A library to manipulate Z-polyhedron in image representation

Guillaume Iooss, Sanjay Rajopadhye

Colorado State University

January 23, 2012

Motivation: the polyhedral model

- *Polyhedral model:* mathematical framework widely used for program analysis/transformation.

$\Rightarrow$ For example, works perfectly to represent regular loop nest.

Motivation: the polyhedral model

- *Polyhedral model:* mathematical framework widely used for program analysis/transformation.

⇒ For example, works perfectly to represent regular loop nest.

- Irregular loop nest (`if` conditions, modulos, non-unit-stride loops...): this model does not apply directly.

⇒ We can still deal with these situations (by adding extra dimensions), but less practical.

Motivation: the polyhedral model

- *Polyhedral model:* mathematical framework widely used for program analysis/transformation.

⇒ For example, works perfectly to represent regular loop nest.

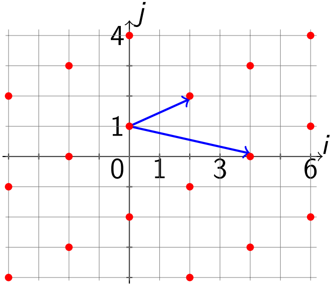
- Irregular loop nest (`if` conditions, modulos, non-unit-stride loops...): this model does not apply directly.

⇒ We can still deal with these situations (by adding extra dimensions), but less practical.

- *Z-polyhedron:* mathematical object that extends integer polyhedron.
  ⇒ Using them is more convenient to deal with such cases.

Affine Lattice

- **Affine Lattice:** $\mathcal{L} = \{L.z + l | z \in \mathbb{Z}^n\} \subset \mathbb{Z}^m$, L and l integer.
- Example:



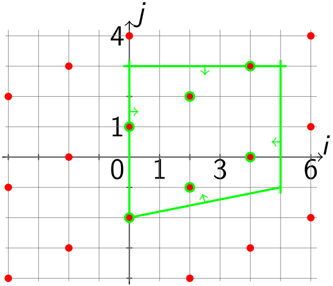
$$L = \begin{bmatrix} 4 & 2 \\ -1 & 1 \end{bmatrix}$$

$$l = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

- *Canonical form:* $\begin{bmatrix} 1 & 0 \\ l & L \end{bmatrix}$ is in HNF and L is full-column rank.
- *Stability properties:* Intersection, difference (infinite and finite), image/preimage by an integer affine function.

Z-polyhedra

- **Z-polyhedron:** Intersection between an integer polyhedron $\mathcal{P}$ and an affine lattice $\mathcal{L}$: $\mathcal{Z} = \mathcal{P} \cap \mathcal{L}$.
- Example:



- *Stability properties:*
  - Intersection, difference, preimage by an integer affine function
  - Image by an unimodular integer affine function is a Z-polyhedron
  - Image by a non-unimodular integer affine function is a union of Z-polyhedra

Representations of a Z-polyhedron

- Two possible representations of a Z-polyhedron:
  - Intersection representation: $\mathcal{Z} = \mathcal{L} \cap \mathcal{P}$ (definition)
  - Image representation: After some rewriting $\mathcal{Z} = \{L.z + l | z \in \mathcal{P}_c\}$ with $\mathcal{P}_c = \{z | Q.z + q \geq 0 \wedge A.z + b = 0\}$

Representations of a Z-polyhedron

- Two possible representations of a Z-polyhedron:
  - Intersection representation: $\mathcal{Z} = \mathcal{L} \cap \mathcal{P}$ (definition)
  - Image representation: After some rewriting $\mathcal{Z} = \{L.z + l | z \in \mathcal{P}_c\}$ with $\mathcal{P}_c = \{z | Q.z + q \geq 0 \wedge A.z + b = 0\}$
- Image representation correspond to the definition of a *Linear Bounded Lattice (LBL)*. However, all LBL is not a Z-polyhedron. (example: $\{i + 3j | 0 \leq j \leq i \leq 3\} = [|0, 12|] - \{8, 10, 11\}$).
- *LeVerge's sufficient condition:*

  $\mathcal{Z} = \{L.z + l | z \in \mathcal{P}_c\}$ is a Z-polyhedron if $Ker \begin{pmatrix} L \\ Q_0 \end{pmatrix} \subset Ker(Q)$,

  with $Ker(Q_0)$ the context of the coordinate polyhedron $\mathcal{P}_c$.

Algorithms

- *Implemented algorithms:* described in [Gautam & Rajopadhye, 2007].
  - Intuitively, same algorithms that for the intersection representation.
  - Slight modifications done to manipulate Z-polyhedron not in canonical form (condition of full-dimensionality on $\mathcal{P}_c$).
  - Because of that, proposed image algorithm does not work anymore.

Algorithms

- *Implemented algorithms:* described in [Gautam & Rajopadhye, 2007].
  - Intuitively, same algorithms that for the intersection representation.
  - Slight modifications done to manipulate Z-polyhedron not in canonical form (condition of full-dimensionality on $\mathcal{P}_c$).
  - Because of that, proposed image algorithm does not work anymore.

- *Image algorithm:* described in [Seghir, Loechner & Meister, 2010].
  - *Idea:* Write the image as a Presburger set and eliminate the existential variables one by one (using equalities, then inequalities).
  - Algorithm translated in image representation.
  - Our current implementation: no heuristic to select which existential variable to eliminate first. Not fully optimized.

Related work

- *ZPolyTrans* (cf previous presentation): `http://zpolytrans.gforge.inria.fr`
  Also a library to manipulate Z-polyhedron, but in C and based on the intersection representation.
- *Omega* is a library that solves feasibility of a Pressburger set.
- *ISL* is a polyhedral library. It handles Z-polyhedra by using existentially quantified dimensions.

Implementation

- This library has been developed in Java.
  Source code: `http://www.cs.colostate.edu/AlphaZsvn/Development/trunk/mde/`
- *Polymodel* used as an underlying polyhedral library (IRISA):
  - Interface to manipulate polyhedron
  - Currently implemented interface: ISL

Tool example

Comparison with integer polyhedra

Operations	Polyhedron	Z-polyhedron
Intersection	$O(N_{constraints})$	$O(n^4.log(\|\|L\|\|))$ (HNF)
Difference	$O(N^2_{constraints})$	$O(n^4.log(\|\|L\|\|))$ (HNF)
Preimage	$O(n^3)$	$O(n^4.log(\|\|L\|\|))$ ($\cap$)
Image (unimodular)	$O(n^3)$	$O(n^3)$ (matrix mult)
Image (non unimodular)	-	Exponential

Comparison with integer polyhedra

Operations	Polyhedron	Z-polyhedron
Intersection	$O(N_{constraints})$	$O(n^4.log(\|\|L\|\|))$ (HNF)
Difference	$O(N^2_{constraints})$	$O(n^4.log(\|\|L\|\|))$ (HNF)
Preimage	$O(n^3)$	$O(n^4.log(\|\|L\|\|))$ ($\cap$)
Image (unimodular)	$O(n^3)$	$O(n^3)$ (matrix mult)
Image (non unimodular)	-	Exponential

- Complexity of Z-polyhedral operations for the 2 representations are asymptotically the same:
  - Intersection/difference: intersection representation faster.
  - Image (unimodular/non unimodular): image representation faster.

Future work

- *About the library:* Implement the missing operations:
  - Going back from the image to the intersection representation,
  - Getting the canonical form / making the coordinate polyhedron full-dimensional,
  - Equality test.

$\Rightarrow$ Need advanced polyhedral feature that are not (yet?) in *PolyModel*.

- Some algorithms can be improved (ex: number of generated Z-polyhedron for a difference).

Future work

- *About the library:* Implement the missing operations:
  - Going back from the image to the intersection representation,
  - Getting the canonical form / making the coordinate polyhedron full-dimensional,
  - Equality test.

⇒ Need advanced polyhedral feature that are not (yet?) in *PolyModel*.

- Some algorithms can be improved (ex: number of generated Z-polyhedron for a difference).

- *About Z-polyhedra:* For program analysis, how does it compared in term of speed with the polyhedral model?

Thanks for listening

- Do you have any questions?