Numerische Mathematik 12, 226—229 (1968)

Finding All Vertices of a Convex Polyhedron

MIROSLAV MAŇAS and JOSEF NEDOMA

Received November 25, 1967

Summary. The paper describes an algorithm for finding all vertices of a convex polyhedron defined by a system of linear equations and by non-negativity conditions for variables. The algorithm is described using terminology of the theory of graphs and it seems to provide a computationally effective method. An illustrative example and some experiences with computations on a computer are given.

1. Introduction

Let A be a real m by n-matrix, b an m-vector and x an n-vector. Then the set

$$X = \{x;\ A\,x = b,\ \ x \geqq 0\}$$

is a convex polyhedron and a point $x \in X$ is called a vertex of X if and only if all positive components x_i of x are the coefficients of linearly independent vectors in the linear combination

$$\sum_{i=1}^{n} x_i\, a^{(i)} = b, \tag{1}$$

where $a^{(i)}$ denotes the i-th column of the matrix A.

In mathematical programming and in the theory of games arises frequently the problem of finding all vertices of a convex polyhedron. Several methods have been proposed that can be used for this purpose: see e.g. BALINSKI [2], HADLEY [3], MOTZKIN, RAIFFA, THOMPSON and THRALL [6] and UZAWA in [1].

Considering these methods the main emphasis must be laid on the computational efficiency, because the number of vertices may increase very rapidly with increasing m and n. As far as it is known to the authors only the method of BALINSKI has been programmed and used to solve larger problems.

In the next section a new algorithm for finding all vertices of a convex polyhedron is proposed. We describe it using some notions of the theory of graphs as it makes possible to present the algorithm very briefly.

2. Algorithm

To any polyhedron X we can adjoin a graph $\Gamma = (V, U)$ by the following way:

(i) The set of nodes V is formed by m-tuples of integers

$$v = (i_1, i_2, \ldots, i_m), \tag{2}$$

$1 \leqq i_j \leqq m$, $j = 1, 2, \ldots, m$. An m-tuple (2) is in V if and only if there is a vertex of X that is defined by linear combination (1) with vectors $a^{(i_1)}, a^{(i_2)}, \ldots, a^{(i_m)}$.

(ii) Two different nodes $v_1=(i_1, i_2, \ldots, i_m)$ and $v_2=(k_1, k_2, \ldots, k_m)$ have the distance $d \leqq m$ if exactly d components of v_2 are different from components of v_1. v_1 and v_2 are neighbours if they have $d=1$.

(iii) An edge (v_1, v_2) is in U if and only if v_1 and v_2 are neighbours.

If $M \subset V$, let us denote by $\Gamma(M)$ the set that we obtain by adding to M all nodes that have a neighbour in M. By the phrase "to compute a node v" we will mean to compute coordinates of the corresponding vertex of X.

The algorithm for finding all vertices now works as follows: We compute an arbitrary node v_0 and construct two finite sequences of sets of nodes $R_1, R_2, \ldots$ and $W_1, W_2, \ldots$ by the following way: We put $R_1=v_0$ and $W_1=\Gamma(v_0)-v_0$. Further we choose an arbitrary node v_1 from W_1 and compute it. Then we put $R_2=v_0 \cup v_1$ and

$$W_2=W_1 \cup \Gamma(v_1)-R_2.$$

Suppose now that we have constructed the sets R_s and W_s and that $W_s \neq \emptyset$ for $s=1, 2, \ldots, k$. Then we construct the sets R_{k+1} and W_{k+1} as follows: If v_{k-1} is the node last included into R_k, let us check if there is a node in W_k that is a neighbour to v_{k-1}. If there is, denote this node v_k, compute it and put

$$R_{k+1}=R_k \cup v_k \tag{3}$$

and

$$W_{k+1}=W_k \cup \Gamma(v_k)-R_{k+1}. \tag{4}$$

If there is not such a node in W_k, check whether there is a node in W_k having from v_{k-1} the distance 2, if there is not, look for the node having the distance 3, etc. By this way we must find a node having a distance $d \leqq m$. Denote now this node v_k and define again R_{k+1} and W_{k+1} using (3) and (4). Evidently after a finite number of steps we have to come to the stage, where the set $W_k=\emptyset$.

It holds now the following statement: If $W_k=\emptyset$ then $R_k=V$ (and thus we have computed the coordinates of all vertices of X).

Proof. The graph Γ is connected. (We can e.g. always find a path from the node v_1 to v_2 using the simplex method to the linear programming problem that has an optimal solution in a vertex corresponding to v_2 with a vertex corresponding to v_1 as an initial solution.) This is equivalent to the statement that there exist no two non-empty sets M, N such that $M \cup N=V$ and

$$\big(\Gamma(M) \cap N\big) \cup \big(M \cap \Gamma(N)\big)=\emptyset. \tag{5}$$

It follows from the construction of the sets W_k and R_k that

$$W_k=\bigcup_{j=0}^{k-1} \Gamma(v_j)-\bigcup_{j=0}^{k-1} v_j=\Gamma(R_k)-R_k$$

and because $R_k \subset \Gamma(R_k)$, $W_k=\emptyset$ implies $\Gamma(R_k)=R_k$.

Thus $\Gamma(R_k) \cap (V-R_k)=\emptyset$ and $\Gamma(V-R_k) \cap R_k=\emptyset$ and if we denote for a moment $M=R_k$ and $N=V-R_k$, it holds for this sets $M \cup N=V$ and (5). Since $R_k \neq \emptyset$ and since Γ is connected, it must be $V-R_k=\emptyset$, q.e.d.

3. An Example and Computational Experiences

Let us consider the following illustrative example: To find all vertices of the convex polyhedron defined by the relations

$$\begin{aligned} 4x_1+x_2+3x_3+x_4 \qquad &= 24 \\ 3x_1+x_2+2x_3 \qquad -x_5 &= \ 4 \\ x_i \geqq 0, \quad i=1,2,\ldots,5. \end{aligned}$$

Using artificial basis technique in standard simplex method we get the node $v_0=(1,5)$ and the tableau

	x_2	x_3	x_4	
x_1	1/4	3/4	1/4	6
x_5	−1/4	1/4	3/4	14

(6)

We put $R_1=\{(1,5)\}$ and in the tableau (6) we easily find that $W_1=\{(2,5),(3,5),(1,4)\}$. We choose the neighbour $v_1=(2,5)$ from W_1 and compute it:

	x_1	x_3	x_4	
x_2	4	3	1	24
x_5	1	1	1	20

We have $R_2=\{(1,5),(2,5)\}$ and $W_2=\{(3,5),(1,4),(2,4)\}$ and we continue by computing the node $(3,5)$:

	x_1	x_2	x_4	
x_3	4/3	1/3	1/3	8
x_5	−1/3	−1/3	2/3	12

We have $R_3=\{(1,5),(2,5),(3,5)\}$, $W_3=\{(1,4),(2,4),(3,4)\}$. Further we compute $(3,4)$

	x_1	x_2	x_3	
x_3	3/2	1/2	−1/2	2
x_4	−1/2	−1/2	3/2	18

and we get $R_4=\{(1,5),(2,5),(3,5),(3,4)\}$, $W_4=\{(1,4),(2,4)\}$. In the next two steps we compute $(1,4)$ and $(3,4)$:

	x_2	x_3	x_5	
x_1	1/3	2/3	−1/3	4/3
x_4	−1/3	1/3	4/3	56/3

$R_5 = \{(1\ 5), (2, 5), (3, 5), (3, 4), (1, 4)\}$, $W_5 = \{(2, 4)\}$,

	x_1	x_3	x_5	
x_2	3	2	-1	4
x_4	1	1	1	20

Finally $R_6 = \{(1, 5), (2, 5), (3, 5), (3, 4), (1, 4), (2, 4)\}$, $W_6 = \emptyset$. Thus, the vertices of the considered convex polyhedron have the coordinates (6, 0, 0, 0, 14), (0, 24, 0, 0, 24), (0, 0, 8, 0, 12), (0, 0, 2, 18, 0), (4/3, 0, 0, 56/3, 0), (0, 4, 0, 20, 0).

In this simple example we have been able to obtain a new vertex of X in every simplex step, i.e. we went through the nodes of Γ along a Hamiltonian path. This, of course, needs not always to be the case. If the problem is non-degenerate, i.e. if any vertex of X has exactly m positive coordinates, the number of simplex steps required for computation lies between r and mr, where r is the number of all vertices. (In the degenerate case can several nodes of Γ correspond to one vertex.) During the computation we need to store a simplex tableau and an integer array of dimension $r \times m$ for the sets R_k and W_k.

The algorithm has been programmed in ALGOL and used to run a number of examples on an NCR/Elliott 4120 computer. To avoid use of arrays with dynamic bounds a procedure has been included into the program that produces an upper bound for r based on the formula given in [5]. This formula is not yet proved for all values of m and n. It has been used because it gives considerably better results then the other known formulas (given in [4] and [7]). The program includes a device indicating that incorrect estimate of r has been used. E.g. a problem having 10 equations and 16 variables produced 106 vertices in 25 minutes. The upper bound for r is in this case 352 (Formula in [4] gives 616 and that in [7] gives 944). In the case of degeneracy the program may print some vertices several times (any at most n-times). A testing degenerate problem having 4 equations and 8 variables gave 11 vertices, one of which has been printed 5 times.

References

1. ARROW, K. J., L. HURWICZ, and H. UZAWA: Studies in linear and nonlinear programming. Stanford: Stanford University Press 1958.
2. BALINSKI, M. L.: An algorithm for finding all vertices of convex polyhedral sets. J. Soc. Indust. Appl. Mathem. **9**, 72—88 (1961).
3. HADLEY, G.: Linear programming. Reading, Mass: Addison-Wesley Publ. Co. 1962.
4. FILIPOVITCH, E. I., and O. M. KOZLOV: On an estimate of the number of iterations in some linear programming methods. In: Mathematical methods of optimal planning. Novosibirsk: Nauka 1966 [in Russian].
5. KLEE, V.: On the number of vertices of a convex polytope. Canadian Journal of Mathematics **16**, 701—720 (1964).
6. MOTZKIN, T. S., H. RAIFFA, G. L. THOMPSON, and R. M. THRALL: The double description method. In: Contributions to the theory of games, vol. II. Princeton: Princeton University Press 1953.
7. SAATY, T. L.: The number of vertices of a polyhedron. The American Mathematical Monthly **62**, 326—331 (1955).

M. MAŇAS
School of Economics
Dept. of Econometrics
Trojanova 13
Praha 2, CSSR

J. NEDOMA
Institute of Economics
Econometric Laboratory
Pol. vězňů 7
Praha 1, CSSR