The Parma Polyhedra Library

Roberto BAGNARA

Elisa RICCI

Enea ZAFFANELLA

University of Parma, Italy

Patricia M. HILL

University of Leeds, United Kingdom

`http://www.cs.unipr.it/ppl/`

PLAN OF THE TALK

① Convex Polyhedra: What and Why

② The Double Description Method by Motzkin et al.

③ DD Pairs and Minimality

④ Advantages of the Dual Description Method

⑤ Handling Not Necessarily Closed Polyhedra

⑥ What Are the Generators of NNC Polyhedra

⑦ Minimization of NNC Polyhedra

⑧ The Parma Polyhedra Library

⑨ PPL Features

⑩ Conclusion

Convex Polyhedra: What and Why

What?

- ➜ regions of $\mathbb{R}^n$ bounded by a finite set of hyperplanes.

Why? Solving Classical Data-Flow Analysis Problems!

- ➜ array bound checking;
- ➜ compile-time overflow detection;
- ➜ loop invariant computations and loop induction variables.

Why? Verification of Concurrent and Reactive Systems!

- ➜ synchronous languages;
- ➜ linear hybrid automata (roughly, FSMs with time requirements);
- ➜ systems based on temporal specifications.

And Again: Many Other Applications. . .

- ➜ inferring argument size relationships in logic programs;
- ➜ termination inference for Prolog: Made in Réunion!

The Double Description Method by Motzkin et al.

Constraint Representation

- If $\boldsymbol{a} \in \mathbb{R}^n$, $\boldsymbol{a} \neq \boldsymbol{0}$, and $b \in \mathbb{R}$, the linear inequality constraint $\langle \boldsymbol{a}, \boldsymbol{x} \rangle \geq b$ defines a closed affine half-space.
- All *closed polyhedra* can be expressed as the conjunction of a finite number of such constraints.

Generator Representation

- If $\mathcal{P} \subseteq \mathbb{R}^n$, a *point of* $\mathcal{P}$ is any $\boldsymbol{p} \in \mathcal{P}$.
- If $\mathcal{P} \subseteq \mathbb{R}^n$ and $\mathcal{P} \neq \varnothing$, a vector $\boldsymbol{r} \in \mathbb{R}^n$ such that $\boldsymbol{r} \neq \boldsymbol{0}$ is a *ray of* $\mathcal{P}$ iff for each point $\boldsymbol{p} \in \mathcal{P}$ and each $\lambda \in \mathbb{R}_+$, we have $\boldsymbol{p} + \lambda \boldsymbol{r} \in \mathcal{P}$.
- All *closed polyhedra* can be expressed as

$$\left\{\, R\boldsymbol{\rho} + P\boldsymbol{\pi} \in \mathbb{R}^n \;\middle|\; \boldsymbol{\rho} \in \mathbb{R}^r_+, \boldsymbol{\pi} \in \mathbb{R}^p_+, \textstyle\sum_{i=1}^{p} \pi_i = 1 \,\right\}$$

  where $R \in \mathbb{R}^{n \times r}$ is a matrix having rays of the polyhedron as columns and $P \in \mathbb{R}^{n \times p}$ has points of the polyhedron for its columns.

The Double Description Method by Motzkin et al. 4

The Double Description Method (cont'd)

Constraint Representation

- ➜ Special case: $n = 0$ and $\mathcal{P} = \varnothing$.
- ➜ The equality constraint $\langle \boldsymbol{a}, \boldsymbol{x} \rangle = b$ defines an affine hyperplane. . .
  - ➜ . . . that is equivalent to the pair $\langle \boldsymbol{a}, \boldsymbol{x} \rangle \geq b$ and $\langle -\boldsymbol{a}, \boldsymbol{x} \rangle \geq -b$.
- ➜ If $\mathcal{C}$ is a finite set of constraints we call it *a system of constraints* and write $\mathrm{con}(\mathcal{C})$ to denote the polyhedron it describes.

Generator Representation

- ➜ Note: $P = \varnothing$ if and only if $\mathcal{P} = \varnothing$.
- ➜ Note: points are not necessarily vertices and rays are not necessarily extreme.
- ➜ We call $\mathcal{G} = (R, P)$ *a system of generators* and write $\mathrm{gen}(\mathcal{G})$ to denote the polyhedron it describes.

DD Pairs and Minimality

Representing a Polyhedron Both Ways

- ➜ Let $\mathcal{P} \subseteq \mathbb{R}^n$. If $\mathrm{con}(\mathcal{C}) = \mathrm{gen}(\mathcal{G}) = \mathcal{P}$, then $(\mathcal{C}, \mathcal{G})$ is said to be a DD pair for $\mathcal{P}$.

Minimality of the Representations

- ➜ $\mathcal{C}$ is in minimal form if there does not exist $\mathcal{C}' \subset \mathcal{C}$ such that $\mathrm{con}(\mathcal{C}') = \mathcal{P}$;
- ➜ $\mathcal{G} = (R, P)$ is in minimal form if there does not exist $\mathcal{G}' = (R', P') \neq \mathcal{G}$ such that $R' \subseteq R$, $P' \subseteq P$ and $\mathrm{gen}(\mathcal{G}') = \mathcal{P}$;
- ➜ the DD pair $(\mathcal{C}, \mathcal{G})$ is in minimal form if $\mathcal{C}$ and $\mathcal{G}$ are both in minimal form.

But, wait a minute…

…why keeping two representations for the same object?

ADVANTAGES OF THE DUAL DESCRIPTION METHOD

Some Operations Are More Efficiently Performed on Constraints

➜ Intersection is implemented as the union of constraint systems.

➜ Adding constraints (of course).

➜ Relation polyhedron-generator (subsumes or not).

Some Operations Are More Efficiently Performed on Generators

➜ Convex polyhedral hull (poly-hull): union of generator systems.

➜ Adding generators (of course).

➜ Projection (i.e., removing dimensions).

➜ Relation polyhedron-constraint (disjoint, intersects, includes . . .).

➜ Finiteness (boundedness) check.

➜ Time-elapse.

Some Operations Are More Efficiently Performed with Both

➜ Inclusion and equality tests.

➜ Widening.

Further Advantages of the Dual Description Method

The Principle of Duality

- ➜ Systems of constraints and generators enjoy a quite strong and useful duality property.
- ➜ Very roughly speaking:
  - ➜ the constraints of a polyhedron are (almost) the generators of the *polar* of the polyhedron;
  - ➜ the generators of a polyhedron are (almost) the constraints of the polar of the polyhedron;
  - ➜ the polar of the polar of a polyhedron is the polyhedron itself.

$\Longrightarrow$ Computing constraints from generators is the same problem as computing generators from constraints.

The Algorithm of Motzkin-Chernikova-Le Verge

- ➜ Solves both problems yielding a minimized system. . .
- ➜ . . . and can be implemented so that the source system is also minimized in the process.

Handling Not Necessarily Closed Polyhedra

Strict Inequalities and NNC Polyhedra

- ➜ If $\boldsymbol{a} \in \mathbb{R}^n$, $\boldsymbol{a} \neq \mathbf{0}$, and $b \in \mathbb{R}$, the linear strict inequality constraint $\langle \boldsymbol{a}, \boldsymbol{x} \rangle > b$ defines an open affine half-space;
- ➜ when strict inequalities are allowed in the system of constraints we have polyhedra that are not necessarily closed: NNC polyhedra.

Encoding NNC Polyhedra as C Polyhedra

- ➜ call $\mathbb{P}_n$ and $\mathbb{CP}_n$ the sets of all NNC and closed polyhedra, respectively;
- ➜ each NNC polyhedron $\mathcal{P} \in \mathbb{P}_n$ can be embedded into a closed polyhedron $\mathcal{R} \in \mathbb{CP}_{n+1}$:
- ➜ the additional dimension of the vector space, usually labeled by the letter ϵ, encodes the topological closedness of each affine half-space in the constraint description for $\mathcal{P}$.

Embedding $\mathbb{P}_n$ Into $\mathbb{CP}_{n+1}$

If $\mathcal{P} \in \mathbb{P}_n$ and $\mathcal{P} = \mathrm{con}(\mathcal{C})$, where

$$\mathcal{C} = \left\{ \langle \boldsymbol{a}_i, \boldsymbol{x} \rangle \bowtie_i b_i \;\middle|\; i \in \{1, \ldots, m\}, \boldsymbol{a}_i \in \mathbb{R}^n, \bowtie_i \in \{\geq, >\}, b_i \in \mathbb{R} \right\},$$

then $\mathcal{R} \in \mathbb{CP}_{n+1}$ is defined by $\mathcal{R} = \mathrm{con}\bigl(\mathrm{con\_repr}(\mathcal{C})\bigr)$, where

$$\begin{aligned} \mathrm{con\_repr}(\mathcal{C}) \stackrel{\mathrm{def}}{=} \bigl\{0 \leq \epsilon \leq 1\bigr\} & \\ \cup \left\{ \langle \boldsymbol{a}_i, \boldsymbol{x} \rangle - 1 \cdot \epsilon \geq b_i \;\middle|\; i \in \{1, \ldots, m\}, \bowtie_i \in \{>\} \right\} & \\ \cup \left\{ \langle \boldsymbol{a}_i, \boldsymbol{x} \rangle + 0 \cdot \epsilon \geq b_i \;\middle|\; i \in \{1, \ldots, m\}, \bowtie_i \in \{\geq\} \right\}. & \end{aligned}$$

What Are the Generators of NNC Polyhedra

- ➜ A fundamental feature of the DD method: the ability to represent polyhedra both by constraints and generators.
- ➜ But what are the generators for NNC polyhedra?
- ➜ From the New Polka manual (s is the ϵ coefficient):

> *Don't ask me the intuitive meaning of $s \neq 0$ in rays and vertices !*

- ➜ From the Polka manual:

> *While strict inequations handling is transparent for constraints [...] the extra dimension added to the variables space is apparent when it comes to generators [...]*
>
> *This makes more difficult to define polyhedra with the only help of generators : one should carefully study the extra vertices with non null `epsilon` coefficients added to constraints defined polyhedra [...]*

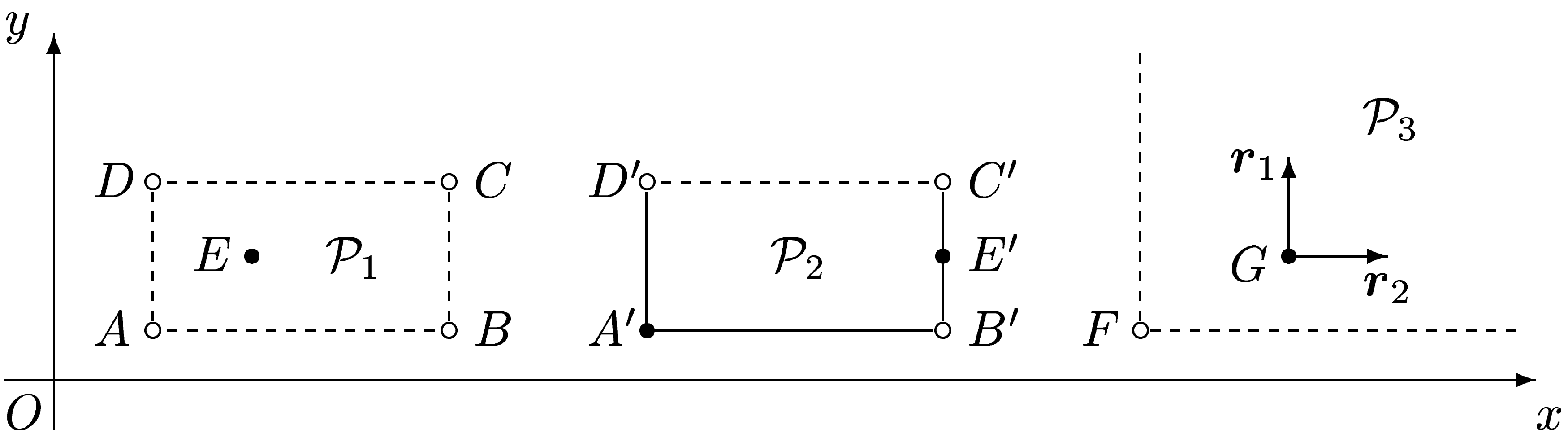
Closure Points to the Rescue

- By decoupling the user interface from the details of the particular implementation, it is possible to provide an intuitive generalization of the concept of generator system.
- The key step is the introduction of a new kind of generators: closure points:
  - a vector $\boldsymbol{c} \in \mathbb{R}^n$ is a *closure point* of $S \subseteq \mathbb{R}^n$ if and only if $\boldsymbol{c} \in \mathbb{C}(S)$.
- Characterization of closure points for NNC polyhedra:
  - a vector $\boldsymbol{c} \in \mathbb{R}^n$ is a closure point of the NNC polyhedron $\mathcal{P} \in \mathbb{P}_n$ if and only if $\mathcal{P} \neq \varnothing$ and for every point $\boldsymbol{p} \in \mathcal{P}$ and $\lambda \in \mathbb{R}$ such that $0 < \lambda < 1$, it holds $\lambda\boldsymbol{p} + (1 - \lambda)\boldsymbol{c} \in \mathcal{P}$.
- All *NNC polyhedra* can be expressed as

$$\left\{\, R\boldsymbol{\rho}+P\boldsymbol{\pi}+C\boldsymbol{\gamma} \in \mathbb{R}^n \;\middle|\; \boldsymbol{\rho} \in \mathbb{R}^r_+, \boldsymbol{\pi} \in \mathbb{R}^p_+, \boldsymbol{\pi} \neq \boldsymbol{0}, \boldsymbol{\gamma} \in \mathbb{R}^c_+, \textstyle\sum_{i=1}^p \pi_i + \sum_{i=1}^c \gamma_i = 1 \,\right\}$$

where $R \in \mathbb{R}^{n \times r}$ is a matrix having rays of the polyhedron as columns, $P \in \mathbb{R}^{n \times p}$ has points of the polyhedron for its columns, and $C \in \mathbb{R}^{n \times c}$ has closure points of the polyhedron for its columns.

Some Examples



Minimization of NNC Polyhedra

- ➜ The problem: in no way does minimization of the representation in $\mathbb{CP}_{n+1}$ imply minimization of the NNC polyhedron in $\mathbb{P}_n$.
- ➜ There are examples where a “minimized” representation has more than half of the constraints that are redundant.
- ➜ This causes both efficiency and usability problems:
  - ➜ the client application must distinguish between the real constraints/generators and the surrounding noise.
- ➜ A solution to this problem is presented in a forthcoming paper:

  Roberto Bagnara, Elisa Ricci, Enea Zaffanella, and Patricia M. Hill
  Possibly Not Closed Convex Polyhedra and the Parma Polyhedra Library
  Technical Report, Department of Mathematics, University of Parma, Italy
  Available at `http://www.cs.unipr.it/`

The Parma Polyhedra Library

- ➜ A collaborative project started in January 2001 at the Department of Mathematics of the University of Parma.
- ➜ It aims at becoming a truly professional library for the handling (not necessarily closed) rational convex polyhedra.
- ➜ Targeted at abstract interpretation and computer-aided verification.
- ➜ Free software released under the GNU General Public License.

Why Yet Another Library? Some Limitations of Existing Ones:

- ➜ data-structures employed cannot grow/shrink dynamically;
- ➜ possibility of overflow, underflow and rounding errors;
- ➜ unsuitable mechanisms for error detection, handling and recovery;
  - ➜ (cannot reliably resume computation with an alternative method, e.g., by reverting to an interval-based approximation).
- ➜ Several existing libraries are free, but they do not provide documentation for the interfaces and code that is adequate for an outsider to make improvements with any real confidence.

PPL Features

Portability Across Different Computing Platforms

- ➜ written in standard C++;
- ➜ but the the client application needs not be written in C++.

Absence of Arbitrary Limits

- ➜ arbitrary precision integer arithmetic for coefficients and coordinates;
- ➜ all data structures can expand automatically (in amortized constant time) to any dimension allowed by the available virtual memory.

Complete Information Hiding

- ➜ the internal representation of constraints, generators and systems thereof need not concern the client application;
- ➜ implementation devices such as the *positivity constraint* are invisible from outside;
- ➜ all the matters regarding the ϵ-representation encoding of NNC polyhedra are also invisible from outside.

PPL Features: Hiding Pays

Expressivity

- ➜ '`X + 2*Y + 5 >= 7*Z`' and '`ray(3*X + Y)`' is valid syntax both for the C++ and the Prolog interfaces;
- ➜ we expect the planned Objective Caml and Mercury interfaces to be as friendly as these;
- ➜ even the C interface refers to concepts like linear expression, constraint and constraint system
  - ➜ (not to their possible implementations such as vectors and matrices).

Failure Avoidance and Detection

- ➜ illegal objects cannot be created easily;
- ➜ the interface invariants are systematically checked.

Efficiency

- ➜ can systematically apply incremental and lazy computation techniques.

PPL Features: Laziness and Incrementality

Dual Description

- ➜ we may have a constraint system, a generator system, or both;
- ➜ in case only one is available, the other is recomputed only when it is convenient to do so.

Minimization

- ➜ the constraint (generator) system may or may not be minimized;
- ➜ it is minimized only when convenient.

Saturation Matrices

- ➜ when both constraints and generators are available, some computations record here the relation between them for future use.

Sorting Matrices

- ➜ for certain operations, it is advantageous to sort (lazily and incrementally) the matrices representing constraints and generators.

PPL Features: Support for Robustness

```
void complex_function(PH& ph1, const PH& ph2 ...) {
  try {
    start_timer(max_time_for_complex_function);
    complex_function_on_polyhedra(ph1, ph2 ...);
    stop_timer();
  }
  catch (Exception& e) { // Out of memory or timeout...
    BoundingBox bb1, bb2;
    ph1.shrink_bounding_box(bb1);
    ph2.shrink_bounding_box(bb2);
    complex_function_on_bounding_boxes(bb1, bb2 ...);
    ph1 = Polyhedron(bb1);
  }
}
```

CONCLUSION

- ➜ Convex polyhedra are the basis for several abstractions used in static analysis and computer-aided verification of complex and sometimes mission critical systems.
- ➜ For that purposes an implementation of convex polyhedra must be firmly based on a clear theoretical framework and written in accordance with sound software engineering principles.
- ➜ In this talk we have presented some of the most important ideas that are behind the Parma Polyhedra Library.
- ➜ The Parma Polyhedra Library is free software released under the GPL: code and documentation can be downloaded and its development can be followed at `http://www.cs.unipr.it/ppl/`.