Computing the Integer Points of a Polyhedron (Extended Abstract)

Rui-Juan Jing[1,2] and Marc Moreno Maza[2]

[1] KLMM, UCAS, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, rjing8@uwo.ca,
[2] University of Western Ontario, moreno@csd.uwo.ca.

Abstract. Let K be a polyhedron in $\mathbb{R}^d$, given by a system of m linear inequalities, with rational number coefficients bounded over in absolute value by L. We propose an algorithm for computing an irredundant representation of the integer points of K, in terms of "simpler" polyhedra, each of them having at least one integer point. Using the terminology of W. Pugh: for any such polyhedron P, no integer point of its grey shadow extends to an integer point of P. We show that, under mild assumptions, our algorithm runs in exponential time w.r.t. d and in polynomial w.r.t m and L. We report on a software experimentation.

1 Introduction

The integer points of polyhedral sets are of interest in many areas of mathematical sciences, see for instance the landmark textbooks of A. Schrijver [19] and A. Barvinok [3], as well as the compilation of articles [4]. One of these areas is the analysis and transformation of computer programs. For instance, integer programming [7] is used by P. Feautrier in the scheduling of for-loop nests [8] Barvinok's algorithm [2] for counting integer points in polyhedra is adapted by M. Köppe and S. Verdoolaege in [16] to answer questions like how many memory locations are touched by a for-loop nest. In [17], W. Pugh proposes an algorithm, called the *Omega Test*, for testing whether a polyhedron has integer points. In the same paper, W. Pugh shows how to use the Omega Test for performing dependence analysis [17] in for-loop nests. Then, in [18], he uses the Omega Test for deciding Presburger arithmetic formulas.

In [18], W. Pugh also suggests, without stating a formal algorithm, that the Omega Test could be used for quantifier elimination on Presburger formulas. This observation is a first motivation for the work presented in this paper: we adapt the Omega Test so as to describe the integer points of a polyhedron via a *projection* scheme, thus performing elimination of existential quantifiers on Presburger formulas.

Projections of polyhedra and parametric programming are tightly related problems, see [12]. Since the latter is essential to the parallelization of for-loop nests [7], which is of interest to the authors [5], we had here a second motivation for developing the proposed algorithm.

In [9], M. J. Fischer and M. O. Rabin show that any algorithm for deciding Presburger arithmetic formulas has a worst case running time which is a doubly exponential in the length of the input formula. However, this worst case scenario is based on a formula alternating existential and universal quantifiers. Meanwhile, in practice, the original *Omega Test* (for testing whether a polyhedron has integer points) can solve "difficult problems" as shown by W. Pugh in [18] and others, e.g. D. Wonnacott in [20]. This observation brings our third motivation: determining realistic assumptions under which our algorithm, based on the Omega Test, could run in a single exponential time.

Our algorithm takes as input a system of linear inequalities $\mathbf{Ax} \leq \mathbf{b}$ where $\mathbf{A}$ is a matrix over $\mathbb{Z}$ with m rows and d columns, $\mathbf{x}$ is the unknown vector and $\mathbf{b}$ is a vector of m coefficients in $\mathbb{Z}$. The points $\mathbf{x} \in \mathbb{R}^d$ satisfying $\mathbf{Ax} \leq \mathbf{b}$ form a polyhedron K and our algorithm decomposes its integer points (that is, $K \cap \mathbb{Z}^d$) into a disjoint union $\left(K_1 \cap \mathbb{Z}^d\right) \uplus \cdots \uplus \left(K_e \cap \mathbb{Z}^d\right)$, where $K_1, \ldots, K_e$ are "simpler" polyhedra such that $K_i \cap \mathbb{Z}^d \neq \varnothing$ holds, for $1 \leq i \leq e$. To use the terminology introduced by W. Pugh for the Omega test, for $1 \leq i \leq e$, no integer point of the grey shadow of the polyhedron K_i extends to an integer point of K_i. As a consequence, applying our algorithm to K_i would return K_i itself, for $1 \leq i \leq e$. Let us present the key principles and features of our algorithm through an example. Consider the polyhedron K of $\mathbb{R}^4$ given below:

$$\begin{cases} 2x + 3y - 4z + 3w \leq 1 \\ -2x - 3y + 4z - 3w \leq -1 \\ -13x - 18y + 24z - 20w \leq -1 \\ -26x - 40y + 54z - 39w \leq 0 \\ -24x - 38y + 49z - 31w \leq 5 \\ 54x + 81y - 109z + 81w \leq 2 \end{cases}.$$

A first procedure, called IntegerNormalize, detect implicit equations and solve them using techniques based on *Hermite normal form*, see Sect. 3. In our example $2x+3y-4z+3w = 1$ is an implicit equation and IntegerNormalize($\mathbf{Ax} \leq \mathbf{b}$) returns a triple $(\mathbf{t}, \mathbf{x} = \mathbf{Pt}+\mathbf{q}, \mathbf{Mt} \leq \mathbf{v})$ where $\mathbf{t}$ is a new unknown vector, the linear system $\mathbf{x} = \mathbf{Pt}+\mathbf{q}$ gives the general form of an integer solution of the implicit equation(s) and $\mathbf{Mt} \leq \mathbf{v}$ is obtained by substituting $\mathbf{x} = \mathbf{Pt} + \mathbf{q}$ into $\mathbf{Ax} \leq \mathbf{b}$. In our example, the systems $\mathbf{x} = \mathbf{Pt} + \mathbf{q}$ and $\mathbf{Mt} \leq \mathbf{v}$ are given by:

$$\begin{cases} x = -3t_1 + 2t_2 - 3t_3 + 2 \\ y = 2t_1 + t_3 - 1 \\ z = t_2 \\ w = t_3 \end{cases} \quad \text{and} \quad \begin{cases} 3t_1 - 2t_2 + t_3 \leq 7 \\ -2t_1 + 2t_2 - t_3 \leq 12 \\ -4t_1 + t_2 + 3t_3 \leq 15 \\ -t_2 \leq -25 \end{cases}.$$

A second procedure, called DarkShadow, takes $\mathbf{Mt} \leq \mathbf{v}$ as input and returns a couple $(\mathbf{t}', \Theta)$ where $\mathbf{t}'$ stands for all $\mathbf{t}$-variables except t_1, and Θ is a linear system in the $\mathbf{t}'$-variables such that any integer point solving of Θ extends to an integer point solving $\mathbf{Mt} \leq \mathbf{v}$. In our example, $\mathbf{t}' = \{t_2, t_3\}$ and Θ is given by:

$$\begin{cases} 2t_2 - t_3 \leq 48 \\ -5t_2 + 13t_3 \leq 67 \\ -t_2 \leq -25 \end{cases}$$

The polyhedron D of $\mathbb{R}^2$ defined by Θ, and the inequalities of $\mathbf{Mt} \leq \mathbf{v}$ not involving t_1, is called the *dark shadow* of the polyhedron defined by $\mathbf{Mt} \leq \mathbf{v}$. On

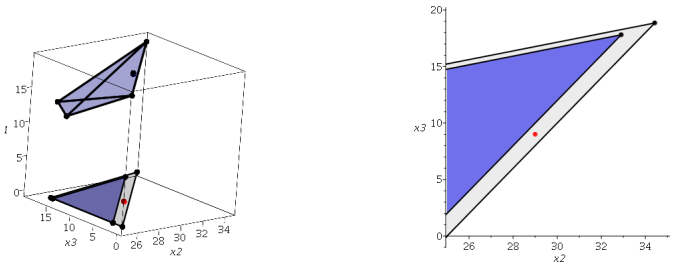

Fig. 1: The real, the dark and the grey shadows of a polyhedron.

the left-hand side of Fig. 1, one can see the polyhedron defined in $\mathbb{R}^3$ by $\mathbf{Mt} \leq \mathbf{v}$ together with its dark shadow D (shown in dark blue) as well as its projection on the (t_2, t_3)-plane, denoted by R and called *real shadow* by W. Pugh. The right-hand side of Fig. 1 gives a planar view of D and R. It turns out that, if $\mathbf{M't'} \leq \mathbf{v'}$ is the linear system generated by applying *Fourier-Motzkin elimination* (without removing redundant inequalities) to $\mathbf{Mt} \leq \mathbf{v}$ (in order to eliminate t_1), then Θ is given by a linear system of the form $\mathbf{M't'} \leq \mathbf{w'}$. This explains why, on the right-hand side of Fig. 1, each facet of the dark shadow D is parallel to a facet of the real shadow R. While this property is observed on almost all practical problems, in particular in the area of analysis and transformation of computer programs, it is possible to build examples where this property does not hold.

On the right-hand side of Fig. 1, one observes that the region $R \setminus D$, called *grey shadow*, contains integer points. Some of them, like $(t_2, t_3) = (29, 9)$, do not extend to an integer solution of $\mathbf{Mt} \leq \mathbf{v}$. Indeed, plugging $(t_2, t_3) = (29, 9)$ into $\mathbf{Mt} \leq \mathbf{v}$ yields $\frac{37}{2} \leq t_1 \leq \frac{56}{3}$, which has no integer solutions. However, other integer points of $R \setminus D$ may extend to integer solutions of $\mathbf{Mt} \leq \mathbf{v}$. In order to determine them, a third procedure, called GreyShadow, is needed. We give a disjoint decomposition of grey shadow by negating each inequality θ of Θ in turn. Then, compute the integer points in each disjoint part. Details are given in Sect. 4.

Returning to our example, the negation of the inequality $2t_2 - t_3 \leq 48$ from Θ, combined with the system $\mathbf{Mt} \leq \mathbf{v}$, yields the following

$$\begin{cases} -2t_1 + 2t_2 - t_3 = 12 \\ 3t_1 - 2t_2 + t_3 \leq 7 \\ -4t_1 + t_2 + 3t_3 \leq 15 \\ -t_2 \leq -25 \\ -2t_2 + t_3 \leq -49 \end{cases},$$

which, by means of IntegerNormalize, rewrites to:

$$\begin{cases} t_1 = t_4 \\ t_2 = t_5 + 1 \\ t_3 = -2t_4 + 2t_5 + 1 \end{cases}, \text{ and } \begin{cases} t_4 \leq 8 \\ -10t_4 + 7t_5 \leq 11 \\ -t_5 \leq -24 \\ -2t_4 - t_5 \leq -48 \end{cases},$$

where t_4, t_5 are new variables. Continuing in this manner with the GreyShadow procedure, a decomposition of the integer points of $\mathbf{Mt} \leq \mathbf{v}$ is given by:

$$\begin{cases} 3t_1 - 2t_2 + t_3 \leq 7 \\ -2t_1 + 2t_2 - t_3 \leq 12 \\ -4t_1 + t_2 + 3t_3 \leq 15 \\ 2t_2 - t_3 \leq 48 \\ -5t_2 + 13t_3 \leq 67 \\ -t_2 \leq -25 \\ 2 \leq t_3 \leq 17 \end{cases}, \begin{cases} t_1 = 15 \\ t_2 = 27 \\ t_3 = 16 \end{cases}, \begin{cases} t_1 = 18 \\ t_2 = 33 \\ t_3 = 18 \end{cases}, \begin{cases} t_1 = 14 \\ t_2 = 25 \\ t_3 = 15 \end{cases}, \begin{cases} t_1 = 19 \\ t_2 = 50 + t_6 \\ t_3 = 50 + 2t_6 \\ -25 \leq t_6 \leq -16. \end{cases}.$$

Denoting these five systems respectively by $S_1, \ldots, S_5$ the integer points of K are finally given by the union of the integer points of the systems $\mathbf{x} = \mathbf{Pt} + \mathbf{q} \cup S_i$, for $1 \leq i \leq 5$. The systems $S_2, \ldots, S_5$ look simple enough to be considered as solution sets. What about S_1? The system S_1, as well as $S_2, \ldots, S_5$, satisfies a "back-substitution" property which is similar to that of a *regular chain* in the theory of polynomial system solving [1]. This property, when applied to S_1, says that for all $1 \leq i \leq 2$, every integer point of $\mathbb{R}^i$ solving all the inequalities of S_1 involving $t_1, \ldots, t_i$ only, extends to an integer point of $\mathbb{R}^{i+1}$ solving all the inequalities of S_1 involving $t_1, \ldots, t_{i+1}$.

With respect to the original Omega Test [17], our contributions are as follows.

1. We turn the decision procedure of the Omega Test into an algorithm decomposing all the integer points of a polyhedron.
2. Our decomposition is disjoint whereas the recursive calls in the original Omega Test may search for integer points in intersecting polyhedral regions.
3. The original Omega Test uses an ad-hoc routine for computing the integer solutions of linear equation systems, while we rely on Hermite normal form for this task. Consequently, we deduce complexity estimates for that task.
4. We also provide complexity estimates for the procedures GreySahdow and DarkShadow under realistic assumptions. From there, we derive complexity estimates for the entire algorithm, whereas no complexity estimates were known for the original Omega Test.

Our algorithm is discussed in Sect. 3 and 4 while complexity estimates are stated in Sect. 5. Sect. 2 gathers background materials on polyhedra.

2 Polyhedral Sets

This section is a review of the theory of polyhedral sets. It is based on the books of B. Grünbaum [10] and A. Schrijver [19], where proofs of the statements below can be found.

Given a positive integer d, we consider the d-dimensional Euclidean space $\mathbb{R}^d$ equipped with the Euclidean topology. Let K be a subset of $\mathbb{R}^d$. The *dimension* $\dim(K)$ of K is $a-1$ where a is the maximum number of affinely independent points in K. Let $\mathbf{a} \in \mathbb{R}^d$, let $b \in \mathbb{R}$ and denote by H the hyperplane defined by $H = \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{a}^T\mathbf{x} = b\}$. We say that the hyperplane H *supports* K if either $\sup\{\mathbf{a}^T\mathbf{x} \mid \mathbf{x} \in K\} = b$ or $\inf\{\mathbf{a}^T\mathbf{x} \mid \mathbf{x} \in K\} = b$ holds, but not both.

From now on, let us assume that K is convex. A set $F \subseteq K$ is a *face* if either $F = \varnothing$ or $F = K$, or if there exists a hyperplane H supporting K such that we have $F = K \cap H$. The set of all faces of K is denoted by $\mathcal{F}(K)$. We say that $F \in \mathcal{F}(K)$ is proper if we have $F \neq \varnothing$ or $F \neq K$. We note that the intersection of any family of faces of K is itself a face of K.

We say that K is a *polyhedral set* or *polyhedron* if it is the intersection of finitely many closed half-spaces of $\mathbb{R}^d$. We say that K is *full-dimensional*, if we have $\dim(K) = d$, that is, if the interior of K is not empty. The proper faces of K that are $\subseteq$-maximal are called *facets* and those of dimension zero are called *vertices*. We observe that every face of K is also a polyhedral set.

Let $H_1, \ldots, H_m$ be closed half-spaces such that the intersection $\cap_{i=1}^{i=m} H_i$ is *irredundant*, that is, $\cap_{i=1}^{i=m} H_i \neq \cap_{i=1, j\neq i}^{i=m} H_i$ for all $1 \leq j \leq m$. We observe that this intersection is closed and convex. For each i, where $1 \leq i \leq m$, let $\mathbf{a}_i \in \mathbb{R}^d$ and $b_i \in \mathbb{R}$ such that H_i is defined by $\mathbf{a}_i^T\mathbf{x} \leq b_i$. We denote by $\mathbf{A}$ the $m \times d$ matrix $(\mathbf{a}_i^T, 1 \leq i \leq m)$ and by $\mathbf{b}$ the vector $(b_1, \ldots, b_d)^T$.

From now on, we assume that $K = \cap_{i=1}^{i=m} H_i$ holds. Such irredundant decomposition of a polyhedral set can be computed from an arbitrary intersection of finitely many closed half-spaces, in time polynomial in both d and m, using linear programming; see L. Khachian in [15]. The following property is essential. For every face F of K, there exists a subset I of $\{1, \ldots, m\}$ such that F corresponds to the set of solutions to the system of equations and inequalities

$$\mathbf{a}_i^T\mathbf{x} = b_i \quad \text{for } i \in I, \quad \text{and} \quad \mathbf{a}_i^T\mathbf{x} \leq b_i \quad \text{for } i \notin I \ .$$

This latter property has several important consequences. For each i $1 \leq i \leq m$, the set $F_i = K \cap H_i$ is a facet of K and the border of K equals $\cup_{i=1}^{i=m} F_i$. In particular, each proper face of K is contained in a facet of K. Each facet of a facet of K is the intersection of two facets of K. Moreover, if the $m \times d$-matrix A has rank d (full column rank) then the $\subseteq$-minimal faces are the vertices. The set $\mathcal{F}(K)$ is finite and has at most 2^m elements

For $\mathbf{a} \in \mathbb{R}^d$ and $b \in \mathbb{R}$, we say that $\mathbf{a}^T\mathbf{x} \leq b$ is an *implicit equation* in $\mathbf{A}\mathbf{x} \leq \mathbf{b}$ if for all $\mathbf{x} \in \mathbb{R}^d$ we have

$$\mathbf{A}\mathbf{x} \leq \mathbf{b} \implies \mathbf{a}^T\mathbf{x} = b \ . \tag{1}$$

Following [19], we denote by $\mathbf{A}^=$ (resp. $\mathbf{A}^+$) and $\mathbf{b}^=$ (resp. $\mathbf{b}^+$) the rows of $\mathbf{A}$ and $\mathbf{b}$ corresponding to the implicit equations (resp. inequalities). The following properties are easy to prove. If K is not empty, then there exists $\mathbf{x} \in K$ satisfying both

$$\mathbf{A}^=\mathbf{x} = \mathbf{b}^= \quad \text{and} \quad \mathbf{A}^+\mathbf{x} < \mathbf{b}^+ \ .$$

The facets of K are in 1-to-1 correspondence with the inequalities of $\mathbf{A}^+\mathbf{x} \leq \mathbf{b}^+$. In addition, if K is full-dimensional, then $\mathbf{A}^+ = \mathbf{A}$ and $\mathbf{b}^+ = \mathbf{b}$ both hold; moreover the system of inequalities $\mathbf{A}\mathbf{x} \leq b$ is a unique representation of K, up to multiplication of inequalities by positive scalars.

From now on and in the sequel of this paper, we assume that variables are ordered as $x_1 > \cdots > x_d$. We call initial coefficient, or simply *initial*, of an inequality $\mathbf{a}_i^T\mathbf{x} \leq b_i$, for $1 \leq i \leq m$, the coefficient of $\mathbf{a}_i^T\mathbf{x}$ in its largest variable. Following the terminology of W. Pugh in [17], if v is the largest variable of the inequality $\mathbf{a}_i^T\mathbf{x} \leq b_i$, we say that this inequality is an *upper* (resp. *lower*) *bound* of v whenever the initial c of $\mathbf{a}_i^T\mathbf{x} \leq b_i$ is positive (resp. negative); indeed, we have $v \leq \frac{\gamma}{c}$ (resp. $v \geq \frac{\gamma}{c}$) where $\gamma = b_i - \mathbf{a}_i^T\mathbf{x} + c\,v$.

Canonical representation. Recall that we assume that none of the inequalities of $Ax \leq b$ is redundant. If K is full-dimensional and if the initial of each inequality in $Ax \leq b$ is 1 or -1, then we call $Ax \leq b$ the *canonical representation* of K w.r.t. the variable ordering $x_1 > \cdots > x_d$ and we denote it by $\mathsf{can}(K; x_1, \ldots, x_d)$.

We observe that the notion of *canonical representation* can also be expressed in a more geometrical and less algebraic way, that is, independently of any coordinate system. Assume again that K is full-dimensional and that the intersection $\cap_{i=1}^{i=n} H_i = K$ of closed half-spaces $H_1, \ldots, H_n$ is irredundant. Since K is full-dimensional, the supporting hyperplane of each facet of K must be the frontier of one half-space among $H_1, \ldots, H_n$. Clearly, two (or more) half-spaces among $H_1, \ldots, H_n$ may not have the same frontier without contradicting one of our hypotheses (K is full-dimensional, $\cap_{i=1}^{i=n} H_i$ is irredundant). Therefore, the half-spaces $H_1, \ldots, H_n$ are in one-to-one correspondence with the facets of K. This implies that there is a unique irredundant intersection of closed half-spaces equaling K and we denote it by $\mathsf{can}(K)$.

Projected representation. Let again $\mathbf{A}\mathbf{x} \leq \mathbf{b}$ be a *canonical representation* of the polyhedral set K w.r.t. the variable ordering $x_1 > \cdots > x_d$. We denote by $\mathbf{A}^{x_1}$ (resp. $\mathbf{A}^{<x_1}$) and $\mathbf{b}^{x_1}$ (resp. $\mathbf{b}^{<x_1}$) the rows of $\mathbf{A}$ and $\mathbf{b}$ corresponding to the inequalities whose largest variable is x_1 (resp. less than x_1). For each upper bound $cx_1 \leq \gamma$ of x_1 and each lower bound $-ax_1 \leq -\alpha$ of x_1 (where $c > 0$, $a > 0$, $\gamma \in \mathbb{R}[x_2, \ldots, x_d]$ and $\alpha \in \mathbb{R}[x_2, \ldots, x_d]$ hold), we have a new inequality $c\alpha - a\gamma \leq 0$. Augmenting $\mathbf{A}^{<x_1}$ with all inequalities obtained in this way, we obtain a new linear system which represents a polyhedral set which is the standard projection of K on the $d-1$ least coordinates of $\mathbb{R}^d$, namely $(x_2, \ldots, x_d)$; hence we denote this latter polyhedral set by $\mathit{\Pi}^{x_2,\ldots,x_d}K$ and we call it the *real shadow*

of K, following the terminology of [17]. The procedure by which $\Pi^{x_2,\ldots,x_d}K$ is computed from K is the well-known Fourier-Motzkin elimination procedure, see [15]. We call *projected representation* of K w.r.t. the variable ordering $x_1 > \cdots > x_d$ and denote by $\mathsf{proj}(K; x_1, \ldots, x_d)$ the linear system given by $\mathbf{A}^{x_1}\mathbf{x} \leq \mathbf{b}^{x_1}$ if $d = 1$ and, by the conjunction of $\mathbf{A}^{x_1}\mathbf{x} \leq \mathbf{b}^{x_1}$ and $\mathsf{proj}(\Pi^{x_2,\ldots,x_d}K; x_2, \ldots, x_d)$, otherwise.

3 Integer Solutions of Linear Equation Systems

We review how Hermite normal forms [6, 19] can be used to represent the integer solutions of systems of linear equations. Let $\mathbf{A} = (a_{i,j})$ and $H = (h_{i,j})$ be two matrices over $\mathbb{Z}$ with m rows and d columns, and let $\mathbf{b}$ be a vector over $\mathbb{Z}$ with d coefficients. We denote by r the rank of $\mathbf{A}$ and by h the maximum bit size of coefficients in the matrix $[\mathbf{A}\ \mathbf{b}]$. Definition 1 is taken from [13], see also [11].

Definition 1. *The matrix H is called a* column Hermite normal form (abbr. column HNF) *if there exists a strictly increasing map f from $[d-r+1, d] \cap \mathbb{Z}$ to $[1, m] \cap \mathbb{Z}$ satisfying the following properties for all $j \in [d-r+1, d] \cap \mathbb{Z}$:*

1. *for all integer i such that $i > f(j)$ holds, we have $h_{i,j} = 0$,*
2. *for all integer k such that $j < k \leq d$ holds, we have $h_{f(j),j} > h_{f(j),k} \geq 0$,*
3. *the first $d-r$ columns of H are equal to zero.*

We say that H is the column Hermite normal form *of $\mathbf{A}$ if H is a column Hermite normal form and there exists a uni-modular $d \times d$-matrix U over $\mathbb{Z}$ such that we have $H = \mathbf{A}U$. When those properties hold, we call $\{f(d-r+1), \ldots, f(d)\}$ the* pivot row set *of $\mathbf{A}$.*

Remark 1. The matrix $\mathbf{A}$ admits a unique column Hermite normal form. Let H be this column Hermite normal form and let U be the uni-modular $d \times d$-matrix given in Definition 1. Let us decompose U as $U = [U_L, U_R]$ where U_L(resp. U_R) consist of the first $d-r$ (resp. last r) columns of U. Then we define $H_L := \mathbf{A}U_L$ and $H_R := \mathbf{A}U_R$. We have $H_L = \mathbf{0}^{m,d-r}$, where $\mathbf{0}^{m,d-r}$ is the zero-matrix with m rows and $d-r$ columns. We observe that U_R is a full column-rank matrix. Moreover, if $\mathbf{A}$ is full row-rank, that is, if $r = m$ holds, then U_R is non-singular.

Lemma 2 shows how to compute the integer solutions of the system of linear equations $\mathbf{Ax} = \mathbf{b}$ when $\mathbf{A}$ is full rank. In the general case, one can use Lemma 1 to reduce to the hypothesis of Lemma 2. While the construction of this latter lemma relies on the HNF, alternative approaches are available. For instance, one can use the *equation elimination procedure* of the Omega Test [17], However, no running-time estimates is known for that procedure.

Notation 1 *For $I \subseteq \{1, \ldots, m\}$, we denote by $\mathbf{A}_I$ (resp. $\mathbf{b}_I$) the sub-matrix (resp. vector) of $\mathbf{A}$ (resp. $\mathbf{b}$) consisting of the rows of $\mathbf{A}$ (coefficients of $\mathbf{b}$) with indices in I.*

Lemma 1. *Let I be the pivot row set of $\mathbf{A}$, as given in Definition 1. Assume that $\mathbf{Ax} = \mathbf{b}$ admits at least one solution in $\mathbb{R}^d$. Then, for any $\mathbf{x} \in \mathbb{R}^d$, we have*

$$\mathbf{Ax} = \mathbf{b} \iff \mathbf{A}_I\mathbf{x} = \mathbf{b}_I \ .$$

Lemma 2. *We use the same notations as in Definition 1 and Remark 1. We assume that H_R is non-singular. Then, the system $\mathbf{Ax} = \mathbf{b}$ has an integer solution if and only if $H_R^{-1}\mathbf{b}$ is integral. In this case, all integral solutions to $\mathbf{Ax} = \mathbf{b}$ are given by $\mathbf{x} = \mathbf{P}\,\mathbf{t} + \mathbf{q}$ where*

1. *the columns of $\mathbf{P}$ consist of a $\mathbb{Z}$-basis of the space $\{\mathbf{x} : \mathbf{Ax} = \mathbf{0}\}$,*
2. *$\mathbf{q}$ is a particular solution of $\mathbf{Ax} = \mathbf{b}$, and*
3. *$\mathbf{t} = (t_1, \ldots, t_{d-r})$ is a vector of $d - r$ unknowns.*

The maximum absolute value of any coefficient in $\mathbf{P}$ (resp. $\mathbf{q}$) can be bounded over by $r^{r+1}L^{2r}$ (resp. $r^{r+1}L^{2r}$), where L is the maximum absolute value of a coefficient in $\mathbf{A}$ (resp. in either $\mathbf{A}$ or $\mathbf{b}$). Moreover, $\mathbf{P}$ and $\mathbf{q}$ can be computed within $O(mdr^2(\log r + \log L)^2 + r^4(\log r + \log L)^3)$ bit operations.

Example 1. Let $\mathbf{A}$, H and U be as follows:

$$\mathbf{A} = \begin{pmatrix} 3 & 4 & -4 & -1 \\ 2 & -2 & 8 & 4 \\ 5 & 2 & 4 & 3 \\ 3 & 5 & -5 & -2 \\ 2 & -3 & 9 & 5 \end{pmatrix}, \quad H = \begin{pmatrix} 0 & -18 & -1 & -15 \\ 0 & \mathbf{18} & \mathbf{2} & \mathbf{16} \\ 0 & 0 & 1 & 1 \\ 0 & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ 0 & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{pmatrix}, \quad U = \begin{pmatrix} -1 & 30 & -3 & -25 \\ 1 & -37 & 4 & 31 \\ 0 & -19 & 2 & 16 \\ 1 & 0 & 0 & 0 \end{pmatrix}.$$

The matrix H is the column HNF of $\mathbf{A}$, with unimodular matrix U and pivot row set $[2, 4, 5]$. We denote by H_R the sub-matrix of H whose coefficients are in bold fonts. Applying Lemma 1, we deduce that for any vector $\mathbf{b}$ such that $\mathbf{Ax} = \mathbf{b}$ admits one rational solution, we have:

$$\begin{cases} 3x_1 + 4x_2 - 4x_3 - x_4 = b_1 \\ 2x_1 - 2x_2 + 8x_3 + 4x_4 = b_2 \\ 5x_1 + 2x_2 + 4x_3 + 3x_4 = b_3 \\ 3x_1 + 5x_2 - 5x_3 - 2x_4 = b_4 \\ 2x_1 - 3x_2 + 9x_3 + 5x_4 = b_5 \end{cases} \Leftrightarrow \begin{cases} 2x_1 - 2x_2 + 8x_3 + 4x_4 = b_2 \\ 3x_1 + 5x_2 - 5x_3 - 2x_4 = b_4 \\ 2x_1 - 3x_2 + 9x_3 + 5x_4 = b_5 \end{cases}. \tag{2}$$

We apply Lemma 2: if $\mathbf{Ax} = \mathbf{b}$ is consistent over $\mathbb{Q}$ and if $H_R^{-1}[b_2, b_4, b_5]^T$ is integral, then all the integer solutions of the second equation system in Relation (2) are given by $\mathbf{x} = \mathbf{Pt} + \mathbf{q}$, where $\mathbf{P} = [-1, 1, 0, 1]^T$, $\mathbf{q} = [\frac{5}{3}b_2 - \frac{19}{3}b_4 - \frac{155}{3}b_5, -\frac{37}{18}b_2 + \frac{73}{9}b_4 + \frac{575}{9}b_5, -\frac{19}{18}b_2 + \frac{37}{9}b_4 + \frac{296}{9}b_5]^T$, $\mathbf{t} = (t_1)$ and t_1 is a new variable.

4 Integer Solutions of Inequality Systems

In this section, we present an algorithm for computing the integer points of a polyhedron $K \subseteq \mathbb{R}^d$, that is, the set $K \cap \mathbb{Z}^d$. As mentioned in the introduction, we adapt the Omega Test invented by W. Pugh [17] for deciding whether or not a polyhedral set has an integer point.

With the notations of Section 2, consider again the polyhedron K:

$$\begin{cases} a_{11}x_1 + \cdots + a_{1d}x_d \le b_1 \\ \vdots \\ a_{m1}x_1 + \cdots + a_{md}x_d \le b_m \end{cases}$$

We aim at eliminating x_1, so we consider two inequalities in x_1:

– one with $a_{i1} > 0$, called upper bound:
$$l_i : \ a_{i1}x_1 + \cdots + a_{id}x_d \le b_i,$$
– another one with $a_{j1} < 0$, called lower bound:
$$l_j : \ a_{j1}x_1 + \cdots + a_{jd}x_d \le b_j.$$

The real projection makes a linear combination r_{ij} of l_i and l_j:

$$-a_{j1}(a_{i2}x_2 + \cdots + a_{id}x_d) + a_{i1}(a_{j2}x_2 + \cdots + a_{jd}x_d) \le -a_{j1}b_i + a_{i1}b_j.$$

The dark projection translates r_{ij} towards the center of K into ds_{ij}:

$$-a_{j1}(a_{i2}x_2+\cdots+a_{id}x_d)+a_{i1}(a_{j2}x_2+\cdots+a_{jd}x_d) \le -a_{j1}b_i+a_{i1}b_j-(a_{i1}-1)(-a_{j1}-1).$$

Lemma 3. *For any integer point $(x_2, \ldots, x_d) \in \mathbb{Z}^{d-1}$ satisfying ds_{ij}, there exists at least one integer $x_1 \in \mathbb{Z}$ satisfying both l_i and l_j.*

Let $S^{<x_1}$ be the subset of the inequalities defining K in which x_1 does not occur. Combining all the real projections with $S^{<x_1}$, we obtain the so-called *real shadow* of the polyhedron K *w.r.t.* x_1, that we denote by R. Meanwhile, combining all the dark projections with $S^{<x_1}$, we obtain the so-called *dark shadow* of the polyhedron K *w.r.t.* x_1, that we denote by D. Without loss of generality, we assume $D = S^{<x_1} \cap ds_1 \cap \cdots \cap ds_t$, where t is a positive integer and each ds_i is a dark projection introduced above for $1 \le i \le t$. Obviously, we have $D = R \cap ds_1 \cap \cdots \cap ds_t$ since $D \subset R \subset S^{<x_1}$.

We define the *grey shadow* of K w.r.t. x_1 as $G := R \setminus D$. Then, it is easy to deduce that $G = \biguplus_{1 \le i \le t} G_i$, where $\uplus$ means disjoint union and $G_i = R \cap ds_1 \cap \cdots \cap ds_{i-1} \cap \neg ds_i$, here $\neg ds_i$ is the negation of ds_i for $1 \le i \le t$. We call each G_i obtained by this way a grey shadow part of K.

We use the following steps to introduce the ideas of procedure GreyShadow:

1. let $result := \{\}$;
2. for each lower bound $a_{j1}x_1 + \cdots + a_{jd}x_d \le b_j$ of x_1 in K_1
3. for each upper bound $a_{i1}x_1 + \cdots + a_{id}x_d \le b_i$ of x_1 in K_1
4. for each integer ℓ from 1 to $\lfloor(-a_{j1}a_{i1} + a_{j1} - a_{i1})/a_{i1}\rfloor$ do
5. let $g_{ji\ell} : a_{j1}x_1 + \cdots + a_{jd}x_d = b_j - \ell$;
6. solving for the integer solutions of $g_{ji\ell}$, we obtain a solution of the form $\mathbf{x} = \mathbf{Pt} + \mathbf{q}$;
7. substitute $\mathbf{x} = \mathbf{Pt} + \mathbf{q}$ into $K \cap \neg ds_{ji}$, we obtain a new polyhedron K_1;
8. let $result = result \cup \{\mathbf{x} = \mathbf{Pt} + \mathbf{q} \cap K_1\}$;
9. end for;
10. let $K = K \cap ds_{ji}$;
11. end for;
12. end for;
13. return $result$.

Example 2. Given a polyhedron K:

$$\begin{cases} 3x_1 - 2x_2 + x_3 \leq 7 \\ -2x_1 + 2x_2 - x_3 \leq 12 \\ -4x_1 + x_2 + 3x_3 \leq 15 \\ -x_2 \leq -25 \end{cases},$$

we have one upper bound and two lower bounds for x_1. For the lower bound $-2x_1 + 2x_2 - x_3 \leq 12$ and the upper bound $3x_1 - 2x_2 + x_3 \leq 7$, we have $\lfloor (-a_{j1}a_{i1} + a_{j1} - a_{i1})/a_{i1} \rfloor = 0$. Thus, one recursive call with:

$$\begin{cases} -2x_1 + 2x_2 - x_3 = 12 \\ 3x_1 - 2x_2 + x_3 \leq 7 \\ -4x_1 + x_2 + 3x_3 \leq 15 \\ -x_2 \leq -25 \end{cases}.$$

Solving $-2x_1 + 2x_2 - x_3 = 12$ via Hermite normal form leads to:

$$\begin{cases} x_1 = x_4 \\ x_2 = x_5 + 1 \\ x_3 = -2x_4 + 2x_5 + 1 \end{cases},$$

where x_4, x_5 are new variables. Thus, the grey shadow part G_{20} is:

$$\begin{cases} x_4 \leq 8 \\ -10x_4 + 7x_5 \leq 11 \\ -x_5 \leq -24 \end{cases}.$$

Befor compute the other grey shadow parts generated by any other lower bounds, we add the dark projection $2x_2 - x_3 \leq 48$ to the polyhedron K. For the lower bound $-4x_1 + x_2 + 3x_3 \leq 15$, we have $\lfloor (-a_{j1}a_{i1} + a_{j1} - a_{i1})/a_{i1} \rfloor = 1$, yielding two other grey shadow parts, which are disjoint with G_{20}.

5 Complexity Analysis

We use the following figure to illustrate our algorithm:

Notation 2 *Fig. 2 illustrates the tree of recursive calls for the main algorithm, the* IntegerSolve *procedure. The root of the tree is labelled with* **S**, *which stands for the input system. The left (resp. right) child of a node, other than a leaf, is labelled by* **D** *(resp.* **G***) which stands for the output of the* DarkShadow *procedure (resp. the* GreyShadow *procedure). Since the* DarkShadow *procedure generates one input, we use a simple* $\rightarrow$ *arrow as an edge to a* **D***-node. However, the* GreyShadow *procedure may generate several inputs, leading to several recursive calls. Thus, we use a* $\Rightarrow$ *arrow as an edge to a* **G***-node. The numbers on the right-hand side of Fig. 2 stand for the levels in the tree.*

Hypothesis 1 *We assume that during the execution of the function call* IntegerSolve*(K), for any polyhedral set K', input of a recursive call, each facet of the dark shadow of K' is parallel to a facet of the real shadow of K'.*

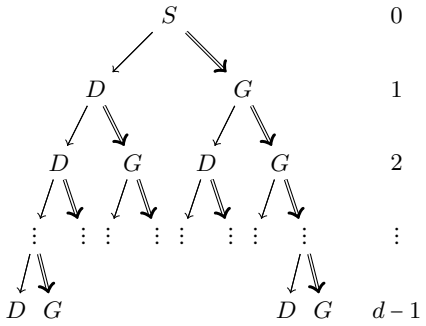

Fig. 2: Illustration of algorithm

To state our main result, we need a notation for the running time of solving a linear program. Indeed, linear programming is an essential tool for removing redundant inequalities generated by Fourier-Motzkin elimination, see [15].

Notation 3 *For an input linear program with total bit size H and with d variables, we denote by $\mathsf{LP}(d, H)$ an upper bound for the number of bit operations required for solving this linear program. For instance, in the case of Karmarkar's algorithm [14], we have $\mathsf{LP}(d, H) \in O(d^{3.5} H^2 \cdot \log H \cdot \log\log H)$.*

Lemma 4. *Let $0 \leq k < d$ be an integer. Let $f_{d,m,k}$ be the number of k-dimensional faces of K. Then, we have*

$$f_{d,m,k} \leq \binom{m}{d-k}.$$

Therefore, we have

$$f_{d,m,0} + f_{d,m,1} + \cdots + f_{d,m,d-1} \;\leq\; m^d.$$

Lemma 5. *Any path in Fig. 2 can be implemented within $O(m^{2r+2} d^{3+\varepsilon} r^{10} (\log d + \log L)^3) + O(r m^{2r+2} \mathsf{LP}(d, d m^r r^3 (\log d + \log L)))$ bit operations.*

Let T_r be the total number of nodes in the r-th level.

Lemma 6. *We have: $T_{r+1} \leq m^{r+1} d^r r^{2r^2} L^{3r^2} T_r$ for $0 \leq r \leq d-2$. Thus, we have $T_{d-1} \leq m^{d^2} d^{3d^3} L^{3d^3}$.*

Theorem 1. *Under Hypothesis 1, the call function $\mathsf{IntegerSolve}(K)$ runs within $O(m^{2d^2} d^{4d^3} L^{4d^3} \mathsf{LP}(d, m^d d^4 (\log d + \log L)))$ bit operations.*

6 Experimentation

We have implemented the algorithm presented in the first paper within the `Polyhedra` library in MAPLE. This library is publicly available in source on the download page of the `RegularChains` library at `www.regularchains.org`

In this section, we gather experimental results obtained with our implementations. Our objectives were two-fold:

1. Check on various test-cases whether or not Hypothesis 1 holds,
2. Compare two strategies for computing the integer solutions of linear systems: the one proposed by W. Pugh in [17] and the one based on Hermite normal form (HNF).

We have used test-cases coming from various application areas: regular polytopes (first 5 examples in Table 1), examples from Presburger arithmetic (next 5 examples in Table 1), random polytopes (next 5 examples in Table 1), random unbounded polyhedra (next 5 examples in Table 1), examples from text-books (next 3 examples in Table 1) and examples from research articles on automatic parallelization of for-loop nests (last 4 examples in Table 1).

For each example, Table 1 gives the number of defining inequalities (Column m), the number of variables (Column d), the maximum absolute value of an input coefficient (Column L), the number of polyhedra returned by IntegerSolve (Column m_{o}), the maximum absolute value of an output coefficient (Column L_{o}) and whether Hypothesis 1 holds or not (Column ?Hyp).

Recall from Section 4.1 that Step $(S4)$ of the IntegerNormalize procedure, can use either the HNF method introduced in Lemma 2, or the method introduced by W. Pugh in [17]. We implemented both of them. It is important to observe that Pugh's method does not solve system of linear equations according to our prescribed variable order, on the contrary of the HNF method. In fact, Pugh's method determines a variable order dynamically, based on coefficient size considerations, In Table 1, the columns t_{H} and t_{P} correspond to the timings for the HNF and Pugh's method, respectively.

From Table 1, we make a few observations:

1. Hypothesis 1 holds for most examples while it ussually does not hold for random ones.
2. For 16 out of 27 examples, IntegerSolve produces a single component, which means that each such input polyhedron has no integer points in its grey shadow; this is, in particular the case for regular polytopes and for examples from automatic parallelization.
3. When a decomposition consists of more than one components, most of those components are points; for example, the decomposition of Unbounded 2 has 61 components and 46 of them are points.
4. Coefficients of the output polyhedra are usually not much larger than the coefficients of the corresponding input polyhedron.
5. Among the challnging problems, some of them are solved faster when IntegerNormalize is based on HNF (e.g. Bounded 7) while others are solved faster when IntegerNormalize is based on Pugh's method (e.g. Bounded 9) which suggests that having both approaches at hand is useful.

Table 1: Implementation

Example	m	d	L	$m_\circ$	$L_\circ$	?Hyp	t_H	t_P
Tetrahedron	4	3	1	1	1	yes	0.695	0.697
Cuboctahedron	14	3	2	1	2	yes	1.855	1.846
Octahedron	8	3	1	1	1	yes	1.357	1.357
TruncatedOctahedron	14	3	3	1	1	yes	1.995	1.977
TruncatedTetrahedron	8	3	1	1	1	yes	1.461	1.468
Presburger 1	3	2	2	1	1	yes	0.083	0.082
Presburger 2	3	2	20	1	20	yes	0.184	0.182
Presburger 3	3	2	18	3	4	yes	0.287	0.260
Presburger 4	3	4	5	2	12	yes	0.706	0.871
Presburger 6	4	5	89	6	35	yes	0.893	0.746
Bounded 5	6	3	19	4	224	yes	16.433	15.091
Bounded 7	8	3	19	3	190	no	138.448	239.637
Bounded 8	4	3	25	5	67	yes	6.462	3.821
Bounded 9	6	3	18	6	74	no	23.574	16.763
Bounded 10	4	3	15	1	176	yes	0.559	0.558
Unbounded 2	3	4	10	61	2255	no	0.547	0.600
Unbounded 3	4	4	20	1	20	no	0.981	0.987
Unbounded 4	6	5	2	1	2	no	0.722	0.510
Unbounded 5	5	4	8	1	8	no	1.321	1.319
Unbounded 6	10	4	8	1	8	no	1.494	1.479
P91	12	3	96	5	96	no	19.318	15.458
Sys_1	6	3	15	2	67	yes	2.413	1.915
Sys_3	8	3	1	1	1	yes	1.481	1.479
Automatic	8	2	999	1	999	yes	0.552	0.549
Automatic2	6	4	1	1	2	yes	1.115	1.113
Automatic3	3	4	1	1	1	yes	0.130	0.135
Automatic4	3	5	1	1	1	yes	0.227	0.232

Bibliography

[1] P. Aubry, D. Lazard, and M. Moreno Maza. On the theories of triangular sets. *J. Symb. Comput.*, 28:105–124, July 1999.

[2] Alexander I. Barvinok. A polynomial time algorithm for counting integral points in polyhedra when the dimension is fixed. *Math. Oper. Res.*, 19(4):769–779, 1994.

[3] Alexander I. Barvinok. *Integer Points in Polyhedra*. Contemporary mathematics. European Mathematical Society, 2008.

[4] M. Beck. *Integer Points in Polyhedra– Geometry, Number Theory, Representation Theory, Algebra, Optimization, Statistics: AMS-IMS-SIAM Joint Summer Research Conference, June 11-15, 2006, Snowbird, Utah*. Contemporary mathematics - American Mathematical Society. American Mathematical Society, 2008.

[5] Changbo Chen, Xiaohui Chen, Abdoul-Kader Keita, Marc Moreno Maza, and Ning Xie. MetaFork: A compilation framework for concurrency models targeting hardware accelerators and its application to the generation of parametric CUDA kernels. In *Proceedings of CASCON 2015*, pages 70–79, 2015.

[6] Henri Cohen. *A course in computational algebraic number theory*, volume 138. Springer Science & Business Media, 2013.

[7] Paul Feautrier. Parametric integer programming. *RAIRO Recherche Op'erationnelle*, 22, 1988. `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.30.9957&rep=rep1&type=pdf`.

[8] Paul Feautrier. Automatic parallelization in the polytope model. In *The Data Parallel Programming Model: Foundations, HPF Realization, and Scientific Applications*, pages 79–103, London, UK, UK, 1996. Springer-Verlag. `http://dl.acm.org/citation.cfm?id=647429.723579`.

[9] M. J. Fischer and M. O. Rabin. Super-exponential complexity of presburger arithmetic. Technical report, Cambridge, MA, USA, 1974.

[10] Branko Grünbaum. *Convex Polytops*. Springer, New York, NY, USA, 2003.

[11] Rui-Juan Jing, Chun-Ming Yuan, and Xiao-Shan Gao. A polynomial-time algorithm to compute generalized hermite normal form of matrices over $\mathbb{Z}[x]$. *CoRR*, abs/1601.01067, 2016.

[12] CN Jones, EC Kerrigan, and JM Maciejowski. On polyhedral projection and parametric programming. *Journal of Optimization Theory and Applications*, 138(2):207–220, 2008.

[13] Ravindran Kannan and Achim Bachem. Polynomial algorithms for computing the smith and hermite normal forms of an integer matrix. *siam Journal on Computing*, 8(4):499–507, 1979.

[14] N. Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, STOC '84, pages 302–311, New York, NY, USA, 1984. ACM.

[15] Leonid Khachiyan. Fourier-motzkin elimination method. In Christodoulos A. Floudas and Panos M. Pardalos, editors, *Encyclopedia of Optimization, Second Edition*, pages 1074–1077. Springer, 2009.

[16] Matthias Köppe and Sven Verdoolaege. Computing parametric rational generating functions with a primal barvinok algorithm. *Electr. J. Comb.*, 15(1), 2008.

[17] William Pugh. The omega test: a fast and practical integer programming algorithm for dependence analysis. In Joanne L. Martin, editor, *Proceedings Super-*

computing '91, Albuquerque, NM, USA, November 18-22, 1991, pages 4–13. ACM, 1991.

[18] William Pugh. Counting solutions to presburger formulas: How and why. In Vivek Sarkar, Barbara G. Ryder, and Mary Lou Soffa, editors, *Proceedings of the ACM SIGPLAN'94 Conference on Programming Language Design and Implementation (PLDI), Orlando, Florida, USA, June 20-24, 1994*, pages 121–134. ACM, 1994.

[19] Alexander Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, Inc., New York, NY, USA, 1986.

[20] David Wonnacott. Omega test. In *Encyclopedia of Parallel Computing*, pages 1355–1365. 2011.