ZPolyTrans: a library for computing and enumerating integer transformations of Z-Polyhedra

Rachid Seghir

(Vincent Loechner because the french embassy did not deliver a visa to him)

Department of Computer Science
University of Batna, Algeria
seghir@univ-batna.dz

January 23, 2012

Outline

1. Introduction
   - What is ZPolyTrans ?
   - Motivation
2. A bit of theory
   - Integer transformations and Presburger formulas
   - Existential variables elimination
   - Counting integer points in unions of parametric $\mathbb{Z}$-polytopes
3. Related work
4. Demonstration

What is ZPolyTrans ?

- *ZPolyTrans* is a program written in C for:
  - computing integer affine transformations of parametric $\mathbb{Z}$-Polytopes,
  - counting integer points in the result of such transformations (unions of parametric $\mathbb{Z}$-polytopes).
- Based on *PolyLib*: for doing polyhedral and matrix operations.
- Based on *Barvinok* library: for counting integer points in a parametric polytope.
- Available on `http://zpolytrans.gforge.inria.fr`

Motivation

Code transformation, optimization and parallelization techniques
$\rightarrow$ computing and enumerating the integer affine transformations of parametric $\mathbb{Z}$-polyhedra.

- Array linearization for hardware design [Turjan et al. 2005],
- Cache access optimization [Ghosh et al. 1999, D'Alberto 2001, Loechner et al. 2002],
- Memory size computation [Zhao and Malik 2000, Zhu et al. 2006],
- Data distribution for NUMA-machines [Heine and Slowik 2000],
- ...

Example

Consider two loop nests accessing an array A.

```
for i = 0 to n do
   for j = i+1 to n do
      A[4*i+2*j] = ...

for k = 0 to n do
   for l = 0 to n do
      A[k+l] = ...
```

Question: which array elements are accessed?
ZPolyTrans computes:

- the accessed array elements as a union of three $\mathbb{Z}$-polytopes: $\{2 \leq x \leq 6n - 8 \wedge x \text{ even}\} \cup \{x = 6n - 4 \wedge n \geq 1 \wedge x \text{ even}\} \cup \{0 \leq x \leq 2n \wedge x \in \mathbb{Z}\}$,
- the number of accessed elements as a piecewise quasi-polynomial, equal to 1 when $n = 0$, 3 when $n = 1$ and $4n - 2$ when $n \geq 2$.

Integer transformations of $\mathbb{Z}$-polyhedra and Presburger formulas

Let $\mathcal{Z} = P_{\mathbf{p}} \cap L$ be a parametric $\mathbb{Z}$-polytope:
$P_{\mathbf{p}} = \left\{\mathbf{x} \in \mathbb{Q}^d \mid A\mathbf{x} \geq B\mathbf{p} + \mathbf{c}\right\}$ a parametric rational polytope,
$L = \left\{A'\mathbf{z} + \mathbf{c}' \mid \mathbf{z} \in \mathbb{Z}^d\right\}$ a lattice.

$$\text{An affine function } T: \begin{array}{ccc} \mathbb{Z}^d & \rightarrow & \mathbb{Z}^k \\ \mathbf{x} & \mapsto & \mathbf{x}' = A''\mathbf{x} + \mathbf{c}'' \end{array}$$

The transformation of $\mathcal{Z}$ by T is a Presburger formula

$$T(\mathcal{Z}) = \Big\{\mathbf{x}' \in \mathbb{Z}^k \mid \exists \mathbf{x}, \mathbf{z} \in \mathbb{Z}^d, \quad A\mathbf{x} \geq B\mathbf{p} + \mathbf{c}, \mathbf{x} = A'\mathbf{z} + \mathbf{c}', \mathbf{x}' = A''\mathbf{x} + \mathbf{c}''\Big\}$$

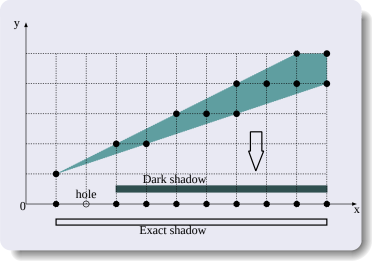
Existential variable elimination

The elimination of the existential variables is done in two steps:

1. Some existential variables can be eliminated by removing equalities implying them (the result is $\mathbb{Z}$-polytope).
2. Recursively eliminating the remaining existential variables from the $\mathbb{Z}$-polytope produced at the first step.
    - rewrite it as a set of lower and upper bounds on the variable to be eliminated $l(\mathbf{x}) \leq \beta z, \alpha z \leq u(\mathbf{x})$
    - the result is given by the intersection of the integer projections of all its pairs of lower and upper bounds.

The integer projection of a pair of bounds

- The projection of a pair of bounds is given in the form: dark shadow $\cup$ {exact shadow $\cap$ sub-lattices of hyperplanes}
- Projection of the pair of bounds $\{x + 2 \leq 3y,\ 2y \leq x + 1\}$

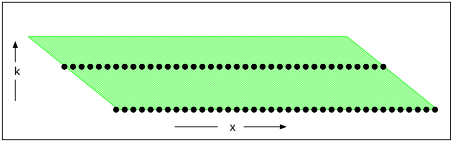


Example of the existential variables elimination

- Elimination of the existential variables from
$$P = \{x \in \mathbb{Z} \mid \exists(i,j,k) \in \mathbb{Z}^3 : \; 1 \le i \le p+4 \;\wedge\; 1 \le j \le 5 \;\wedge\; 3 \le 3k \le 2p \;\wedge\; x = 3i + 4j + 6k + 1\}$$

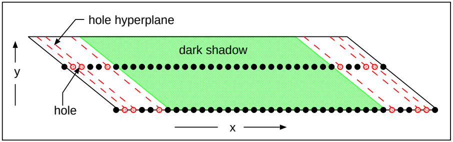
- After removing the equality
$$\left\{ x \in \mathbb{Z} \;\middle|\; \exists(i',k) \in \mathbb{Z}^2 : \begin{array}{rcl} -3x - 2k & \le 4i' \le & -3x - 2k + p + 3 \;\wedge \\ -2x - 3k - 6 & \le 3i' \le & -2x - 3k - 2 \;\wedge \\ 3 & \le 3k \le & 2p \end{array} \right\}$$

- Result of the rational elimination of i' (when $p = 4$).



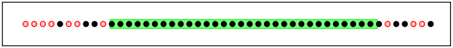
Example of the existential variables elimination (2)

- Result of the integer elimination of i' (when $p = 4$).



When p = 4

- Result of the integer elimination of k (when $p = 4$).



- Number of integer points in the projection (for $p \geq 2$)
  - Integer projection: $\mathcal{E}(p) = 7p + [14, 10, 12]_p$
  - $\neq$ rational projection: $\mathcal{E}(p) = 7p + 20$

Counting integer points in unions of parametric $\mathbb{Z}$-polytopes

- The integer projection of a parametric $\mathbb{Z}$-polytope may result in a non-disjoint union of parametric $\mathbb{Z}$-polytopes.
- To compute the number of integer points in such a union,
  - apply the inclusion-exclusion principle to the original set (rather than computing its disjoint union)
  - call to *Barvinok* library to calculate the Ehrhart quasi-polynomial corresponding to each $\mathbb{Z}$-polytope and their intersections (if non empty).
- The results are finally combined (addition/substration) into a single quasi-polynomial.

Related work

Many approaches have been proposed:

- The work of W. Pugh [Pugh, 1994] on *integer* affine projections based on the Fourier-Motzkin variable elimination.
- The theoretical rational generating function based approach [Barvinok and Woods, 2002]; [Verdoolaege and Woods, 2005]; [Koeppe et al., 2008].
- The weak quantifier elimination approach [Lasaruk and Sturm 2007].
- The $\mathbb{Z}$-polyhedral model [Gautam and Rajopadhye, 2007]; [Iooss and Rajopadhye, 2012].
- The work of [Verdoolaege et al. 2005] based on applying rewriting rules and solving a parametric integer linear programming problem, implemented in *Barvinok* library.

Demonstration

To the demo!