Pesquisa Operacional (2022) 42: e254570 p.1-20
doi: 10.1590/0101-7438.2022.042.00254570

Printed version ISSN 0101-7438 / Online version ISSN 1678-5142
www.scielo.br/pope
ARTICLES

VERTEX ENUMERATION OF POLYHEDRA

Caio Lopes Assad[1], Gudelia Morales[2] and José Arica[3*]

Received July 23, 2021 / Accepted May 26, 2022

ABSTRACT. The vertex enumeration problem of a polyhedron P in $\Re^n$, given by m inequalities, is widely discussed in the literature. In this work it is introduced a new algorithm to solve it. The algorithm is based on lexicographic pivoting and the worst-case time complexity is $O\left(m(m+n)^{2\times min\{m,n\}}\right)$, which is $O(mn|V(P)|)$ for the case of non-degenerate polyhedra, where $|V(P)|$ is the number of vertices of P. The proposed algorithm was coded in Matlab and numerical experiments performed for several randomly generated problems show its efficiency.

Keywords: vertex enumeration, polyhedra, pivoting.

1 INTRODUCTION

There are two equivalent forms to represent a polyhedron P in $\Re^n$: as the intersection of m closed half-spaces or as the convex hull of its vertices added to the conic hull of its extreme rays. The vertex enumeration problem of a polyhedron consists of computing its vertex representation from its half-space representation.

This problem is a widely addressed issue in the literature. The formal study of polyhedra dates back to Classical Antiquity, it is already found in The Elements of Euclid, around 300 BC (Encyclopædia Britannica, 2008), but it is only about 2000 years later, in the 1700s, that Euler published a result relating the number of vertices, faces and edges of a three-dimensional polyhedron (Matheiss and Rubin, 1980). Regarding the vertex enumeration problem for the case of non-degenerate polyhedra, there are algorithms that run in polynomial time (i.e., in running time bounded by a polynomial function of input and output size) (Avis and Fukuda, 1982; Dyer, 1983).

*Corresponding author
[1]Universidade Estadual do Norte Fluminense, Laboratório de Engenharia de Produção, Av. Alberto Lamego, 2000, Parque Califórnia, 28013-602, Campos dos Goytacazes-RJ, Brazil – E-mail: caioassad@outlook.com – https://orcid.org/0000-0001-6185-6513
[2]Universidade Estadual do Norte Fluminense, Laboratório de Engenharia de Produção, Av. Alberto Lamego, 2000, Parque Califórnia, 28013-602, Campos dos Goytacazes-RJ, Brazil – E-mail: gudelia@uenf.br – https://orcid.org/0000-0003-4320-8648
[3]Universidade Estadual do Norte Fluminense, Laboratório de Engenharia de Produção, Av. Alberto Lamego, 2000, Parque Califórnia, 28013-602, Campos dos Goytacazes-RJ, Brazil – E-mail: arica@uenf.br – https://orcid.org/0000-0001-7341-5117

But, a problem that remains open is the existence of an algorithm that enumerates the vertices of a bounded polyhedron (a polytope) in polynomial time (Avis et al., 1997; Avis and Fukuda, 1992). In general, since there are polyhedra where the number of vertices grows exponentially accoding to their representation, it is not possible to solve the problem in polynomial time in the number of vertices for any unbounded polyhedron (Provan, 1994; Reimers and Stougie, 2016).

Fundamentally, there are two algorithmic approaches to address the vertex enumeration problem of a polyhedron: those based on pivoting and those based on non-pivoting (Avis et al., 1997). In the first approach, generally, a first search starts from a given vertex, through simplex pivoting. There is a difficulty associated with determining whether a given vertex has already been found, so the vertices are stored in a given tree. Several implementations of polynomial complexity, in relation to the number of vertices, are found in the literature (cf. Dyer, 1983). The second approach is based on the "double description" method by Motzkin et al. (1953), in which the polyhedron is built sequentially, adding a constraint each time, where the new vertices produced are in the hyperplane defined by the constraint which enters.

Many authors have worked on this problem. Matheiss and Rubin (1980) and Dyer (1983) present well known surveys of it. Dyer (1983) introduces an implementation of pivoting algorithms that takes $O\left(mn^2|V(P)|\right)$ time for simple polytopes (where $|V(P)|$ denotes the number of vertices). On the other hand, Chand and Kapur (1970) discovered that the dual problem of vertex enumeration problem is the facet enumeration problem and, lately, Avis and Jordan (2018) and Yang (2021), working on a parallel vertex/facet enumeration code and on a facet enumeration algorithm, respectively, introduced algorithms having similar complexity. Avis and Jordan (2018), as mentioned by Yang (2021), is a good source to find related works. Avis and Fukuda (1982), however, presented a pivoting algorithm that finds all the $|V(P)|$ vertices of a simple polytope (or the $|V(P)|$ faces corresponding to the convex hull of m points in $\Re^n$, where each facet contains exactly n given points) in time $O(mn|V(P)|)$. Lately, Najt (2021), working on the complexity of sampling vertices of a polytope, based on Reimers and Stougie (2016), uses a theoretical approach of the vertex enumeration problem different from the two described in the previous paragraph, taking a branch decomposition perspective to define a parameter of polytopes called branchwidth, which, when bounded, allows solving the problem in polynomial time.

The aim of this paper is to present a new algorithm that solves the vertex enumeration problem of a polyhedron, based on pivots. The main idea of the proposed algorithm is to work with an equivalent maximization problem, where, given a vertex of the polyhedron, a maximization vertex is defined and all the different directed paths from the first to the second vertex are found (composed by edges and vertices of the original polyhedron). Since the proposal may include vertex degeneration, lexicographic pivoting is used. In general, for a general polyhedron, the computational complexity of this algorithm is $O\left(m(m+n)^{2\times min\{m,n\}}\right)$; which, in the case of non-degenerate polyhedra, as in Avis and Fukuda (1992), is $O(mn|V(P)|)$, polynomial in $mn|V(P)|$.

This paper is organized as follows: Section 2 presents the main concepts for defining the problem. Section 3 introduces a linear programming problem equivalent to the addressed problem and

the generation of the paths vertices at the polyhedron is discussed. Section 4 defines a graph associated with the generated vertices. Section 5 introduces the proposed algorithm. Section 6 establishes the computational complexity of the algorithm. Section 7 illustrates the proposal with a numerical example. Section 8 presents some randomly generated numerical tests. At last, Section 9 presents the final considerations.

2 PRELIMINARIES

In this paper, a polyhedron P is defined as the solution set of a system of m linear inequalities in n non-negative variables. Thus,

$$P = \{x \in \Re^n : Ax \leq b, x \geq 0\} \tag{1}$$

where $A \in \Re^{m \times n}$ e $b \in \Re^m$. Here, vectors are considered column matrices. For further discussion of the definitions and proves introduced in this section, the reader can consult several texts on Mathematical Programming (c.f., Blum and Oettli (1975) and Minoux (1986)).

A *vertex* of a polyhedron P is an element $x \in P$ that satisfies, as equalities, a linearly independent set of n of the $m+n$ inequalities that define P. The polyhedron P is said to be *simple* if each vertex is determined by a single set of n linearly independent inequalities. If a polyhedron is not simple, it is said *degenerated*. Here, it is assumed that a vertex $x \in P$ is known. When the polyhedron is assumed to be bounded, it is called a *polytope*. For the algorithm proposed here, under the assumption that P is a polytope or simple, as will be discussed in Section 6, the time complexity is $O(mn|V(P)|)$, where $|V(P)|$ is the number of vertices.

As known, the set of solutions of the polyhedron P can be written as

$$\begin{bmatrix} A & I \end{bmatrix} \begin{pmatrix} x \\ s \end{pmatrix} = b, \quad x \geq 0, s \geq 0 \tag{2}$$

where $I \in \Re^{m \times m}$ denotes the identity matrix. There exists an equivalence between the solutions $y = \begin{pmatrix} x \\ s \end{pmatrix} \in \Re^{n+m}$ of system (2), obtained by zeroing n components, such that the square system thus generated has a unique nonnegative solution, and the vertices $x \in P$. This way, by the selection of a set of m independent columns of $\begin{bmatrix} A & I \end{bmatrix}$, to define a submatrix B, it is possible a partition of matrix $\begin{bmatrix} A & I \end{bmatrix}$ as $\begin{bmatrix} B & N \end{bmatrix}$, permuting the columns of $\begin{bmatrix} A & I \end{bmatrix}$ properly, to have an associated vertex, given by the x variables corresponding to the solution $y = \begin{pmatrix} y_B \\ y_N \end{pmatrix} = \begin{pmatrix} B^{-1}b \\ 0 \end{pmatrix}$ of (2). If $y_B = B^{-1}b \geq 0$, matrix B is called a *feasible basis* and the associated vertex x a *feasible vertex*. The problem now is to find all the feasible vertices.

Note that, in terms of a feasible basis B and an appropriate partition of $\begin{bmatrix} A & I \end{bmatrix}$, system (2) can be written as

$$\begin{bmatrix} B_{11} & 0 & N_{11} & I_{N_2} \\ B_{21} & I_{B_2} & N_{21} & 0 \end{bmatrix} \begin{pmatrix} x_{B_1} \\ s_{B_2} \\ x_{N_1} \\ s_{N_2} \end{pmatrix} = \begin{pmatrix} b_{B_1} \\ b_{B_2} \end{pmatrix} , \tag{3}$$
$$x_{B_1} \geq 0, x_{N_1} \geq 0, s_{N_2} \geq 0, s_{B_2} \geq 0$$

where $B = \begin{bmatrix} B_1 & B_2 \end{bmatrix} = \begin{bmatrix} B_{11} & 0 \\ B_{21} & I_{B_2} \end{bmatrix}$, with B_{11} non-singular, $N = \begin{bmatrix} N_1 & N_2 \end{bmatrix} = \begin{bmatrix} N_{11} & I_{N_2} \\ N_{21} & 0 \end{bmatrix}$.

Thus, for $y_N = \begin{pmatrix} x_{N_1} \\ s_{N_2} \end{pmatrix} = 0$, the vector $y_B = \begin{pmatrix} x_{B_1} \\ s_{B_2} \end{pmatrix} = \begin{pmatrix} B_{11}^{-1} b_{B_1} \\ b_{B_2} - B_{21} B_{11}^{-1} b_{B_1} \end{pmatrix}$ solves the system:

$$\begin{bmatrix} B_{11} & 0 \\ B_{21} & I_{B_2} \end{bmatrix} \begin{pmatrix} x_{B_1} \\ s_{B_2} \end{pmatrix} = \begin{pmatrix} b_{B_1} \\ b_{B_2} \end{pmatrix}, x_{B_1} \geq 0, s_{B_2} \geq 0. \tag{4}$$

The vector $y_B = \begin{pmatrix} x_{B_1} \\ s_{B_2} \end{pmatrix} = \begin{pmatrix} B_{11}^{-1} b_{B_1} \\ b_{B_2} - B_{21} B_{11}^{-1} b_{B_1} \end{pmatrix}$ is called a feasible basic solution associated to basis $\begin{bmatrix} B_1 & B_2 \end{bmatrix} = \begin{bmatrix} B_{11} & 0 \\ B_{21} & I_{B_2} \end{bmatrix}$ and the vector $x = \begin{pmatrix} x_{B_1} \\ x_{N_1} \end{pmatrix} = \begin{pmatrix} B_{11}^{-1} b_{B_1} \\ 0 \end{pmatrix}$ is the correspondent vertex.

Other vertices of the polyhedron P can be found by finding other feasible bases of matrix $\begin{bmatrix} A & I \end{bmatrix}$.

Proposition 1. *Consider system (2) and $\overline{B} = \begin{bmatrix} \overline{B}_1 & \overline{B}_2 \end{bmatrix}$ a feasible basis of matrix $\begin{bmatrix} A & I \end{bmatrix}$, where $\overline{B}_1$ is submatrix of A and $\overline{B}_2$ submatriz of I. If $\begin{pmatrix} x_{\overline{B}_1} \\ s_{\overline{B}_2} \end{pmatrix}$ is the feasible basic solution associated to basis $\overline{B}$, then $x = \begin{pmatrix} x_{\overline{B}_1} \\ x_{\overline{N}_1} \end{pmatrix} = \begin{pmatrix} x_{\overline{B}_1} \\ 0 \end{pmatrix}$ is a vertex of the polyhedron P.*

Proof. The proof is straightforward, from the definition of vertex of a polyhedron. □

3 VERTICES ENUMERATION

Associated to a given vertex $x \in P$ and its associated basis $B = \begin{bmatrix} B_{11} & 0 \\ B_{21} & I_{B_2} \end{bmatrix}$, relation (3) establishes that $z = \sum_{i \in N_1} x_i + \sum_{i \in N_2} s_i = 0$. For any other feasible basis of (3), $\widetilde{B} = \begin{bmatrix} \widetilde{B}_{11} & 0 \\ \widetilde{B}_{21} & I_{\widetilde{B}_2} \end{bmatrix}$, and

the respective $\widetilde{N} = \begin{bmatrix} \widetilde{N}_{11} & I_{\widetilde{N}_2} \\ \widetilde{N}_{21} & 0 \end{bmatrix}$, it holds that $z = \sum_{i \in \widetilde{N}_1} x_i + \sum_{i \in \widetilde{N}_2} s_i \geq 0$. Thus, all the other vertices of polyhedron P, as will be seen in Proposition 3, can be found by considering the feasible basis associated to the next linear program (5):

$$\begin{array}{ll} \textit{maximizar} & z = \sum_{i \in N_1} x_i + \sum_{i \in N_2} s_i \\ & \begin{bmatrix} B_{11} & N_{11} & I_{N_2} & 0 \\ B_{21} & N_{21} & 0 & I_{B_2} \end{bmatrix} \begin{pmatrix} x_{B_1} \\ x_{N_1} \\ s_{N_2} \\ s_{B_2} \end{pmatrix} = \begin{pmatrix} b_{B_1} \\ b_{B_2} \end{pmatrix} . \\ & x_{B_1} \geq 0, x_{N_1} \geq 0, s_{N_2} \geq 0, s_{B_2} \geq 0 \end{array} \tag{5}$$

The spirit of the algorithm to be proposed is to enumerate the vertices of P by finding alternative directed paths, composed of vertices and edges of P, from the given initial vertex x (a minimum of (5)), until a final vertex in the solution set (the set of maxima of (5)) or an extreme ray (if P were unbounded); repeating the process as many times as necessary to find all those different paths.

This way, the simplex *tableau* associated to basis $\begin{bmatrix} B_{11} & 0 \\ B_{21} & I_{B_2} \end{bmatrix}$ for linear program (5) is given, permuting columns conveniently, by the matrix T:

$$T = \left[T_{ij}\right]_{\substack{0 \leq i \leq m \\ 0 \leq j \leq n+m}} = \begin{array}{c} \begin{matrix} \quad x_{B_1} & s_{B_2} & \quad x_{N_1} & \quad s_{N_2} \end{matrix} \\ \begin{pmatrix} 0 & 0^T & 0^T & -1 \ldots -1 & -1 \ldots -1 \\ B_{11}^{-1} b_{B_1} & I_{B_1} & 0 & B_{11}^{-1} N_{11} & B_{11}^{-1} \\ b_{B_2} - B_{21} B_{11}^{-1} b_{B_1} & 0 & I_{B_2} & N_{21} - B_{21} B_{11}^{-1} N_{11} & -B_{21} B_{11}^{-1} \end{pmatrix} \end{array} = \begin{bmatrix} d_0^T \\ d_1^T \\ \vdots \\ d_m^T \end{bmatrix} . \tag{6}$$

At the top of (6), starting at the second column, it is shown the association of the columns of T with the basic variables (x_{B_1} and s_{B_2}) and non-basic variables (x_{N_1} and s_{N_2}) corresponding to x and s, respectively. The first column of T ($[T_{i0}]_{0 \leq i \leq m}$) corresponds to the value of the objective function of the linear program (5) ($T_{00} = z = 0$) evaluated at the current solution $\begin{pmatrix} x \\ s \end{pmatrix}$ and to the value of the basic variables ($[T_{i0}]_{i \in \mathscr{B}_1} = x_{B_1} = B_{11}^{-1} b_{B_1} \geq 0$ and $[T_{i0}]_{i \in \mathscr{B}_2} = s_{B_2} = b_{B_2} - B_{21} B_{11}^{-1} b_{B_1} \geq 0$), respectively. The entries of the first line of T, starting at the second component ($\left[T_{0j}\right]_{1 \leq j \leq n+m}$) correspond to the reduced costs, of the basic and non-basic variables, associated with the current base of (5).

Note that tableau T is feasible, but not optimal for the linear program (5). Therefore, one can try to improve the value of the objective function by changing adequately the current basis. Since there may be degeneration in tableau T, the change of basis can be carried out by using rules that ensure the finite termination of the Simplex Method, such as, for example, the lexicographic criterion (cf. Blum and Oettli, 1975, Chapter 2) or the Bland's Rule (Bland, 1977). Thus, the generation of new feasible bases, starting from the T matrix, will determine new feasible vertices of polyhedron P.

Both rules cited for the finite termination of the Simplex Method (lexicographic criteria and the Bland rule) are used in the literature to treat degeneration, with advantages and disadvantages: "Bland's Rule ... drawback is that your choice of incoming column may not be a very good one"(Dantzig and Thapa, 1997, p. 160); "The first and widely used such tool [to ensure finite termination of simplex methods] is lexicographic simplex rule" (Terlaky, 2001); "Although [Bland's Rule] is much simpler than the lexicographic rule, it also usually takes a lot longer for the Simplex algorithm to converge using this rule" (Lewis, 2008, p. 37). In this work the lexicographic method will be used. For further discussion of the definitions and proofs set forth here, the reader may search for Blum and Oettli, 1975 (Chapter 2).

Definition 1. *Given* $d, f \in \Re^n$, *it is said that*

(i) d *is lexicographically positive, denoted by* $d \succ 0$, *if* $d \neq 0$ *and its first component not zero is positive.*

(ii) $d \succcurlyeq 0$, *if* $d \succ 0 \vee d = 0$.

(iii) $d \succ f$, *if* $d - f \succ 0$.

Since $\left(\Re^n, \succcurlyeq\right)$ is a totally ordered set, any finite set $\{d_i\}_{i\in J} \subset \Re^n$, with all its elements different, has a single element d_{i_o}, such that $d_{i_o} \prec d_i, \forall i \in J$; d_{i_o} is called *lexicographic minimum* of $\{d_i\}_{i\in J}$, and it is denoted by $d_{i_o} = minlex\{d_i, i \in J\}$.

Note that rows of matrix T, from the second onwards, are lexicographically positive (i.e., $d_i^T \succ 0, 1 \leq i \leq m$).

Given the current basic variables, x_{B_1} and s_{B_2}, which in matrix T correspond to the vector $[T_{i0}]_{1\leq i\leq m}$, and the reduced costs of non-basic variables, x_{N_1} and s_{N_2}, which in matrix T correspond to the vector $\left[T_{0j}\right]_{m+1\leq j\leq n+m} = (-1, \ldots, -1, -1, \ldots, -1)$, the lexicographically pivoting is given by:

(PivLex1) (*Rule to determine the non-basic variable that enters the basis)* Consider as non-basic variable that enters the basis anyone corresponding to an index p, such that $T_{0p} \leq 0, m+1 \leq p \leq n+m$.

(PivLex2) (*Rule to determine the basic variable that leaves the basis)* Leaves the basis the variable corresponding to index q defined by the lexicographically pivoting, given by

$$\frac{d_q^T}{T_{qp}} = minlex\left\{\frac{d_i^T}{T_{ip}} : T_{ip} > 0, 1 \leq i \leq m\right\}. \tag{7}$$

If the leaving variable q, *(7), cannot be defined (because* $T_{ip} \leq 0, \forall i$*), then an extreme ray of the feasible set was found.*

Thus, a new matrix $\overline{T}$ is generated, which differs only in a single basic variable of the matrix T. Denoting the set of basic indices associated with matrix $\overline{T}$ by $\overline{\mathscr{B}}$, we have that $\overline{\mathscr{B}} = \overline{\mathscr{B}}_1 \cup \overline{\mathscr{B}}_2 =$

$\mathscr{B}\cup\{p\}\{q\}$, where $\overline{\mathscr{B}}_1$ corresponds to columns of A and $\overline{\mathscr{B}}_2$ to columns of I at the basis. The simplex method by applying, successively, the rules of lexicographic pivoting is finite (even in the case of degeneration) (Blum and Oettli, 1975, Chapter 2).

The tableau

$$\overline{T} = \left[\overline{T}_{ij}\right]_{\substack{0\leq i\leq m \\ 0\leq j\leq n+m}} = \begin{bmatrix} \overline{d}_0^T \\ \overline{d}_1^T \\ \vdots \\ \overline{d}_m^T \end{bmatrix}, \tag{8}$$

obtained from tableau T, has the following format:

$$\begin{array}{c} \overline{d}_i^T = d_i^T - \frac{T_{ip}}{T_{qp}} d_q^T, i\neq q \\ \overline{d}_q^T = \frac{1}{T_{qp}} d_q^T \end{array}. \tag{9}$$

Note that, as in matrix T, $\overline{d}_i^T \succ 0, 1\leq i\leq m$; i.e., $\overline{T}$ is also a feasible tableau for linear program (5) and the associated vertex of polyhedron P is $\bar{x} = \begin{pmatrix} x_{\overline{B}_1} \\ x_{\overline{N}_1} \end{pmatrix} = \begin{pmatrix} [T_{i0}]_{i\in\overline{\mathscr{B}}_1} \\ 0 \end{pmatrix}$, with $z = \overline{d}_{00} \geq 0$. Note that after applied (9), a suitable reordering of the columns of $\overline{T}$ must be done to keep the first m columns basic, as in (6). From here on, it is assumed that the tables obtained from relation (9) are reordered in the sense indicated.

4 THE VERTICES OF THE LINEAR PROGRAM (5) FEASIBLE SET AND AN ASSOCIATED DIRECTED GRAPH

Consider a directed graph whose vertices are feasible solutions associated with feasible bases of the linear program (5), where two vertices are adjacent if the corresponding bases differ only in a single index and the direction of the edges is determined in the sense that the objective function is not decreasing. Thus, if the bases $\mathscr{H}$ and $\mathscr{F}$ differ in a single index, the associated vertices are adjacent. If the values of the objective function at the bases $\mathscr{H}$ and $\mathscr{F}$ are $z_{\mathscr{H}}$ and $z_{\mathscr{F}}$, respectively, and $z_{\mathscr{H}} < z_{\mathscr{F}}$, the edge is directed from the corresponding vertex of $\mathscr{H}$ to that of $\mathscr{F}$. Similarly, if $z_{\mathscr{H}} = z_{\mathscr{F}}$, the edge is undirected (or bidirectional).

Remark 1. *If $\Omega_{\mathscr{H}}$ and $\Omega_{\mathscr{F}}$ denote the sets of vertices obtained by changing of basis with a non-decreasing objective function, from bases $\mathscr{H}$ and $\mathscr{F}$, it is clear that $\Omega_{\mathscr{H}} \supset \Omega_{\mathscr{F}}$.*

Consider a feasible tableau $T^{(r;k,s)}$, where $(r;k,s)$ means that degeneration will occur by a change of basis, where the non-basic variable r enters and it is removed either basic variable

k or s. Matrix (10) shows the format of $T^{(r;k,s)}$, where basic columns k and s and non-basic column r are stand out.

$$T^{(r;k,s)} = \begin{array}{c} \\ \\ \\ \\ \\ \\ \\ \end{array} \begin{array}{c} \begin{array}{cccc} & k & r & s \end{array} \\ \begin{pmatrix} T_{00}^{(r;k,s)}\cdots & 0\cdots & T_{0r}^{(r;k,s)}\cdots & 0\cdots \\ \vdots & \vdots & \vdots & \vdots \\ T_{0k}^{(r;k,s)}\cdots & 1\cdots & T_{kr}^{(r;k,s)} & 0\cdots \\ \vdots & \vdots & \vdots & \vdots \\ T_{0s}^{(r;k,s)}\cdots & 0\cdots & T_{sr}^{(r;k,s)}\cdots & 1\cdots \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix} \end{array} \tag{10}$$

If $T_{kr}^{(r;k,s)} > 0$ and $T_{sr}^{(r;k,s)} > 0$, simple computations show that by introducing the non-basic variable r into the basis and removing variables k and s from the basis, respectively, in table $T^{(r;k,s)}$, the degenerate feasible tableaus $T^{(k;r,s)}$ and $T^{(s;k,r)}$ are generated. In addition, it is possible to generate, from $T^{(k;r,s)}$, the tableau $T^{(s;k,r)}$ and vice versa. Therefore, any of the three tableaus can be obtained from any other. Note that in all cases the value of the objective function is the same.

As consequence, the vertex sets of the polyhedron P, obtained from the tableaux $T^{(k;r,s)}$ and $T^{(s;k,r)}$ are the same. Therefore, to determine the vertices of the polyhedron P, it is sufficient to consider, in the associated graph, only one of the sequences $T^{(r;k,s)}$ and $T^{(k;r,s)}$ or $T^{(r;k,s)}$ and $T^{(s;k,r)}$.

The following proposition establishes that if a vertex is obtained from another one by lexicographical change of basis associated to linear program (5), then, analogously, that vertex can be obtained from this one by considering minimization instead of maximization in (5).

Proposition 2. *If tableau T, given in (6), generates tableau $\overline{T}$, from the linear maximization program (5), by the lexicographic pivot rules and by the relations (8) and (9), then T can be generated from $\overline{T}$, from the linear program (5), substituting the maximization process by the minimization process, by lexicographic pivot rules and by relations (8) and (9).*

Proof. Consider that $\overline{T}$ is obtained from T by changing the entering non-basic variable s by the leaving basic variable r. Thus, $\overline{T}$ is generated from T by replacing the respective

columns corresponding to variables s and r by the respective transformed columns defined by lexicographic pivot rules and by relations (8) and (9); i.e.,

$$\left[\overline{T}_{ir}\right]_{0\leq i\leq m} = \begin{bmatrix} \overline{T}_{0r} \\ \overline{T}_{1r} \\ \vdots \\ \overline{T}_{rr} \\ \vdots \\ \overline{T}_{mr} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \text{ and } \left[\overline{T}_{is}\right]_{0\leq i\leq m} = \begin{bmatrix} \overline{T}_{0s} \\ \overline{T}_{1s} \\ \vdots \\ \overline{T}_{rs} \\ \vdots \\ \overline{T}_{ms} \end{bmatrix} = \begin{bmatrix} -T_{0s}/T_{rs} \\ -T_{1s}/T_{rs} \\ \vdots \\ 1/T_{rs} \\ \vdots \\ -T_{ms}/T_{rs} \end{bmatrix}. \tag{11}$$

Note that, by the transformations rules, $T_{0s}\leq 0$, $\frac{d_r^T}{\overline{T}_{rs}} = minlex\left\{\frac{d_i^T}{\overline{T}_{is}} : T_{is} > 0, 1\leq i\leq m\right\}$, $\overline{d}_r^T = \frac{1}{\overline{T}_{rs}} d_r^T$ e $\overline{d}_i^T = d_i^T - \frac{T_{is}}{\overline{T}_{rs}} d_r^T, i\neq r$. Additionally, note that column $\left[\overline{T}_{ir}\right]_{0\leq i\leq m}$ corresponds to the basic variable s and that column $\left[\overline{T}_{is}\right]_{0\leq i\leq m}$ corresponds to the non-basic variable r.

Consider, now, the linear program (5) as a minimization problem. Note that tableau $\overline{T}$ is not optimal for this program, because $\overline{T}_{0s} = -T_{0s}/T_{rs} \geq 0$. Thus, the non-basic variable r, corresponding to the column $\left[\overline{T}_{is}\right]_{0\leq i\leq m}$, can enter the basis, eventually reducing the value of the objective function.

On the other hand, once the variable that enters the basis is defined, the variable that leaves the basis must be determined by the lexicographic pivot rules; i.e., the variable that leaves the basis corresponds to the one given by $minlex\left\{\frac{\overline{d}_i^T}{\overline{T}_{is}} : \overline{T}_{is} > 0, 1\leq i\leq m\right\}$.

Thus, for $i\neq r$ and $\overline{T}_{is} = -T_{is}/T_{rs} > 0$, $\frac{\overline{d}_i^T}{\overline{T}_{is}} = \frac{1}{-T_{is}/T_{rs}}\left(d_i^T - \frac{T_{is}}{T_{rs}} d_r^T\right)$, we have $\frac{\overline{d}_i^T}{\overline{T}_{is}} = -\frac{T_{rs}}{T_{is}} d_i^T + d_r^T$. Then, since $-\frac{T_{rs}}{T_{is}} > 0$, results that $\frac{\overline{d}_i^T}{\overline{T}_{is}} \succ d_r^T$. On the other hand, since $\frac{\overline{d}_r^T}{\overline{T}_{rs}} = \frac{\frac{1}{T_{rs}} d_r^T}{1/T_{rs}} = d_r^T$, we have $\frac{\overline{d}_r^T}{\overline{T}_{rs}} = minlex\left\{\frac{\overline{d}_i^T}{\overline{T}_{is}} : \overline{T}_{is} > 0, 1\leq i\leq m\right\}$.

Therefore, the variable that leaves the basis is variable r and the new basis corresponds to tableau T. □

Remark 2. *Proposition 2 establishes that if vertex $v_{\overline{T}}$, (corresponding to tableau $\overline{T}$, in the maximization process) is obtained from the vertex v_T (corresponding to the table T), by changing the non-basic variable s for the basic variable r, then, reciprocally, the vertex v_T can be obtained from the vertex $v_{\overline{T}}$, changing the maximization by minimization, considering (in table $\overline{T}$) as non-basic variable that enters the variable r and using the lexicographic pivot rules. In this sense, Proposition 2 establishes that a vertex, different from the initial one, belongs to the graph if, and only if, it has a predecessor that belongs to the graph.*

5 PROPOSED ALGORITHM FOR VERTEX ENUMERATION

The proposed algorithm for vertex enumeration of a polyhedron $P = \{x \in \Re^n : Ax \leq b, x \geq 0\}$ is described in this section. It is considered the linear program (5) and it is assumed to be known an initial vertex $x = \begin{pmatrix} x_{B_1} \\ x_{N_1} \end{pmatrix} = \begin{pmatrix} B_{11}^{-1} b_{B_1} \\ 0 \end{pmatrix}$, where $\begin{bmatrix} B_{11} & 0 \\ B_{21} & I_{B_2} \end{bmatrix}$ is an associated basis of matrix $\begin{bmatrix} A & I \end{bmatrix}$ and $\begin{pmatrix} x \\ s \end{pmatrix}$, with $s = \begin{pmatrix} s_{N_2} \\ s_{B_2} \end{pmatrix} = \begin{pmatrix} 0 \\ b_{B_2} - B_{21} B_{11}^{-1} b_{B_1} \end{pmatrix}$, is a feasible solution of (5). In the following, matrix T is given by (6).

Algorithm 1. Algorithm for vertex enumeration of a polyhedron *P*.

Step 0. Set

$k := 1$ (*Current number of found vertices*); $x_k := x$ (*Current vertex*); *Vertices* $:= \{x_k\}$ (*Current set of found vertices*);

$\mathscr{B} := B_1 \cup B_2 := \{$*Initial basic indexes associated to variables x an s, respectively*$\}$;

$\aleph := N_1 \cup N_2 := \{$*Initial* n*on-basic indexes associated to variables x an s, respectively*$\}$;

$T :=$ *Tableau associated to vertex* x_k*, given by (6);*

$\widehat{B}_1 := B_1$, $\widehat{B}_2 := B_2$, $\widehat{\mathscr{B}} := \widehat{B}_1 \cup \widehat{B}_2$ (*Current basic indexes*); $\widehat{\widehat{\aleph}} := \aleph$; *Bases* $:= \{\widehat{\mathscr{B}}\}$ (Current set of found bases);

EntraBase $:= 0$; $\mathscr{M} := \emptyset$ (*list to control the found vertices which may possess undiscovered neighbours in the graph*); $\mathscr{M}_{EntraBase} = \emptyset$.

Step 1. (*Determination of the non-basic variable* p *that enters the basis)*

If $\widehat{\widehat{\aleph}} = \emptyset$, STOP. Otherwise, choose $p \in \widehat{\widehat{\aleph}}$ as the index that enters the basis. Set $k := k+1$ and $\widehat{\widehat{\aleph}} := \widehat{\widehat{\aleph}} \setminus \{p\}$. Continue.

Step 2. (*Determination of the basic variable* q *that leaves the basis - Determination of a new vertex or an extreme ray* as edge of P*)*

Apply the lexicographic pivot rules (PivLex2) to find the variable q that leaves the basis. If q cannot be found (because $T_{ip} \leq 0, \forall i \in \widehat{\mathscr{B}}$), go to Step 3 (an extreme ray was found as edge of P). Otherwise, set $\widehat{\mathscr{B}} := \widehat{\mathscr{B}} \cup \{p\} \setminus \{q\}$ *and actualize* $\widehat{B}_1$ *and* $\widehat{B}_2$*, set of basic indexes corresponding to x an s, respectively, so that* $\widehat{\mathscr{B}} := \widehat{B}_1 \cup \widehat{B}_2$. If basis $\widehat{\mathscr{B}} \in$ *Bases*, go to Step 3. Otherwise, set *Bases* $:=$ *Bases* $\cup \widehat{\mathscr{B}}$, compute tableau T and vertex y associated to basis $\widehat{\mathscr{B}}$.

If $y \notin$ *Vértices*, set $k := k+1$, $x_k := y$ and *Vertices* $:=$ *Vertices* $\cup \{x_k\}$. Generate $\widehat{\aleph}$ (non-basic indexes associated to basis $\widehat{\mathscr{B}}$) and $\widehat{\aleph}_- := \{j \in \widehat{\aleph} : \overline{T}_{0j} \leq 0\}$. If $\left|\widehat{\aleph}_-\right| \geq 2$, set *EntraBase* $:=$ *EntraBase* $+1$, $\mathscr{M}_{EntraBase} := \{\widehat{\mathscr{B}}, \widehat{\aleph}, \widehat{\aleph}_-\}$, $\mathscr{M} := \mathscr{M} \cup \mathscr{M}_{EntraBase}$ and continue.

Step 3. If $\mathscr{M} = \emptyset$, go to Step 4. Otherwise, for $\mathscr{M}_{EntraBase} := \{\widehat{\mathscr{B}}, \widehat{\mathfrak{K}}, \widehat{\mathfrak{K}}_-\}$, if $\widehat{\mathfrak{K}}_- = \emptyset$, set $\mathscr{M}_{EntraBase} := \emptyset$ and $EntraBase := EntraBase - 1$. Otherwise, choose $p \in \widehat{\mathfrak{K}}_-$, set $\widehat{\mathfrak{K}}_- := \widehat{\mathfrak{K}}_- \{p\}$ and $\mathscr{M}_{EntraBase} := \{\widehat{\mathscr{B}}, \widehat{\mathfrak{K}}, \widehat{\mathfrak{K}}_-\}$. Go to Step 2.

Step 4. If $k = 2$, find, if they exist, all the corresponding optimal vertices of the linear program (5) and save them in the set *Vertices*. Otherwise, go to Step 1.

Proposition 3. *Algorithm 1 is finite and find all the vertices of polyhedron P.*

Proof. Algorithm 1 determines for each non-basic variable that enters the basis, at Step 1, a sequence of feasible vertices that ends in a solution of (5), a vertex already found in a previous sequence of vertices or an extreme ray as an edge of P. Thus, since $\mathfrak{K} := N_1 \cup N_2$, the initial set of non-basic indexes, at Step 0, is finite, we have that Algorithm 1 is finite.

On the other hand, to prove that all the vertices of polyhedron P are found, suppose that there exists one vertex v_0 of polyhedron which was not found by Algorithm 1. Then, because of the Proposition 2 and the Remark 2, there exists a sequence of vertices initiating at v_0 and ending at the initial vertex, which is not found by the Algorithm 1. This is a contradiction with the fact that, from the initial vertex, it is possible only to exit through vertices that Algorithm 1 finds. Therefore, all the vertices of polyhedron P are found. □

Algorithm 1, as illustrated in Fig. 2, finds a set of paths from the initial vertex to a final vertex, running through all the vertices of the polyhedron P.

6 COMPLEXITY

Here it is discussed the complexity of Algorithm 1. It is considered a polyhedron as given by $P = \{x \in \Re^n : Ax \leq b, x \geq 0\}$, where $A \in \Re^{m \times n}$ and $b \in \Re^m$. It is not assumed that P is bounded. To begin, note that Algorithm 1 does not work with the ordinary simplex tableau, but a simplex tableau ordered by format T, given in (6). It is possible to get format T, from an ordinary simplex tableau, in $O\left(m^2\right)$ complexity time. Later tables with format T are obtained, after lexicographic pivoting, with two simple attributions.

It is convenient to begin by estimating the complexicity associated to the set *Bases*, the bases found by the algorithm. This set is composed by not repited m-vectors, which requires saving them as ordered and compared m-vectors. In that sense, it is only necessary to sort the first basis, because the following ones are obtained by changing one single index (they are, basically, sorted). Thus, the comparison of two basis is obtained in $O(m)$. Since, the cardinality of the *Bases* is $|Bases|$, the total number of comparisons is obtained in $O(m|Bases|^2)$.

To compute the complexity associated with the other steps of Algorithm 1, note that it consists of the (main) loop, starting at Step 1 and ending at Step 4 (Step 1-Step 4 loop), in which all

the vertices of the polyhedron P are found. At Step 2, lexicographic pivots are performed to compute tableau $\overline{T}$ from tableau T, under the following considerations:

(i) Determination of index $p \in N$, which enters the basis. Since $|N| = n$, the complexity of the determination of all these indices is $O(n)$.

(ii) Given $p \in N$, it is defined index $q \in \mathscr{B}$, which leaves the basis, according to (9), $\frac{d_q^T}{T_{qp}} = minlex\left\{\frac{d_i^T}{T_{ip}} : T_{ip} > 0, 1 \leq i \leq m\right\}$. In the worst case, when all $T_{ip} > 0$, it should be computed all the vectors $\frac{d_i^T}{T_{ip}}$, to find the lexicographic minimum. In this case m comparisons ($T_{ip} > 0$ are necessary, each one involving the computation of the vector $\frac{d_i^T}{T_{ip}}$, which involves $(n+1)$ operations. Thus, this computation involves $m(n+1)$ operations. Additionally, since the final computation of $minlex$ could involve, in the worst case, the lexicographic comparison of pairs of vectors in the set $\left\{\frac{d_i^T}{T_{ip}}\right\}_{i=1}^{m}$, up to $(m-1)$ times, and each comparison can take $2n$ operations, that final computation can demand up to $2n(m-1)$ operations. Thus, the maximum number of operations to perform a lexicographic pivoting results:

$$\underbrace{m(n+1)}_{computation\ of\ d_j^T/T_{jp}, 1 \leq j \leq m} + \underbrace{2n(m-1)}_{comparisons} \in O(mn).$$

This way, each time a lexicographic pivoting is performed, it takes a computational complexity $O(mn)$.

The Step 1-Step 4 loop, in addition to the lexicographic pivot carried out in Step 2, involves, on the one hand, assignments and updates of sets with changes of two indices and, on the other hand, comparisons of sets. Note that comparisons are dominant, in terms of the number of operations, in relation to assignments and updates. So that, the complexity associated with these operations is determined by the comparisons made in Step2: "If basis $\overline{\mathscr{B}} \in Bases$, go to Step 3", "If $x_k \notin V\acute{e}rtices$, set $Vertices := Vertices \cup \{x_k\}$"

Thus, denoting the set of vertices of polyhedron P by $V(P)$, we have $|V(P)| \leq |Bases| \leq \binom{m+n}{m}$, where $Bases$, defined at Step 0 of Algorithm 1, is the set of feasible basis of linear program (5) found by the algorithm. Note that $|V(P)|$ may be much less than $|Bases|$ and $|Bases|$ much less than $\binom{m+n}{m}$ (in the non-degenarate case $|V(P)| = |Bases|$).

Therefore, the number of bases found by Step 1-Step 4 loop is at most a multiple of $|Bases|$ and each basis found involves a lexicographic pivoting, totaling around $mn|Bases| \in O\left(mn\binom{m+n}{m}\right)$ operations, where $|Bases| = |V(P)|$ for the non-degenerate case.

Finally, the number of operations associated with Algorithm 1 corresponds to the number of operations associated with lexicographic pivotings plus the complexity associated to the set *Bases* ($m|Bases|^2$). This way, the number of operations associated with Algorithm 1 can be estimated by

$$mn\,|Bases| + m|Bases|^2 \leq mn\binom{m+n}{m} + m\binom{m+n}{m}^2.$$

Since $\binom{m+n}{m} = \frac{(m+n)!}{m!n!} \in O((m+n))$, then

$$mn\binom{m+n}{m} + m\binom{m+n}{m}^2 \in O\left(m\,(m+n)^{2\times min\{m,n\}}\right).$$

This way, the computational complexity associated with Algorithm 1 for a polyhedron P is $O\left(m\,(m+n)^{2\times min\{m,n\}}\right)$.

Note, on the other hand, that for the case of non-degenerate polytope P, the number of comparisons to define the set *Bases* is $O(|V(P)|)$, because if the Algorithm 1 generates L different directed paths initiating at initial vertex x_1, each one of cardinality $K_i, i = 1,\ldots L$, then $\sum_{i=1}^{L} K_L \cong |V(P)|$; thus, for the first sequence, no comparisons are needed; and, for each one of the other sequences are needed $|V(P)| - (K_1 + \ldots + K_{i-1})\,, i = 2,\ldots,L$, comparisons. Therefore, the total number of comparisons is $\sum_{i=1}^{L-1}\left(|V(P)| - \left(\sum_{j=1}^{i} K_j\right)\right) \in O(|V(P)|)$ and, since each vector is ordered in m comparisons, *Bases* is generated in $O\,(m\,|V(P)|)$. Thus, in the case of non-degenerate polytope P, the computational complexity associated with Algorithm 1 is $O\,(mn\,|V(P)|)$.

7 NUMERICAL EXAMPLES

As illustration, in this section, Algorithm 1 is applied to two polyhedra, $P0 \subset \Re^3$ and $P00 \subset \Re^3$ (adapted from Campêlo and Scheimberg, 2005). The original example constraints are given by the first eight inequalities of relation (12). $P0$ is given by relations (12) and $P00$ is given by relations (12) without the ninth constraint ($x_1 - 2x_2 + 20x_3 \leq 8$). Figure 1 illustrates the polyhedra $P0$ and $P00$. Figure 2 shows the sequence of vertices found by the algorithm for $P0$, starting from the known vertex $I = (1,1,0.45)$.

$$
\begin{array}{c}
x_1+x_2+x_3\leq 3 \quad \leftarrow BCGF \\
x_1+x_2-x_3\geq 1 \quad \leftarrow ADHK \\
-x_1+x_2+x_3\leq 1 \quad \leftarrow ABFK \\
x_1-x_2+x_3\leq 1 \quad \leftarrow DCGH \\
16x_1-6x_2+60x_3\leq 37 \quad \leftarrow FIG \\
6x_1-16x_2+60x_3\leq 17 \quad \leftarrow KIH \\
6x_1-6x_2+60x_3\leq 27 \quad \leftarrow KIF \\
16x_1-16x_2+60x_3\leq 27 \quad \leftarrow GIH \\
x_1-2x_2+20x_3\leq 8 \quad \leftarrow EFIJ \\
x_1\geq 0, x_2\geq 0, x_3\geq 0
\end{array}
\tag{12}
$$

To apply Algorithm 1 to $P0$, consider (12) as a set of equalities with nonnegative variables (adding the respective slack variables). This way, $P0$ is given by a set of 9 equations and 12 nonnegative variables, where the first 3 variables are x_1, x_2 and x_3 and the last 9 ones are the slack variables associated to the respective inequalities ($s_1,\ldots,s_9$), corresponding to $x_4,\ldots,x_{12}$.

To illustrate how Algorithm 1 travels the vertices of the polyhedron, from vertex I, Figure 3 shows non-basic variables, the vertices of $P0$ which corresponds to the respective bases and the value of the objective function of linear program (5) at the considered vertex. For example, I^0 :

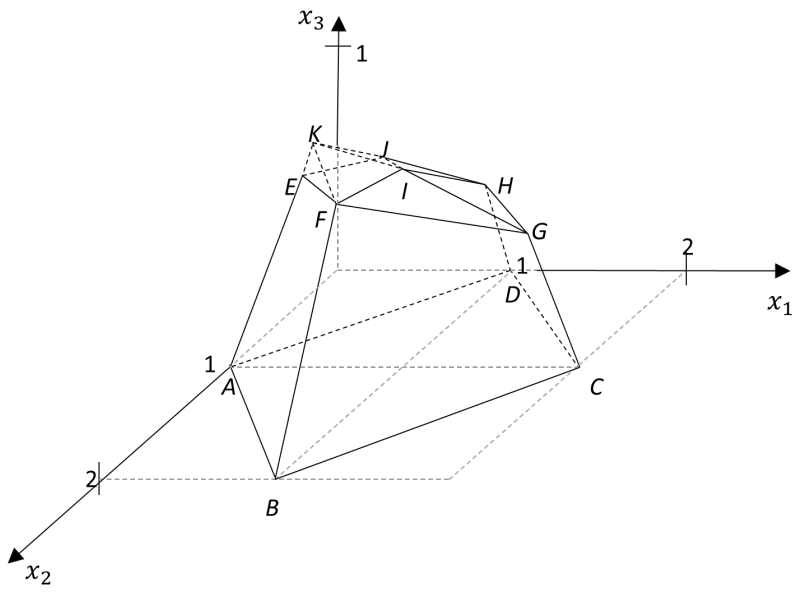

Figure 1 – Illustration of polyhedra $P0$ and $P00$, defined from (12) ($P00$ includes the pyramid $EFIJK$, but $P0$ does not).

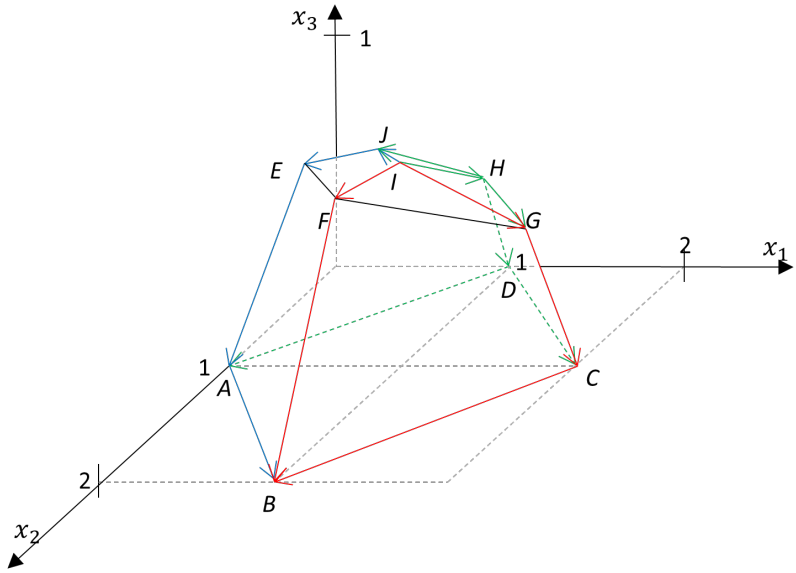

Figure 2 – Illustration of polyhedron $P0$, showing the known initial vertex I and the sequence of vertices found by the Algorithm 1 (in order: red, blue, green).

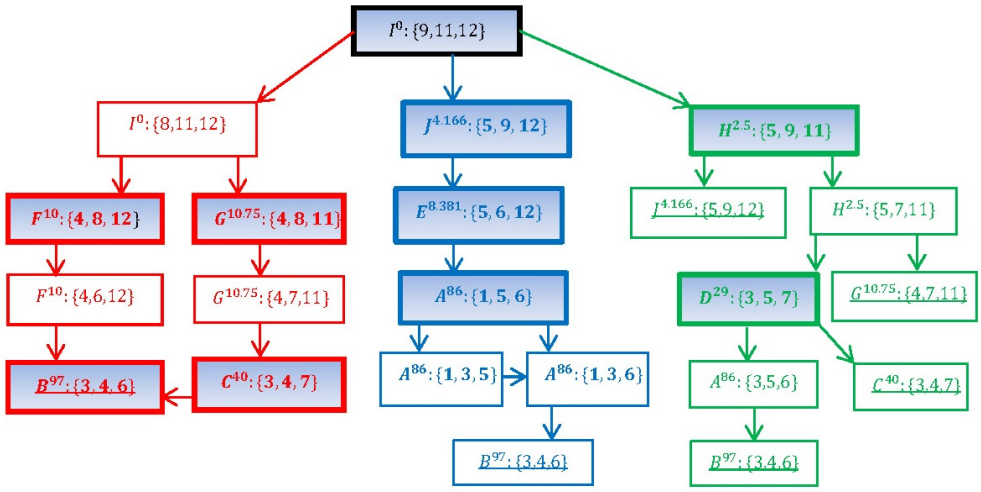

Figure 3 – The sequences of vertices found by Algorithm 1 are shown, starting at vertex I. Red, blue and green show the sequences generated when the non-basic variables 9, 11 and 12 enter the basis, respectively. The new vertices found in each sequence are shown in bold. The underlined correspond to the final vertices of the respective sequences (because they are a maximum of (5) or correspond to bases already found).

$\{9,11,12\}$ means that, in the non-basic variables $\{9,11,12\}$ (which define the basic variables $\{1,2,3,4,5,6,7,8,10\}$), the corresponding vertex is I and that $z = \sum_{i\in N_1} x_i + \sum_{i\in N_2} s_i = 0$. From now on, $I^0 : \{9,11,12\}$ and the others analogous notation will be called, indistinctly, as the defined vertex I and the respective defined vertices.

From the non-basic variables, the lexicographic simplex algorithm is applied to perform basis changes, changing the vertex (or not, if it is degenerate).

Figure 3 shows the mapping performed by the algorithm. The first vertex is the known vertex I ($I^0 : \{9,11,12\}$). Since any one of the 3 non-basic variables can enter the basis, there are 3 different paths: the first one, when variable 9 enters the basis (shown in red in Figure 3); the second one, when variable 11 enters the basis (shown in blue in Figure 3); and the third one, when variable 12 enters the basis (shown in green in Figure 3). A path eventually complements the vertices found by the other paths, as shown in Figure 3.

Algorithm 1 identifies the changes of basis, performing the respective pivotings. Note that the top rectangle in Figure 3 corresponds to $I^0 : \{9,11,12\}$, considering that non-basic variable 9 enter the basis, the red path is initiated, going sequentially to vertex $I^0 : \{8,11,12\}$, $F^{10} : \{4,6,12\}$, $F^{10} : \{4,8,12\}$ and $B^{97} : \{3,4,6\}$ (the optimal point of linear program (5)). The blue and green paths are generated when, from the initial vertex $I^0 : \{9,11,12\}$, enters the basis the non-basic variables 11 and 12, respectively.

The found vertices, in the order they were found, are: $I = (1,1,0.45)$, $F = (1,1.5,0.5)$, $B = (1,2,0)$, $G = (1.75,1,0.25)$, $C = (2,1,0)$, $J = (0.5833,0.8750,0,4583)$, $E = (0.4762,1,0.4762)$, $A = (0,1,0)$, $H = (1,0.25,0.25)$, $D = (1,0,0)$ (results are shown at Table 1, corresponding to Problem *P0*).

As already mentioned, eliminating the ninth constraint at (12), ($x_1 - 2x_2 + 20x_3 \leq 8$), it is obtained the polyhedron *P*00 (see Figure 1). The application of Algorithm 1 to polyhedron *P*00 generates the sequence of vertices: $I = (1,1,0.45)$, $K = (0.5,1,0.5)$, $A = (0,1,0)$, $H = (1,0.25,0.25)$, $D = (1,0,0)$, $F = (1,1.5,0.5)$, $G = (1.75,1,0.25)$, $C = (2,1,0)$, $B = (1,2,0)$ (results are shown in Table 1, corresponding to Problem *P00*).

8 COMPUTATIONAL TESTS

Algorithm 1 was coded in Matlab and run for different examples on a desktop with an i5-2400 CPU@3.10GHz processor, 4.00GB RAM (using a single core). Table 1 shows the results corresponding to computational tests run with 55 polyhedra of different sizes, where the number of constraints, m, varies between 8 and 80 and the number of variables, n, between 3 and 30. Except for the first two polyhedra, *P*0 and *P*00, which correspond to polyhedron (12) and a modification of it, respectively (see Section 7), for the other 53 polyhedra the coefficient matrices, $A \in \Re^{m\times n}$, and the right side vectors, $b \in \Re^m$, were randomly generated, with integer coefficients between 1 and 100, in such a way that $0 \in \Re^n$ is always a vertex of the generated polyhedron. In all those tests, the density of matrix A is near 1. As expected, Table 1 shows that the number of vertices increases as m and n increase.

Table 1 – Results of computational tests corresponding to randomly generated polyhedra: m, number of linear inequalities; n, number of non-negative variables; density of matrix $A \cong 1$.

Problem and size *Problem*: (m, n)	Run time (seconds)	$\|Bases\|$	$\|Vertices\|$	Problem and size *Problem*: (m, n)	Run time (seconds)	$\|Bases\|$	$\|Vertices\|$
P0: (9,3) *	0.161	20	10	P27: (55,20)	892.589	872	860
P00: (8,3) *	0.144	20	9	P28: (55,20)	241.063	444	430
P1: (10,3)	0.175	24	14	P29: (55,20)	2279.671	1410	1410
P2: (20,3)	0.126	14	14	P30: (60,10)	1.426	35	35
P3: (20,3)	0.077	5	5	P31: (60,10)	0.716	20	20
P4: (30,3)	0.081	4	4	P32: (60,15)	0.778	16	16
P5: (40,3)	0.102	6	6	P33: (60,15)	25.085	156	156
P6: (50,3)	0.087	4	4	P34: (60,15)	4.869	60	60
P7: (8,5)	0.261	24	24	P35: (60,15)	33.989	186	186
P8: (8,5)	0.355	30	30	P36: (60,20)	1.522	21	21
P9: (10,6)	0.395	28	28	P37: (60,20)	8.773	72	72
P10: (10,6)	0.177	16	16	P38: (60,20)	1.510	21	21
P11: (20,6)	0.146	13	13	P39: (60,20)	1.533	21	21
P12: (50,10)	1.283	36	36	P40: (70, 10)	3.227	56	56
P13: (50,10)	0.597	20	20	P41: (70, 10)	1.712	36	36
P14: (50,10)	25.216	200	200	P42: (70, 15)	24.622	134	134
P15: (50,10)	1.772	45	45	P43: (70, 15)	3.303	42	42
P16: (50,15)	163.308	416	416	P44: (70, 20)	244.404	420	420
P17: (50,15)	136.704	390	390	P45: (70, 20)	6394.456	2286	2286
P18: (50,15)	76.230	290	290	P46: (70, 20)	19.378	108	108
P19: (55,15)	52.234	234	234	P47: (70, 30)	15000.444	2724	2724
P20: (55,20)	782.577	792	792	P48: (70, 30)	70364.359**	6842	6482
P21: (55,20)	10.058	76	76	P49: (70, 30)	137.446	168	168
P22: (55,20)	411.810	596	592	P50: (70, 30)	4404.743	1272	1272
P23: (50,20)	1457.964	1087	1087	P51: (80,30)	52692.922**	5993	5993
P24: (50,20)	761.823	800	800	P52: (80,30)	3786.238	1220	1220
P25: (50,20)	1686.378	1195	1195	P53: (80,30)	371.640	310	310
P26: (50,20)	284.356	483	483				

(*) Non-random polyhedra, described at the end of Section 7. (**) Computational process interrupted at the indicated time.

Table 2 shows the results corresponding to computational tests run with 45 polyhedra of equal sizes ($m = 50, n = 10$), randomly generated, where the density (δ) of matrix A varies between 0 and 1. The data corresponds to 5 polyhedra for each one of the indicated density. Note that, except for $\delta = 1$, the mean of bases do not appear to be correlated with the density of the matrix A. For all these cases, the number of bases found ranges between 216 and 736, for $\delta = 0.40$ and $\delta = 0.70$, respectively, with a large standard deviation.

The authors tested some popular algorithms to generate a polyhedron from a set of vertices, in different dimensions. Unfortunately, rounding errors and/or processing time did not allow using them for the tests in Table 1. Anyway, all the generated vertices of Table 1 and Table 2 were tested to be vertices of the respective given polyhedron, resulting true.

Table 2 – Means and standard deviations (std) results of computational tests corresponding to randomly generated polyhedra: $m = 50$, number of linear inequalities; $n = 10$, number of non-negative variables; δ, density of matrix A.

Size problem and density (m, n), δ	Mean / std Run time (seconds)	Mean / std $\lvert Bases \rvert$	Mean / std $\lvert Vertices \rvert$
(50,10), 1	19.586 / 29.597	90 / 89.922	88 / 91.712
(50,10), 0.90	285.645 / 392.269	369.800 / 293.841	369.800 / 293.841
(50,10), 0.80	199.153 / 144.075	344.400 / 191.187	344.400 / 191.187
(50,10), 0.70	1046.700 / 1403.300	736.200 / 575.057	736.200 / 575.057
(50,10), 0.60	114.511 / 142.723	246 / 187.717	246 / 187.717
(50,10), 0.50	515.670 / 626.282	494.400 / 432.004	494.400 / 432.004
(50,10), 0.40	86.925 / 141.656	216.800 / 199.799	216.800 / 199.799
(50,10), 0.30	678.527 / 629.801	629.400 / 337.804	629.400 / 337.804
(50,10), 0.25	149.765 / 133.060	316 / 147.838	175.6 / 167.490

9 CONCLUSIONS

In this paper it is introduced an algorithm to enumeration of the vertices of a polyhedron $P = \{x \in \Re^n : Ax \leq b, x \geq 0\}$, with $A \in \Re^{m \times n}$ and $b \in \Re^m$. The proposed algorithm is based on pivoting and works in any polyhedron, but the polynomial time complexity with respect to the number of vertices is only guaranteed for polytopes. The time complexity for the case of a simple polyhedron is $O(mn|V(P)|)$, where $|V(P)|$ is the number of vertices, and for the case of a general polyhedron the time complexity is $O\left(m(m+n)^{2 \times min\{m,n\}}\right)$. The proposed algorithm was coded in Matlab and tested for several different size random examples (see Table 1), showing good performance.

Acknowledgements

The authors thank anonymous reviewers for their valuable comments and for many corrections.

References

AVIS D, BREMNER D & SEIDEL, R. 1997. How good are convex hull algorithms? *Computational Geometry: Theory and Applications*, **7**: 265–301.

AVIS D & FUKUDA, K. 1992. A Pivoting Algorithm for Convex Hulls and Vertex Enumeration of Arrangements and Polyhedra. *Discrete & Computational Geometry*, **8**: 295–313.

AVIS D & JORDAN C. 2018. mplrs: A scalable parallel vertex/facet enumeration code. *Mathematical Programming Computation*, **10**(2): 267–302.

BLAND, R.G. 1977. New finite pivoting rules for the simplex method. *Mathematics of Operations Research*, **2**: 103–107.

BLUM E, OETTLI, W. 1975. Mathematische Optimierung. Grundlagen und Verfahren. *Okonometrie und Unternehmensforschung*, 20. Springer-Verlag, Berlin-Heidelberg-New York.

CAMPÊLO M & SCHEIMBERG, S. 2005. A Simplex Approach for Finding Local Solutions of a Linear Bilevel Program by Equilibrium Points. *Annals of Operations Research*, **138**(1): 143–157.

CHAND DR & KAPUR SS. 1970. An Algorithm for Convex Polytopes. *Journal of the Association for Computing Machinery*, **17**(1): 78–86.

DANTZIG GB & THAPA MN. 1997. *Linear Programming 1: Introduction.* Springer-Verlag, Berlin.

DYER ME. 1983. The Complexity of Vertex Enumeration Methods. *Mathematics of Operations Research*, **8**(3): 381-402.

ENCYCLOPÆDIA BRITANNICA. 2008. *Euclid (Greek Mathematician)*. https://www.britannica.com/biography/Euclid-Greek-mathematician (consulted on July 31, 2008).

LEWIS C. 2008. *Linear Programming: Theory and Applications*. Whitman College Mathematics Department.

MATHEISS TH., RUBIN, D. 1980. A Survey and Comparison of Methods for Finding All Vertices of Convex Polyhedral Sets. *Mathematics of Operations Research*, **5**(2): 167–185.

MINOUX M. 1986. *Mathematical Programming: Theory and Algorithms*. New York: John Wiley.

MOTZKIN TS.; RAIFFA H, THOMPSON, G. L.; THRALL, R. M. 1953. The double description method. *In*: H. W. Kuhn and A. W. Tacker (eds). *Contributions to the Theory of Games*, Vol. If. Princeton University Press, Princeton, N.J., p. 51–73.

NAJT L. 2021. *Sampling the vertices of a polytope is still hard even when the branch-width is bounded*. https://lorenzonajt.github.io/Documents/Polytope\Paper\Draft.pdf (consulted on December 3, 2021).

PROVAN JS. 1994. Efficient enumeration of the vertices of polyhedra associated with network LP' s. *Mathematical Programming*, **63**: 47–64.

REIMERS A & STOUGIE L. 2016. *Polynomial time vertex enumeration of convex polytopes of bounded branch-width*. arXiv preprint, arXiv:1404.5584v2.

TERLAKY T. 2001. Lexicographic Pivoting Rules. *In*: Floudas C.A., Pardalos P.M. (eds) *Encyclopedia of Optimization*. Springer, Boston, MA.

YANG Y. 2021. https://arxiv.org/abs/1909.11843v2 arXiv:1909.11843v2 [math.OC].

How to cite

ASSAD CL, MORALES G & ARICA J. 2022. Vertex enumeration of polyhedra. *Pesquisa Operacional*, **42**: e254570. doi: 10.1590/0101-7438.2022.042.00254570.