algorithme de recherche de "point au plus près" dans un polytope.
Mots clefs: point au plus près dans un polytope, sommets, polytope convexe, methode gravitationelle.

---

[1] INRIA Domaine de Voluceau, Rocquencourt BP 105, 78153 LE CHESNAY Cedex, FRANCE.

[2] Institute of Mathematics, P.O. Box 631 Boho, 10000 Hanoi, Vietnam.

A PARALLEL PROBABILISTIC ALGORITHM TO FIND "ALL" VERTICES OF A POLYTOPE

IVAN LAVALLEE[1], PHAM CANH DUONG[2]

Abstract

This paper presents a new parallel algorithm to find all vertices of a convex polytope. This algorithm is of monte-carlo type and requires no assumption of problem nondegeneracy. Redundant constraints have no influence on the main computation. The vertex locating routine uses a new algorithm for nearest point problem in polytopes.

key words: Nearest point problem, vertex, convex polytope, gravitational method.

---

[1] INRIA Domaine de Voluceau, Rocquencourt BP 105, 78153 LE CHESNAY Cedex, FRANCE.

[2] Institute of Mathematics, P.O. Box 631 Boho, 10000 Hanoi, Vietnam.

1 Introduction

The problem of finding all vertices of a polytope, a bounded convex set defined by a finite number of linear inequalities, arises from several fields of research such as economic modeling, pattern recognition theory, nonlinear optimization theory, ... (see[1–4]). For its solution, a large number of methods has been proposed [3, 5, 6, 7]. Existing methods allow to enumerate exactly all vertices of a polytope and are quite efficient for problem of moderate size. For large scale problems, having considerably larger number of vertices, monte–carlo like algorithms seem to be more suitable because they are often simpler to implement and, what is more important, to parallelize. In this case, the notion of *"all vertices"* must be replaced by the one of *"all essential vertices"* that will be discussed in Section 3.

In this paper we present a new parallel algorithm based on stochastic techniques and on an improvement of the gravitational model, proposed by Chang and Murty in [10, 11]. An overview of the method is presented in Section 3. Each of its steps will be discussed in detail in Sections 4, 5 and 6.

2 Preliminaries

Let D be a convex polytope given by

$$D = \{x \in I\!R^n : Ax^T \geq b\}; \tag{1}$$

where A is a $m \times n$–matrix, b is a column vector in $I\!R^m$. Denote by N the index set $N = \{1, \ldots, m\}$ of rows of A.

For simplicity of expression, assume that no redundant constraints are present and that the origin of $I\!R^n$ is a strictly interior point of D. Later, in Section 3, we will show that these assumptions may be removed.

For an n–dimensional polytope with m facets the maximum number of vertices is given by (see [14, 15])

$$V_{\max}(m,n) = \binom{m - [(n+1)/2]}{m-n} + \binom{m - [(n+2)/2]}{m-n}. \tag{2}$$

Let $\mathcal{F}$ be a ℓ–dimensional face of D, i.e. there is a subset $N(\mathcal{F}) \subset N$ such that $|N(\mathcal{F})| = n - \ell$ and the linear subspace

$$H(\mathcal{F}) = \{ x \in I\!R^n : A_i x^T = b_i, \, i \in N(\mathcal{F}) \} \tag{3}$$

is the affine hull of $\mathcal{F}$.

Denote by $A_{\mathcal{F}}$ the matrix (A_i) , $i \in N(\mathcal{F})$. Since there are no redundant constraints, $A_{\mathcal{F}}$ is of full rank which is equal to $(n - \ell)$.

Denote by $Pos(A_{\mathcal{F}})$ the positive hull of vectors A_i , $i \in N(\mathcal{F})$, i.e.

$$Pos(A_{\mathcal{F}}) = \{ x \in I\!R^n : x = \sum_{i \in N(\mathcal{F})} \alpha_i A_i \, , \, \alpha_i \geq 0 \}.$$

Definition 2.1. *For each face $\mathcal{F}$ of D we define the set $Pol_{\mathcal{F}}$ by*

$$Pol_{\mathcal{F}} = \mathcal{F} - Pos(A_{\mathcal{F}}). \tag{4}$$

Noting that if $\mathcal{F}$ is a 0--dimensional face of D, i.e. $\mathcal{F}$ is a vertex of D, then $Pol_{\mathcal{F}}$ is the polar cone of the cone $\{x : A_i x^T \geq b_i \, , \, i \in N(\mathcal{F})\}$, we also call $Pol_{\mathcal{F}}$ the polar cone of the face $\mathcal{F}$.

This notion has a direct relationship with the nearest point problem in D. Let $\mathbf{a} \notin D$ and $\mathbf{a} \in Pol_{\mathcal{F}}$ for some facet $\mathcal{F}$ of D. From (4) there exist a point $p_a \in \mathcal{F}$ and a vector $y \in -Pos(A_{\mathcal{F}})$ such that

$$\mathbf{a} = p_a + y. \tag{5}$$

Theorem 1. *p_a is the nearest point in D to* $\mathbf{a}$.

Proof. From (1) and (3) it is clear that

$$D \subset \{x \in I\!R^n : A_i x^T \geq b_i \, , \, i \in N(\mathcal{F})\}. \tag{6}$$

Set $\tilde{y} = y/\|y\|$ and consider the hyperplane

$$H = \{x \in I\!R^n : \tilde{y} x^T = \tilde{y} p_a^T\}. \tag{7}$$

The hyperplane H passes throw p_a and is orthogonal to $y = \mathbf{a} - p_a$.

Since $\tilde{y} \in -Pos(A_{\mathcal{F}})$, there exist non negative numbers α_i, $i \in N(\mathcal{F})$ such that

$$\tilde{y} = -\sum \alpha_i A_i \ , i \in N(\mathcal{F}).$$

(6) then follows that

$$\tilde{y}x^T \leq \sum_{i \in N(\mathcal{F})} \alpha_i b_i \ \text{ for all } \ x \in D.$$

On the other hand, from (3) we have

$$\tilde{y}p_a^T = \sum_{i \in N(\mathcal{F})} \alpha_i b_i.$$

The last two expressions give

$$\tilde{y}x^T \leq \tilde{y}p_a^T \ \text{ for all } \ x \in D.$$

It means that the hyperplane (7) separates the point a from the polytope D. So, p_a is the nearest point in D to **a** □

The above Theorem has the physical interpretation that, during the projection process onto D, faces of D act like attractors with respect to points in their polar cones. This observation may by formalized as follows.

Let $S(0,r)$ and $B(0,r)$ be resp. the sphere and the ball of radius r and centered at 0. In the sequel, if no confusions can arise, they will be referred to simply as S and B respectively. Suppose that $D \subset B(0,r)$. Now, let's consider the mapping

$$\mathcal{P} : S(0,r) \to D$$

which puts in correspondence to each $x \in S(0,r)$ its nearest point in D, i.e. $\mathcal{P}$ is the projection mapping from $S(0,r)$ onto D.

Definition 2.2. *For a face $\mathcal{F} \subset D$ we define the* domain of attraction *of $\mathcal{F}$ in $S(0,r)$, denoted by $\sigma_{\mathcal{F}}$, to be the intersection of the sphere S with the polar cone $Pol_{\mathcal{F}}$ of $\mathcal{F}$*

$$\sigma_{\mathcal{F}} = S(0,r) \cap Pol_{\mathcal{F}}. \tag{8}$$

Let Σ_r be the area on the spherical surface $S(0,r)$ occupied by the union of domains of attraction of all non zero - dimensional faces of D (the remaining part

$S\backslash\Sigma_r$ of $S(0,r)$ is then occupied by domains of attraction of all vertices of D). Our algorithm is based on the following interesting property of that set :

$$\lim_{r\to\infty} \frac{\text{mes}(\Sigma_r)}{\text{mes}(S)} = 0\,, \tag{9}$$

Where $\text{mes}(\Sigma_r)$ denotes the (jordan) mesure of Σ_r on spherical surface $S(0,r)$. An equivalent form of (9) will be proved in Section 5.

3 Description of the algorithm

Expression (9) means that the probability that a point x, uniformly distributed on the sphere $S(0,r)$, will fall within domains of attraction of vertices of D may be made arbitrarily closed to 1 by increasing r.

First, we present a conceptual description of the algorithm. Given two positive numbers $p < 1$ and $\alpha < 1$, suppose that σ_v is the domain of attraction of some vertex $v \in D$ such that

$$\frac{\text{mes}(\sigma_v)}{\text{mes}(S)} \geq \alpha. \tag{10}$$

Here, α may be considered as the probability that a trial point, drawn at random on S, will fall within σ_v. Then the probability P that at least one trial point out of N sampled points is falling within σ_v is given by

$$P = 1 - (1-\alpha)^N.$$

Solving N gives

$$N = \frac{\log(1-P)}{\log(1-\alpha)}.$$

By (9), choosing the radius r of the sphere S large enough so that

$$\frac{\text{mes}(\Sigma_r)}{\text{mes}(S)} \ll \alpha$$

will guarantee that almost all of N trial points will fall within domains of attraction of vertices of D. Then, after projecting onto D these points gives us all vertices of D whose domain of attraction satisfies (10).

Remark 1. Geometrically, $\text{mes}(\sigma_v)$ may be viewed as a *"mesure of sharpness"* of the vertex v. The larger $\text{mes}(\sigma_v)$ is, the sharper v will be. So, the proposed procedure tends to filter out all vertices that are too *"plane"* (not satisfying (10)). Such a vertex may be called *"non-essential vertex"* of the polytope D. This property may be used to save computer time while dealing with large scale practical problems.

Remark 2. In the case when the number of detected vertices is equal to the maximal expected number of vertices for D, given in (2), we can conclude that all vertices of D are found.

We now describe the algorithm.

Algorithm 1

Initialization : Given $0 < P < 1$ and $0 < \alpha < 1$. Choose the total number N of trial points to be sampled on S as

$$N = \frac{\log(1-P)}{\log(1-\alpha)}.$$

Draw at random l points on S then find their projection on D (l is chosen according to the number of processors available on the computer system in use).

Denote by

X the set of trial points tested so far;
X^* the set of vertices of D found so far;
u the number of points $y \in X$ whose projection onto D is not a vertex of D;
$S_k = S(0, 2^k.r)$ the current bounding sphere.

Step 1. [Sample, reduce sample]

1.1. Sample a point x at random on S_k.
Find $x^* \in X^*$ such that $\|x - x^*\| = \min_{y \in X^*} \|x - y\|$, i.e. x^* is the nearest point in X^* to x.

1.2. [Deleting rule] If x belongs to the domain of attraction of vertex x^*, i.e. if $x \in \sigma_{x^*}$, then delete x and return to 1.1.
Otherwise, go to *Step 2.*

Step 2. [Update X^*]

2.1. Find the projection of x onto D. If the projection point is a vertex of D, add it to X^* and go to *Step 3.*
Otherwise, increase u by 1 then go to substep 2.2.

2.2. [Update S_k] If $\frac{u}{|X|} > (1 - P)$, then replace S_k by $S_{k+1} = S(0, 2^{k+1}.r)$ and return to *Step 1.*
Otherwise, return to *Step 1.*

Step 3. [Stopping rule] If the expected number of vertices is attained or $|X|$ exceeds N, then stop, else return to *Step 1.*

Each Step of the above algorithm will be discussed in detail in next Sections.

4 A new algorithm for nearest point problem in a polytope

The first major problem we have to solve in implementing the above algorithm is how to find the projection of a given point on a convex body. That important problem has been studied by many authors [8, 9, 12, 13]. Efficient algorithms have been proposed for special cases where the convex body is given as a simplicial cone (Murty and Fathi [8]) or as the convex hull of a collection of points in $\mathbb{R}^n$ (P. Wolfe [12, 13]).

In this Section we present a new algorithm based on an original method, called *"stepest descent gravitational method"*, developped by S.Y. Chang and K.G. Murty [10, 11] for solving linear programs. We show that Chang and Murty algorithm may be modified into an efficient algorithm for nearest point problem in a polytope, given in the form (1).

We now recall briefly that method (see [11] for more details). Let's consider the following linear program

$$\begin{aligned} &\text{Minimize} \quad \mathbf{c}x^T ; \\ &\text{subject to} \quad x \in D = \{x \in \mathbb{R}^n \ : \ Ax^T \geq b\} , \end{aligned} \tag{11}$$

where A is a matrix of order $m \times n$, $b \in I\!R^m$ is a column vector, and $\mathbf{c} \in I\!R^n$ is a row vector.

Without loss of generality we can assume that D has an interior point x_0. Let's introduce a havy spherical liquid drop centered at x_0 with radius $\epsilon > 0$, chosen so that the drop is completely contained in D. Make the faces of D impermeable. Then introduce a gravitational force in D in the direction -$\mathbf{c}$ and release the drop. The drop will fall down and, after touching the faces of D, roll down on those faces under the influence of the gravitational force. Then the final halting position of the drop is the lowest possible point in the direction -$\mathbf{c}$, that the drop can get to in D. If the radius of the drop, ϵ, is sufficiently small, the touching faces of (11) at this final halting position, will determine an actual optimum solution of the **LP**(11).

The gravitational method consists of a sequence of stages according to the chosen value of ϵ. Such a stage, in its turn, consists of the following two substeps.

Suppose that the current position of the center of the drop is $\bar{x}$. Denote by $J(\bar{x}) = \{i \ : \ A_i\bar{x} = b_i + \epsilon\|A_i\|\}$, i.e. $J(\bar{x})$ is the index set of touching constraints at that time.

Algorithm 2. (Stepest descent gravitational method. See [11])
Substep 1. Find the *gravitational direction* at the current interior feassible solution. As shown in [11], this involves the solution of a special nearest point problem of the form

$$\begin{aligned} &\text{Minimize} \quad (\mathbf{c} - \eta A_J)(\mathbf{c} - \eta A_J)^T, \\ &\text{subject to} \ \ \eta \geq 0 \, , \end{aligned} \tag{12}$$

where A_J is the submatrix of the matrix A that contains all rows A_i of A such that $i \in J(\bar{x})$. η is a row vector $(\eta_i \ : \ i \in J(\bar{x}))$. Let $\bar{\eta}$ be an optimum solution for (12). The vector $\xi_{\bar{x}} = \mathbf{c} - \bar{\eta}A_J$ is called the *residual vector* in this step. If $\xi_{\bar{x}} \neq 0$, the gravitational direction at $\bar{x}$ is $-\xi_{\bar{x}}/\|\xi_{\bar{x}}\|$.

Substep 2. In this substep, we move the drop straight in the gravitational direction, found in substep 1, to the maximal extent possible, until it is blocked again by the boundary of the polytope D. The step length of the move is determined as the

minimum of λ satisfying the following inequalities

$$\frac{A_i(\bar{x} + \lambda\xi_{\bar{x}}) - b_i}{\|A_i\|} \geq \epsilon \quad \text{for all } i = 1, \ldots, m. \tag{13}$$

After the drop halts, the final position of its center may be used to detect the optimum solution of the **LP**(11). This special step involves projection on the affine subspace defined by the touching constraints at that time, treated as equations. If this final step yields a feassible solution for D, we have the optimum solution for **LP**(11). Otherwise, we reduce the radius ϵ of the drop and restart the algorithm with this smaller drop. The algorithm yields an optimum solution for **LP**(11) after a finite number of stages.

As indicated in [11], this algorithm has several advantages over other methods for linear programming. We note some of them that are important for our purpose.

1. In the gravitational method redundant constraints never enter into the major computation, i.e. in the direction finding routine.

2. The gravitational method does not require the non–degeneracy of the problem for its finite convergence.

This exellent algorithm may be modified in the following way to solve the nearest point problem in a polytope, given as (1). Suppose that we have to define the projection on D of a point **a** lying outside D. Instead of the constant gravitational field, as it was used in [11], we introduce a powerfull gravitational force concentrated around the point **a** (the point **a** then acts like a *"black hole"*). In this case, our liquid drop will move toward the point **a**. Its final position is the point of shortest distance from **a** that the drop can get to in D. This leads to the direction finding and step length finding routines analoguous to those of the Chang and Murty algorithm :

Algorithm 3

Substep 1. [Direction finding routine]

With $\bar{x}$, $J(\bar{x})$, η, A_J, $\xi_{\bar{x}}$ defined as in (12) and $\alpha = (\bar{x} - \mathbf{a})/\|\bar{x} - \mathbf{a}\|$. Solve the

following nearest point problem in a convex cone

$$\begin{aligned} &\text{Minimize} \quad (\alpha - \eta A_J)(\alpha - \eta A_J)^T, \\ &\text{subject to} \quad \eta \geq 0 \,, \end{aligned} \tag{14}$$

Let $\bar{\eta}$ be the optimum solution to (13), define the *residual vector* in this step as

$$\xi_{\bar{x}} \;=\; \alpha - \bar{\eta} A_J.$$

Hence, if $\xi_{\bar{x}} \neq 0$, the moving direction at $\bar{x}$ is $-\xi_{\bar{x}}/\|\xi_{\bar{x}}\|$.

With this moving direction the substep 2 remains unchanged:

Substep 2. [Step length routine] Find $\min \lambda$ that satisfies (13).

The stopping rule must be modified in the following way. Denote by ϕ the moving direction found at the previous pass of substep 1.

Stopping rule. Terminate the current stage if one of the following situations occurs

1. $\xi_{\bar{x}} = 0$;
2. $\xi_{\bar{x}}/\|\xi_{\bar{x}}\| = \phi$. It means that the new moving direction is opposite to the one found at the previous step.

After terminating the current stage, use the *"final special step in a stage"*, described in [11], to determine whether or not a new stage, with smaller drop, must be carried out. If it is not the case, this final step gives us the exact solution to the nearest point problem.

Remark 1. At the end of a stage, following informations are available:

1. The projection x^* of the point x on D;
2. The index set $J(x^*)$ of touching constraints defining x^*;
3. x^* is not a vertex of D if either $|J(x^*)| < n$ or stopping condition 2 holds. This observation is used in Step 2 of our algorithm 1 to detect whether or not the projection of the trial point x is a vertex of D.

Remark 2. In the substep 1.2 ([Deleting rule]) of the algorithm we have to verify the inclusion $x \in \sigma_{x^*}$. This task may be done by projecting x onto the cone $x^* + Pol_{x^*}$. If the resulting point coinsides with x then we have that $x \in x^* + Pol_{x^*}$. Hence, $x \in \sigma_{x^*}$. So, together with all known vertices $x^* \in X^*$, the information about Pol_{x^*}, supplied by the above stopping rule, must be stored.

Now the last thing we need is how to justify the use of the substep 2.2 ([Update S_k]). This question will be discussed in the next Section.

5 Updating the bounding sphere S_k

In the sequel we will denote by $\rho(x, \Sigma)$ the usual hausdorff distance from point x to the set Σ; by $\langle \alpha, \beta \rangle$ the inner product of vectors α and β. Assume that $D \subset B(0,1)$. Define the set Σ to be

$$\Sigma = \bigcup_{\mathcal{F}} (S(0,1) \cap Pos(A_{\mathcal{F}}))$$

for all nonzero–dimensional faces $\mathcal{F}$ of D.

The following theorem is an equivalent form of the expression (9) in Section 2.

Theorem 2. *Given an arbitrarily small number $\epsilon > 0$. For every $x \notin \Sigma$ with $\rho(x, \Sigma) \geq \epsilon$, there exists a positive integer $k_0(\epsilon)$ such that, for all integer $k > k_0(\epsilon)$, the point x belongs to the domain of attraction of some vertex of the reduced polytope $(1/k).D$.*

Proof. Choose $k_0(\epsilon) = (\sqrt{2}/\epsilon)$ and suppose that p_x is the projection of x on $k_0^{-1}.D$. Put $\epsilon_0 = k_0^{-1}$.

Assume the contrary that p_x is in the relative interior of some non trivial facet $\mathcal{F}$ of $\epsilon_0.D$. From (4) x may be written in the form

$$x = p_x + \beta v \tag{15}$$

with some normed vector $v \in -Pos(A_{\mathcal{F}})$ and positive number β.

It is clear that

$$\|p_x\| \leq \epsilon_0 , \tag{16}$$

$$\langle p_x, v\rangle \leq \epsilon_0 . \tag{17}$$

Let y be a point in $S(0,1)$ such that

$$y = 0 + v . \tag{18}$$

By definition of Σ it is obvious that $y \in \Sigma$. As p_x is the projection point of x on $\epsilon_0.D$, the hyperplane H, passing across p_x and orthogonal to v, will separate x from the polytope $\epsilon_0.D$. Since $0 \in \text{int}(\epsilon_0.D)$, this hyperplane separates strictly x from 0. It then implies that

$$\langle p_x, v\rangle > 0 \text{ and } \beta < 1 . \tag{19}$$

As $x \in S(0,1)$ we have

$$(p_x + \beta v)^2 = p_x^2 + 2\beta\langle p_x, v\rangle + \beta^2 = 1 .$$

Combining with (16), (17) and (19) gives

$$\epsilon_0^2 + 2\beta\epsilon_0 + \beta^2 > 1 \Rightarrow (\epsilon_0 + \beta)^2 > 1 .$$

Hence,

$$0 < (1 - \beta) < \epsilon_0 . \tag{20}$$

Now, let's estimate the distance between x and y.

$$\|y - x\|^2 = (v - (p_x + \beta v))^2 = (1 - \beta)^2 v^2 - 2(1 - \beta)\langle p_x, v\rangle + p_x^2 .$$

Taking into account (16), (19) and (20) we deduce

$$\|y - x\|^2 < 2\epsilon_0^2 .$$

Hence,

$$\|y - x\| < \sqrt{2}\epsilon_0 = \epsilon .$$

This contradiction proves the theorem □

Since mes(Σ) = 0 with respect to the mesure defined on the spherical surface $S(0,1)$, Theorem 2 means that with k large enough the probability that a point x, drawn at random on S, will fall into the polar cone of some vertex of D may be made arbitrarily closed to 1. In substep 2.2 of the algorithm 1 we use the value $u/|X|$ to estimate this probability. The magnification coefficient k (which corresponds to the radius of the bounding sphere S_k) is updated each time that estimate value turns out to be greater than α. Hence, the probability that the projection of a trial point is not a vertex of D is much less than α. It allows to reduce the number of nearest point subproblems involved which is the most expensive part in the main computation.

References

[1] L.C. Barbosa and E. Wong, *On a class of interative algorithms for linear inequalities with applications to pattern classification,* in: Proceeding of the first annual Princeton conference of information sciences and systems, 1967, pp. 86–89.

[2] B. Von Hohenbalken, *A finite algorithm to maximize certain pseudoconcave function on polytopes,* Mathematical Programming 9 (1975) 189–206.

[3] T.H. Mattheiss and D.S. Rubin, *A survey and comparison of methods for finding all vertices of convex polyhedral sets,* Math. Oper. Res. **5**, (1980) 167–185.

[4] R. Horst, J. de Vries and N.V. Thoai, *On finding new vertices and redundant constraints in cuting plane algorithms for global optimization,* Oper. Res. Letters, Vol. 7 (1988), No 2, 85–90.

[5] D.B. Khang and O. Fujiwara, *A new algorithm to find all vertices of a polytope,* Oper. Res. Letters **8** (1989) 261–264.

[6] F.A. Al-Khayyal, *An implicit enumeration algorithm for the general linear complementarity problems,* Tech. Report., School of Indust. and Syst. Engin., The Georgia Inst. of Tech. (1985).

[7] M. Manas and J. Nedoma, *Finding all vertices of a convex polyhedral set,* Numer. Math. **9** (1974), 35–39.

[8] B.K. Schmidt and T.H. Mattheiss, *Computational results on an algorithm for finding all vertices of a polytope,* Math. Progr., Vol. **18** (1980), 308–329.

[9] K.G. Murty and Y. Fathi, *A critical index algorithm for nearest point problems on simplicial cones,* Math. Progr. **23** (1982) 206–215.

[10] K.G. Murty, *Linear complementarity, linear and nonlinear programming* (Heldermann, Berlin, F.R.G., 1988).

[11] K.G. Murty, *The gravitational method for linear programming,* Oper. Res. **23** (1986) 206–214.

[12] S.Y. Chang and K.G. Murty, *The stepest descent gravitational method for linear programming,* Discrete Appl. Math. **25** (1989), No 3, 211–241.

[13] P. Wolfe, *Algorithm for a least distance programming problem,* Math. Programming Study 1 (1974) 190–205.

[14] P. Wolfe, *Finding the nearest point in a polytope,* Math. Progr. **11** (1976) 128–149.

[15] V. Klee, *Polytope pairs and their relationship to linear programming,* Acta Mathematica **133** (1974), 1–25.

[16] P. McMullen, *The maximum number of faces of a convex polytope,* Mathematika **17** (1970), 179–184.

ISSN 0249-6399