Computing the Integer Points of a Polyhedron

Algorithm

Rui-Juan Jing[1,2] and Marc Moreno Maza[2,3]

[1]Key Laboratoty of Mathematics Mechnization,
Academy of Mathematics and Systems Science,
Chinese of Academy of Sciences
[2]University of Western Ontario, London, Ontario
[3]IBM Center for Advanced Studies, Markham, Ontario

CASC 2017, September 19

Plan

Motivating example: cache-line accessed by a for-loop

for $i = 2$ to $N - 1$ do

  for $j = 2$ to $N - 1$ do

    $a(i,j) = 2 * a(i,j) + a(i-1,j) + a(i+1,j) + a(i,j-1) + a(i,j+1)$

Motivating example: cache-line accessed by a for-loop

for $i = 2$ to $N - 1$ do
    for $j = 2$ to $N - 1$ do
        $a(i, j) = 2 * a(i, j) + a(i - 1, j) + a(i + 1, j) + a(i, j - 1) + a(i, j + 1)$

Cache lines touched by this loop:

$$(\Sigma\ x, y : (\exists\ i, j, \triangle i, \triangle j :$$
$$x = (i + \triangle i - 1) \div 16 \wedge y = j + \triangle j \wedge 2 \leq i, j \leq N - 1$$
$$\wedge -1 \leq \triangle i + \triangle j, \triangle i - \triangle j \leq 1)$$
$$: 1)$$

Motivating example: cache-line accessed by a for-loop

for $i = 2$ to $N - 1$ do
  for $j = 2$ to $N - 1$ do
    $a(i, j) = 2 * a(i, j) + a(i - 1, j) + a(i + 1, j) + a(i, j - 1) + a(i, j + 1)$

Cache lines touched by this loop:

$$(\Sigma\ x, y : (\exists\ i, j, \triangle i, \triangle j : \\ x = (i + \triangle i - 1) \div 16 \wedge y = j + \triangle j \wedge 2 \leq i, j \leq N - 1 \\ \wedge -1 \leq \triangle i + \triangle j, \triangle i - \triangle j \leq 1) \\ : 1)$$

$$\begin{cases} 0 \leq (i + \triangle i - 1)/16 - x < 1 \\ y = j + \triangle j \\ 2 \leq i, j \leq N - 1 \\ -1 \leq \triangle i + \triangle j, \triangle i - \triangle j \leq 1 \end{cases}$$

Motivating example: cache-line accessed by a for-loop

for $i = 2$ to $N - 1$ do
  for $j = 2$ to $N - 1$ do
    $a(i,j) = 2 * a(i,j) + a(i-1,j) + a(i+1,j) + a(i,j-1) + a(i,j+1)$

Cache lines touched by this loop:

$$(\Sigma\ x, y : (\exists\ i, j, \triangle i, \triangle j : \\ x = (i + \triangle i - 1) \div 16 \wedge y = j + \triangle j \wedge 2 \le i,j \le N-1 \\ \wedge -1 \le \triangle i + \triangle j, \triangle i - \triangle j \le 1) \\ : 1)$$

$$\begin{cases} 0 \le (i + \triangle i - 1)/16 - x < 1 \\ y = j + \triangle j \\ 2 \le i,j \le N-1 \\ -1 \le \triangle i + \triangle j, \triangle i - \triangle j \le 1 \end{cases}$$

Simplifying with our code

$$\begin{cases} -x \le 0, & 16x - y - N \le -3 \\ 16x - N \le -1, & 16x + y - 2N \le -2 \\ 1 \le y - N \le 0, & -N \le -3 \end{cases}$$

Motivating example: cache-line accessed by a for-loop

for $i = 2$ to $N - 1$ do
    for $j = 2$ to $N - 1$ do
        $a(i,j) = 2 * a(i,j) + a(i-1,j) + a(i+1,j) + a(i,j-1) + a(i,j+1)$

Cache lines touched by this loop:

$$(\Sigma\ x, y : (\exists\ i, j, \triangle i, \triangle j : x = (i + \triangle i - 1) \div 16 \wedge y = j + \triangle j \wedge 2 \leq i, j \leq N - 1 \wedge -1 \leq \triangle i + \triangle j, \triangle i - \triangle j \leq 1) : 1)$$

$$\begin{cases} 0 \leq (i + \triangle i - 1)/16 - x < 1 \\ y = j + \triangle j \\ 2 \leq i, j \leq N - 1 \\ -1 \leq \triangle i + \triangle j, \triangle i - \triangle j \leq 1 \end{cases}$$

Simplifying with our code

$$\begin{cases} -x \leq 0, & 16x - y - N \leq -3 \\ 16x - N \leq -1, & 16x + y - 2N \leq -2 \\ 1 \leq y - N \leq 0, & -N \leq -3 \end{cases}$$

When $N = 500$, $(\Sigma\ x, y : 0 \leq x \leq 31 \wedge 1 \leq y \leq 500 : 1) = 16000$

Related work

1. Fourier-Motzkin elimination: computing the rational points (thus all the points) of a polyhedron in $\mathbb{R}^d$ given by m inequalities;
   Complexity: polynomial in m^d, thus single exponential in d
   (Fourier-Motzkin algorithm; L. Khachiyan, 2009)
2. Counting the number of integer points of a bounded polyhedron;
   Complexity: polynomial for fixed dimention.
   (A. Barvinok, 1999)
3. Deciding Presburger arithmetic such as
   $$(\forall x \in \mathbb{Z})\,(\exists y \in \mathbb{Z}) \,:\, (y + y = x) \vee (y + y + 1 = x)$$
   Complexity: doubly exponential in d
   (Fischer & Rabin, 1974).
4. *Omega test*, can decide Presburger arithmetic;
   essential in the analysis and transformation of computer programs;
   (W. Pugh, 1991).
   Complexity: No complexity estimate known until our work.

Our contribution

1. Based on the Omega test, we propose an algorithm for decomposing a polyhedron into "simpler" polyhedra, each of them having at least one integer point and good structural properties;

2. Under a mild assumption (almost always verified in practice), this decomposition can be computed within

$$O(m^{2d^2} d^{4d^3} L^{4d^3} \mathrm{LP}(d, m^d d^4 (\log d + \log L))) \text{ bit operations,}$$

   where $\mathrm{LP}(d, H)$ is an upper bound for solving a linear program with total bit size H and d variables;

3. We implemented two versions of our algorithm in Maple:
   - one "sparse" with explict equations and inequalities as input,
   - another with dense matrices encoding equations and inequalities.

Decomposing the integer points of a polyhedron

Example

Input: $K_1 : \begin{cases} 3x_1 - 2x_2 + x_3 \le 7 \\ -2x_1 + 2x_2 - x_3 \le 12 \\ -4x_1 + x_2 + 3x_3 \le 15 \\ -x_2 \le -25 \end{cases}$, assume $x_1 \; > \; x_2 \; > x_3$.

Output: $K_1^1, K_1^2, K_1^3, K_1^4, K_1^5$ given by:

$$\begin{cases} 3x_1 - 2x_2 + x_3 \le 7 \\ -2x_1 + 2x_2 - x_3 \le 12 \\ -4x_1 + x_2 + 3x_3 \le 15 \\ 2x_2 - x_3 \le 48 \\ -5x_2 + 13x_3 \le 67 \\ -x_2 \le -25 \\ 2 \le x_3 \le 17 \end{cases}, \quad \begin{cases} x_1 = 15 \\ x_2 = 27 \\ x_3 = 16 \end{cases}, \quad \begin{cases} x_1 = 18 \\ x_2 = 33 \\ x_3 = 18 \end{cases}, \quad \begin{cases} x_1 = 14 \\ x_2 = 25 \\ x_3 = 15 \end{cases}, \quad \begin{cases} x_1 = 19 \\ x_2 = 50 + t \\ x_3 = 50 + 2t \\ -25 \le t \le -16. \end{cases}$$

Decomposing the integer points of a polyhedron

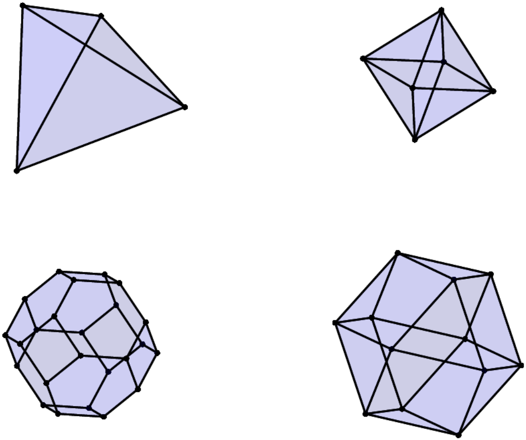
Output: $K_1^1, K_1^2, K_1^3, K_1^4, K_1^5$ given by:

$$\begin{cases} 3x_1 - 2x_2 + x_3 \le 7 \\ -2x_1 + 2x_2 - x_3 \le 12 \\ -4x_1 + x_2 + 3x_3 \le 15 \\ 2x_2 - x_3 \le 48 \\ -5x_2 + 13x_3 \le 67 \\ -x_2 \le -25 \\ 2 \le x_3 \le 17 \end{cases}, \begin{cases} x_1 = 15 \\ x_2 = 27 \\ x_3 = 16 \end{cases}, \begin{cases} x_1 = 18 \\ x_2 = 33 \\ x_3 = 18 \end{cases}, \begin{cases} x_1 = 14 \\ x_2 = 25 \\ x_3 = 15 \end{cases}, \begin{cases} x_1 = 19 \\ x_2 = 50 + t \\ x_3 = 50 + 2t \\ -25 \le t \le -16. \end{cases}$$

- An integer point solves K_1 iff it solves either K_1^1, K_1^2, K_1^3, K_1^4 or K_1^5.
- Each of $K_1^1, K_1^2, K_1^3, K_1^4, K_1^5$ has at least one integer point.
- For each K_1^i, each integer point in the projection can be lifted to an integer point in the polyhedron.

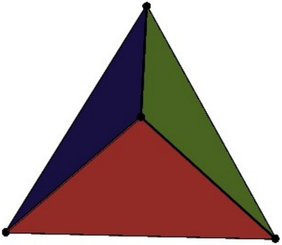
Plan

How polyhedra look like?



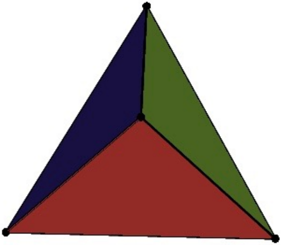
Polyhedra geometrically

- $\mathbb{R}^d$: d-dim Euclidean space
- for $\mathbf{a} \in \mathbb{R}^d$, $b \in \mathbb{R}$, the set $P = \{\mathbf{x} \in \mathbb{R}^d | \mathbf{a}^T \mathbf{x} = b\}$ is a *hyperplane* and,
- $H = \{\mathbf{x} \in \mathbb{R}^d | \mathbf{a}^T \mathbf{x} \leq b\}$ is a *closed half-space*.



Polyhedra geometrically

- $\mathbb{R}^d$: d-dim Euclidean space
- for $\mathbf{a} \in \mathbb{R}^d$, $b \in \mathbb{R}$, the set $P = \{\mathbf{x} \in \mathbb{R}^d | \mathbf{a}^T\mathbf{x} = b\}$ is a *hyperplane* and,
- $H = \{\mathbf{x} \in \mathbb{R}^d | \mathbf{a}^T\mathbf{x} \leq b\}$ is a *closed half-space*.



- A *polyhedron* K is the intersection of finitely many closed half-spaces $H_1, \ldots, H_m$, that is, $K = \cap_{i=1}^{i=m} H_i$
- This intersection is *irredundant* if $K \neq \cap_{i=1, j\neq i}^{i=m} H_i$ for all $1 \leq j \leq m$
- P_i: hyperplane associated with H_i.
- Then, any $\varnothing \neq \left(\bigcap_{i \in I} P_i\right) \cap K$ is a *face* of K, for $I \subseteq \{1, \ldots, m\}$

Polyhedra algebrically

Consider $K_0 = \cap_{i=1}^{i=6} H_i$ with

$$\begin{cases} H_1: & 2x + 3y - 4z + 3w \leq 1 \\ H_2: & -2x - 3y + 4z - 3w \leq -1 \\ H_3: & -13x - 18y + 24z - 20w \leq -1 \\ H_4: & -26x - 40y + 54z - 39w \leq 0 \\ H_5: & -24x - 38y + 49z - 31w \leq 5 \\ H_6: & 54x + 81y - 109z + 81w \leq 2 \end{cases}$$

Polyhedra algebrically

Consider $K_0 = \cap_{i=1}^{i=6} H_i$ with

$$\left\{\begin{array}{lr} H_1 : & 2x + 3y - 4z + 3w \leq 1 \\ H_2 : & -2x - 3y + 4z - 3w \leq -1 \\ H_3 : & -13x - 18y + 24z - 20w \leq -1 \\ H_4 : & -26x - 40y + 54z - 39w \leq 0 \\ H_5 : & -24x - 38y + 49z - 31w \leq 5 \\ H_6 : & 54x + 81y - 109z + 81w \leq 2 \end{array}\right. \quad \Leftrightarrow \left\{\begin{array}{l} 2x + 3y - 4z + 3w = 1 \\ -13x - 18y + 24z - 20w \leq -1 \\ -26x - 40y + 54z - 39w \leq 0 \\ -24x - 38y + 49z - 31w \leq 5 \\ 54x + 81y - 109z + 81w \leq 2 \end{array}\right.$$

Polyhedra algebrically

Consider $K_0 = \cap_{i=1}^{i=6} H_i$ with

$$\begin{cases} H_1: & 2x + 3y - 4z + 3w \leq 1 \\ H_2: & -2x - 3y + 4z - 3w \leq -1 \\ H_3: & -13x - 18y + 24z - 20w \leq -1 \\ H_4: & -26x - 40y + 54z - 39w \leq 0 \\ H_5: & -24x - 38y + 49z - 31w \leq 5 \\ H_6: & 54x + 81y - 109z + 81w \leq 2 \end{cases} \Leftrightarrow \begin{cases} 2x + 3y - 4z + 3w = 1 \\ -13x - 18y + 24z - 20w \leq -1 \\ -26x - 40y + 54z - 39w \leq 0 \\ -24x - 38y + 49z - 31w \leq 5 \\ 54x + 81y - 109z + 81w \leq 2 \end{cases}$$

Remark

Any polyhedron can be represented by

$$\begin{cases} \mathbf{A}^{=}\mathbf{x} = \mathbf{b}^{=} \\ \mathbf{A}^{\leq}\mathbf{x} \leq \mathbf{b}^{\leq} \end{cases}$$

- where $\mathbf{x} = [x_1, \ldots, x_d]^T$, $\mathbf{A}^{=} \in \mathbb{R}^{m_1 \times d}$, $\mathbf{b}^{=} \in \mathbb{R}^{m_1}$, $\mathbf{A}^{\leq} \in \mathbb{R}^{m_2 \times d}$, $\mathbf{b}^{\leq} \in \mathbb{R}^{m_2}$, and
- $\mathbf{A}^{\leq}\mathbf{x} \leq \mathbf{b}^{\leq}$ no implicit equations.

Plan

Algorithm

Consider the polyhedron K of $\mathbb{R}^4$ given below ($\mathbf{Ax} \leq \mathbf{b}$):

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

Algorithm-IntegerNormalize

Procedure 1- IntegerNormalize($\mathbf{Ax} \le \mathbf{b}$):

1. Solve integer solutions for (implicit) equations:
   - Tools: Hermite normal form;
   - Return $\mathbf{x} = \mathbf{Pt} + \mathbf{q}$, where $\mathbf{t}$ is a new unknown vector with less length than $\mathbf{x}$.
2. Substitute $\mathbf{x} = \mathbf{Pt} + \mathbf{q}$ into $\mathbf{Ax} \le \mathbf{b}$ and remove redundant inequalities:
   - $\mathbf{cx} \le d$ is implied by $\mathbf{Ax} \le \mathbf{b} \iff \sup\{-(\mathbf{cx} - d) | \mathbf{Ax} \le \mathbf{b}\} = 0$;
   - Return $\mathbf{Mt} \le \mathbf{v}$.

Algorithm-IntegerNormalize

Procedure 1- IntegerNormalize($\mathbf{Ax} \le \mathbf{b}$):

1. Solve integer solutions for (implicit) equations:
   - Tools: Hermite normal form;
   - Return $\mathbf{x} = \mathbf{Pt} + \mathbf{q}$, where $\mathbf{t}$ is a new unknown vector with less length than $\mathbf{x}$.
2. Substitute $\mathbf{x} = \mathbf{Pt} + \mathbf{q}$ into $\mathbf{Ax} \le \mathbf{b}$ and remove redundant inequalities:
   - $\mathbf{cx} \le d$ is implied by $\mathbf{Ax} \le \mathbf{b} \iff \sup\{-(\mathbf{cx} - d) | \mathbf{Ax} \le \mathbf{b}\} = 0$;
   - Return $\mathbf{Mt} \le \mathbf{v}$.

In our example, implicit equation: $2x + 3y - 4z + 3w = 1$
the systems $\mathbf{x} = \mathbf{Pt} + \mathbf{q}$ and $\mathbf{Mt} \le \mathbf{v}$ are given by:

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

Algorithm-DarkShadow

- Consider a polyhedron K:

$$\begin{cases} a_{11}t_1 + \cdots + a_{1d}t_d & \leq b_1 \\ \vdots & \\ a_{m1}t_1 + \cdots + a_{md}t_d & \leq b_m \end{cases}$$

- We aim at eliminating t_1, so we consider 2 inequalities in t_1.
- one where $a_{i1} > 0$, called upper bound:

$$l_i : \; a_{i1}t_1 + \cdots + a_{id}t_j \leq b_i$$

- one where $a_{j1} < 0$, called lower bound:

$$l_j : \; a_{j1}t_1 + \cdots + a_{jd}t_d \leq b_j.$$

Algorithm-DarkShadow

- Consider a polyhedron K:

$$\begin{cases} a_{11}t_1 + \cdots + a_{1d}t_d & \leq b_1 \\ & \vdots \\ a_{m1}t_1 + \cdots + a_{md}t_d & \leq b_m \end{cases}$$

- We aim at eliminating t_1, so we consider 2 inequalities in t_1.
- one where $a_{i1} > 0$, called upper bound:

$$l_i : \ a_{i1}t_1 + \cdots + a_{id}t_j \leq b_i$$

- one where $a_{j1} < 0$, called lower bound:

$$l_j : \ a_{j1}t_1 + \cdots + a_{jd}t_d \leq b_j.$$

Fourier-Motzkin elimination: real projection r_{ij}:

$$-a_{j1}(a_{i2}t_2 + \cdots + a_{id}t_d) + a_{i1}(a_{j2}t_2 + \cdots + a_{jd}t_d) \leq -a_{j1}b_i + a_{i1}b_j.$$

Algorithm-DarkShadow

- Consider a polyhedron K:

$$\begin{cases} a_{11} t_1 + \cdots + a_{1d} t_d & \leq b_1 \\ \vdots & \\ a_{m1} t_1 + \cdots + a_{md} t_d & \leq b_m \end{cases}$$

- We aim at eliminating t_1, so we consider 2 inequalities in t_1.
  - one where $a_{i1} > 0$, called upper bound:

$$l_i : \; a_{i1} t_1 + \cdots + a_{id} t_j \leq b_i$$

  - one where $a_{j1} < 0$, called lower bound:

$$l_j : \; a_{j1} t_1 + \cdots + a_{jd} t_d \leq b_j.$$

Fourier-Motzkin elimination: real projection r_{ij}:

$$-a_{j1}(a_{i2} t_2 + \cdots + a_{id} t_d) + a_{i1}(a_{j2} t_2 + \cdots + a_{jd} t_d) \leq -a_{j1} b_i + a_{i1} b_j.$$

Omega test: dark projection d_{ij} (translates r_{ij} towards the center of K):

$$-a_{j1}(a_{i2} t_2 + \cdots + a_{id} t_d) + a_{i1}(a_{j2} t_2 + \cdots + a_{jd} t_d) \leq -a_{j1} b_i + a_{i1} b_j - (a_{i1} - 1)(-a_{j1} - 1).$$

Algorithm-DarkShadow

- Consider a polyhedron K:

$$\begin{cases} a_{11}t_1 + \cdots + a_{1d}t_d & \leq b_1 \\ \vdots & \\ a_{m1}t_1 + \cdots + a_{md}t_d & \leq b_m \end{cases}$$

- We aim at eliminating t_1, so we consider 2 inequalities in t_1.
  - one where $a_{i1} > 0$, called upper bound:

    $$l_i : \; a_{i1}t_1 + \cdots + a_{id}t_j \leq b_i$$

  - one where $a_{j1} < 0$, called lower bound:

    $$l_j : \; a_{j1}t_1 + \cdots + a_{jd}t_d \leq b_j.$$

Fourier-Motzkin elimination: real projection r_{ij}:

$$-a_{j1}(a_{i2}t_2 + \cdots + a_{id}t_d) + a_{i1}(a_{j2}t_2 + \cdots + a_{jd}t_d) \leq -a_{j1}b_i + a_{i1}b_j.$$

Omega test: dark projection d_{ij} (translates r_{ij} towards the center of K):

$$-a_{j1}(a_{i2}t_2 + \cdots + a_{id}t_d) + a_{i1}(a_{j2}t_2 + \cdots + a_{jd}t_d) \leq -a_{j1}b_i + a_{i1}b_j - (a_{i1} - 1)(-a_{j1} - 1).$$

Lemma

For any integer point $(t_2, \ldots, t_d) \in \mathbb{Z}^{d-1}$ satisfying d_{ij}, there exists at least one integer $t_1 \in \mathbb{Z}$ satisfying both l_i and l_j.

Algorithm-DarkShadow

- Consider a polyhedron K:

$$\begin{cases} a_{11}t_1 + \cdots + a_{1d}t_d & \leq b_1 \\ & \vdots \\ a_{m1}t_1 + \cdots + a_{md}t_d & \leq b_m \end{cases}$$

- We aim at eliminating t_1, so we consider 2 inequalities in t_1.
  - one where $a_{i1} > 0$, called upper bound:

$$l_i : \; a_{i1}t_1 + \cdots + a_{id}t_j \leq b_i$$

  - one where $a_{j1} < 0$, called lower bound:

$$l_j : \; a_{j1}t_1 + \cdots + a_{jd}t_d \leq b_j.$$

Fourier-Motzkin elimination: real projection r_{ij}:

$$-a_{j1}(a_{i2}t_2 + \cdots + a_{id}t_d) + a_{i1}(a_{j2}t_2 + \cdots + a_{jd}t_d) \leq -a_{j1}b_i + a_{i1}b_j.$$

Omega test: dark projection d_{ij} (translates r_{ij} towards the center of K):

$$-a_{j1}(a_{i2}t_2 + \cdots + a_{id}t_d) + a_{i1}(a_{j2}t_2 + \cdots + a_{jd}t_d) \leq -a_{j1}b_i + a_{i1}b_j - (a_{i1} - 1)(-a_{j1} - 1).$$

Lemma

For any integer point $(t_2, \ldots, t_d) \in \mathbb{Z}^{d-1}$ satisfying d_{ij}, there exists at least one integer $t_1 \in \mathbb{Z}$ satisfying both l_i and l_j.

Note that this property does not apply to r_{ij}.

Algorithm-DarkShadow

- Consider a polyhedron K:

$$\begin{cases} a_{11}t_1 + \cdots + a_{1d}t_d & \leq b_1 \\ \vdots & \\ a_{m1}t_1 + \cdots + a_{md}t_d & \leq b_m \end{cases}$$

- We aim at eliminating t_1, so we consider 2 inequalities in t_1.
  - one where $a_{i1} > 0$, called upper bound:

$$l_i : \ a_{i1}t_1 + \cdots + a_{id}t_j \leq b_i$$

  - one where $a_{j1} < 0$, called lower bound:

$$l_j : \ a_{j1}t_1 + \cdots + a_{jd}t_d \leq b_j.$$

Fourier-Motzkin elimination: real projection r_{ij}:

$$-a_{j1}(a_{i2}t_2 + \cdots + a_{id}t_d) + a_{i1}(a_{j2}t_2 + \cdots + a_{jd}t_d) \leq -a_{j1}b_i + a_{i1}b_j.$$

Omega test: dark projection d_{ij} (translates r_{ij} towards the center of K):

$$-a_{j1}(a_{i2}t_2 + \cdots + a_{id}t_d) + a_{i1}(a_{j2}t_2 + \cdots + a_{jd}t_d) \leq -a_{j1}b_i + a_{i1}b_j - (a_{i1} - 1)(-a_{j1} - 1).$$

Let $S^{<t_1}$ consist of the inequalities defining K without t_1.

- Combining all the real projections with $S^{<t_1}$ $\Rightarrow$ real shadow
- Combining all the dark projections with $S^{<t_1}$ $\Rightarrow$ dark shadow

Example

Continue with above polyhedron K_1:

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

Example

Continue with above polyhedron K_1:

$$\begin{cases} 3t_1 - 2t_2 + t_3 \le 7 \\ -2t_1 + 2t_2 - t_3 \le 12 \\ -4t_1 + t_2 + 3t_3 \le 15 \\ -t_2 \le -25 \end{cases}$$

Real Shadow R_1 w.r.t. t_1:

$$\begin{cases} 2t_2 - t_3 \le 50 \\ -5t_2 + 13t_3 \le 73 \\ -t_2 \le -25 \end{cases}.$$

Dark Shadow D_1 w.r.t. t_1:

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

Example

Continue with above polyhedron K_1:

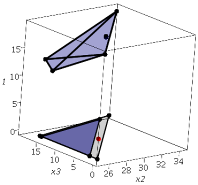
$$\begin{cases} 3t_1 - 2t_2 + t_3 \leq 7 \\ -2t_1 + 2t_2 - t_3 \leq 12 \\ -4t_1 + t_2 + 3t_3 \leq 15 \\ -t_2 \leq -25 \end{cases}$$

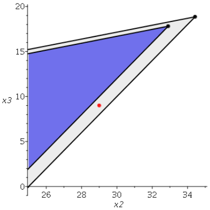
Real Shadow R_1 w.r.t. t_1:

$$\begin{cases} 2t_2 - t_3 \leq 50 \\ -5t_2 + 13t_3 \leq 73 \\ -t_2 \leq -25 \end{cases}.$$

Dark Shadow D_1 w.r.t. t_1:

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





$(t_2, t_3) = (29, 9) \in R_1 \setminus D_1$ does not extend to an integer point of K_1.
Indeed, plugging $(t_2, t_3) = (29, 9)$ into K_1 yields $\frac{37}{2} \leq t_1 \leq \frac{56}{3}$.

Algorithm-Grey shadow

Procedure 3- GreyShadow($\mathbf{Mt} \leq \mathbf{v}$)

Technical details

Check the hyperplans which may contain integer points in the grey shadow ($R_1 \setminus D_1$):

- for each upper bound $l_i : a_{i1}t_1 + \cdots + a_{id}t_d \leq b_i$ and lower bound $l_j : a_{j1}t_1 + \cdots + a_{jd}t_d \leq b_j$ of t_1 in K_1, let $B = \lfloor \frac{-a_{i1}a_{j1} - a_{i1} + a_{j1}}{a_{i1}} \rfloor$;

Algorithm-Grey shadow

Procedure 3- GreyShadow($\mathbf{Mt} \leq \mathbf{v}$)

Technical details

Check the hyperplans which may contain integer points in the grey shadow ($R_1 \setminus D_1$):

- for each upper bound $l_i : a_{i1}t_1 + \cdots + a_{id}t_d \leq b_i$ and lower bound $l_j : a_{j1}t_1 + \cdots + a_{jd}t_d \leq b_j$ of t_1 in K_1, let $B = \lfloor \frac{-a_{i1}a_{j1} - a_{i1} + a_{j1}}{a_{i1}} \rfloor$;
- for k from 0 to B, let $K_1' := K_1 \cap \{a_{j1}t_1 + \cdots + a_{jd}t_d = b_j - k\}$; apply IntegerNormalize to K_1' yields one grey shadow part of K_1;

Algorithm-Grey shadow

Procedure 3- GreyShadow($\mathbf{Mt} \leq \mathbf{v}$)

Technical details

Check the hyperplans which may contain integer points in the grey shadow $(R_1 \setminus D_1)$:

- for each upper bound $l_i : a_{i1}t_1 + \cdots + a_{id}t_d \leq b_i$ and lower bound $l_j : a_{j1}t_1 + \cdots + a_{jd}t_d \leq b_j$ of t_1 in K_1, let $B = \lfloor \frac{-a_{i1}a_{j1} - a_{i1} + a_{j1}}{a_{i1}} \rfloor$;
- for k from 0 to B, let $K_1' := K_1 \cap \{a_{j1}t_1 + \cdots + a_{jd}t_d = b_j - k\}$; apply IntegerNormalize to K_1' yields one grey shadow part of K_1;
- (disjoint) let $K_1 := K_1 \cap \{-a_{j1}l_i + a_{i1}l_j \leq -(a_{i1} - 1)(-a_{j1} - 1)\}$, repeat.

Algorithm-Grey shadow

Procedure 3- GreyShadow($\mathbf{Mt} \le \mathbf{v}$)

Technical details

Check the hyperplans which may contain integer points in the grey shadow ($R_1 \setminus D_1$):

- for each upper bound $l_i : a_{i1}t_1 + \cdots + a_{id}t_d \le b_i$ and lower bound $l_j : a_{j1}t_1 + \cdots + a_{jd}t_d \le b_j$ of t_1 in K_1, let $B = \lfloor \frac{-a_{i1}a_{j1} - a_{i1} + a_{j1}}{a_{i1}} \rfloor$;
- for k from 0 to B, let $K_1' := K_1 \cap \{a_{j1}t_1 + \cdots + a_{jd}t_d = b_j - k\}$; apply IntegerNormalize to K_1' yields one grey shadow part of K_1;
- (disjoint) let $K_1 := K_1 \cap \{-a_{j1}l_i + a_{i1}l_j \le -(a_{i1} - 1)(-a_{j1} - 1)\}$, repeat.

Lemma

L: the maximum absolute value of a coefficient in K_1;
m: the number of inequalities representing K_1.
Then, the number of grey shadow parts of K_1 w.r.t. t_1 is at most mL.

Algorithm-GreyShadow

- For the polyhedron K_1:

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

Algorithm-GreyShadow

- For the polyhedron K_1:

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

- For upper bound $3t_1 - 2t_2 + t_3 \le 7$,
lower bound $-2t_1 + 2t_2 - t_3 \le 12$,
we have
$B := \lfloor (-a_{j1}a_{i1} + a_{j1} - a_{i1})/a_{i1} \rfloor = 0$.

Algorithm-GreyShadow

- For the polyhedron K_1:
$$\begin{cases} 3t_1 - 2t_2 + t_3 \leq 7 \\ -2t_1 + 2t_2 - t_3 \leq 12 \\ -4t_1 + t_2 + 3t_3 \leq 15 \\ -t_2 \leq -25 \end{cases},$$
- For upper bound $3t_1 - 2t_2 + t_3 \leq 7$,
lower bound $-2t_1 + 2t_2 - t_3 \leq 12$,
we have
$B := \lfloor(-a_{j1}a_{i1} + a_{j1} - a_{i1})/a_{i1}\rfloor = 0$.
- Set $K_1' := \begin{cases} -2t_1 + 2t_2 - t_3 = 12 \\ 3t_1 - 2t_2 + t_3 \leq 7 \\ -4t_1 + t_2 + 3t_3 \leq 15 \\ -t_2 \leq -25 \end{cases}$

Algorithm-GreyShadow

- For the polyhedron K_1:

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

- For upper bound $3t_1 - 2t_2 + t_3 \le 7$, lower bound $-2t_1 + 2t_2 - t_3 \le 12$, we have $B := \lfloor(-a_{j1}a_{i1} + a_{j1} - a_{i1})/a_{i1}\rfloor = 0$.

- Set $K_1' := \begin{cases} -2t_1 + 2t_2 - t_3 = 12 \\ 3t_1 - 2t_2 + t_3 \le 7 \\ -4t_1 + t_2 + 3t_3 \le 15 \\ -t_2 \le -25 \end{cases}$

- Apply IntegerSolve to K_1', we get

$$\begin{cases} t_1 = t_4 \\ t_2 = t_5 + 1 \\ t_3 = -2t_4 + 2t_5 + 1 \end{cases} \quad \begin{cases} t_4 \le 8 \\ -10t_4 + 7t_5 \le 11 \\ -t_5 \le -24 \end{cases}$$

Algorithm-GreyShadow

- For the polyhedron K_1:

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

- For upper bound $3t_1 - 2t_2 + t_3 \leq 7$, lower bound $-2t_1 + 2t_2 - t_3 \leq 12$, we have $B := \lfloor (-a_{j1}a_{i1} + a_{j1} - a_{i1})/a_{i1} \rfloor = 0$.

- Set $K_1' := \begin{cases} -2t_1 + 2t_2 - t_3 = 12 \\ 3t_1 - 2t_2 + t_3 \leq 7 \\ -4t_1 + t_2 + 3t_3 \leq 15 \\ -t_2 \leq -25 \end{cases}$

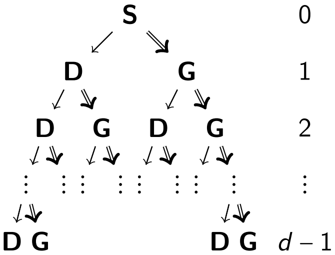
- Apply IntegerSolve to K_1', we get

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

- Let $K_1 := \begin{cases} 3t_1 - 2t_2 + t_3 \leq 7 \\ -2t_1 + 2t_2 - t_3 \leq 12 \\ -4t_1 + t_2 + 3t_3 \leq 15 \\ -t_2 \leq -25 \\ 2t_2 - t_3 \leq 48 \end{cases}$

  Choose another pair of upper bound and lower bound, repeat.

Outline of the algorithm



- **S**: input system, **D**: dark shadow, **G**: grey shadow.
- →: one path of dark shadow, ⇒: many pathes of grey shadow.
- With **N** being either **D** or **G**, each path in the tree has the form

  $\mathbf{S} \to \mathbf{N}_1 \to \mathbf{N}_2 \to \cdots \to \mathbf{N}_r$ for some positive integer $r \leq d - 1$.
- For example, the leftmost path is

$$\mathbf{S} \to \mathbf{D}_1 \to \mathbf{D}_2 \to \cdots \to \mathbf{D}_r$$

Plan

Summary

- We have presented an algorithm for computing the integer points of a polyhedron, based on the Omega test procedure proposed by W. Pugh.

Summary

- We have presented an algorithm for computing the integer points of a polyhedron, based on the Omega test procedure proposed by W. Pugh.
- This is done by decomposing the input polyhedron into simpler polyhedra, each of them with at least one integer point.

Summary

- We have presented an algorithm for computing the integer points of a polyhedron, based on the Omega test procedure proposed by W. Pugh.
- This is done by decomposing the input polyhedron into simpler polyhedra, each of them with at least one integer point.
- We improve it by making use of Hermite normal form and controlling the size of the intermediate coefficients.