Ranking Functions for Linear-Constraint Loops

Amir Ben-Amram

FOPARA 2013

Ranking Functions for Loops

Example 1 (GCD program):

```
while (x > 1, y > 1)
  if  x<y then (x,y) := (x, y-x)
          else  (x,y) := (y, x-y)
```

Here $f(x,y) = x+y$ is a ranking function

- non-negative in all (enabled) states
- strictly decreasing
- proves termination

Ranking Functions for Loops

Example 2:

```
while (x > 1, y > 1)
  if ...   then (x,y) := (x,     y-1)
           else  (x,y) := (x-1, f(x,y))
```

Here $f(x,y) = \langle x,y \rangle$ is a ranking function

- strictly decreasing **lexicographically**

Outline

- Ranking functions for linear-constraint loops: a well-understood tool in termination and resource analysis
- In this talk: **linear** ranking functions &

  lexicographic linear ranking functions

- For each type:
  - review one major technique
  - proceed to recent work (B. & Genaim)

Why Ranking Functions?

- Termination of **imperative programs**
- Termination of **functional and logic programs**
- Complexity analysis (execution time, etc)
  - The ranking function bounds the number of iterations / length of call chain
- Loop parallelization
  - how to schedule computations that depend on previous results

Loops

- A loop:

```
while (1< x+y+z) {
   x := x + 1;
   y := y – 1;
   z := z – 1;
}
```

Guard

update

- Consider numerical variables (most often integer)
- The "update" is straight-line code

Loops

- A loop can have multiple paths

```
while (1< x+y+z) {
   if (y>0)
            x := x + 1;    y := y – 1;
    else
            z := z – 1;
}
```

Loops

- A loop in linear constraint representation:

while ($y \le x$, $x+y \ge 1$)

$x' - x \le 0$, $y' - y + 2x \le 1$

next state

while ($B\vec{x} \le \vec{b}$) do

$$A\begin{pmatrix}\vec{x}\\ \vec{x}'\end{pmatrix} \le \vec{a}$$

- Update may be non-deterministic, and can model non-linear operations, e.g., integer division

$x := (2*x)/5$

$5x' <= 2x$, $5x' >= 2x - 4$

Loops

- A multiple path loop:

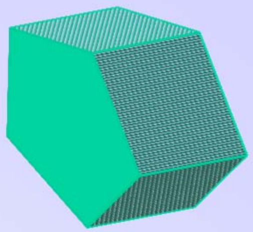
loop

$$(B_1\vec{x} \leq \vec{b}) \wedge A_1 \begin{pmatrix} \vec{x} \\ \vec{x}' \end{pmatrix} \leq \vec{a}_1$$

$$\mid \ (B_2\vec{x} \leq \vec{b}) \wedge A_2 \begin{pmatrix} \vec{x} \\ \vec{x}' \end{pmatrix} \leq \vec{a}_2$$

States & Transitions

- State: $\vec{x} = (x_1, \ldots, x_n) \in \mathbb{Z}^n$ (typically)
  - perhaps of $\mathbb{Q}^n$ (as an over approximation easier to solve)
- Transition: $\vec{x}'' = (\vec{x}\ \vec{x}') \quad = (x_1, \ldots, x_n, x'_1, \ldots, x'_n)$
- Each path of a loop is a subset of $\mathbb{Z}^{2n}$ ($\mathbb{Q}^{2n}$)
- May be written $A''\ \vec{x}'' \leq c''$
  - this specifies a convex polyhedron $\mathcal{Q} \subset \mathbb{Q}^{2n}$,
    the transition polyhedron
  - the set of its integer points is $\mathrm{I}(\mathcal{Q})$
  - we can analyze the same loop "over rationals" or "over integers"



Ranking Functions with values in $\mathbb{Q}$

f: $\mathbb{Q}^n \mapsto \mathbb{Q}$

$\vec{x}'' \in \mathcal{Q} \Rightarrow f(\vec{x}) \geq 0$ (Bounded)

$\vec{x}'' \in \mathcal{Q} \Rightarrow f(\vec{x}) - f(\vec{x}') \geq 1$ (Descending)

Essentially a function into ω

r.f. over integers:
consider only $\vec{x}'' \in \mathrm{I}(\mathcal{Q})$

Linear Ranking Functions (LRF)

- $f(\vec{x}) = \vec{\lambda} \cdot \vec{x} + \lambda_0$ for rational $\vec{\lambda}, \lambda_0$

$$x \in \mathcal{Q} \Rightarrow \vec{\lambda} \cdot \vec{x} + \lambda_0 \geq 0 \quad \text{(B)}$$

$$\text{and} \quad \vec{\lambda} \cdot \vec{x} - \vec{\lambda} \cdot \vec{x}' \geq 1 \quad \text{(D)}$$

r.f. over integers:
$\vec{x}'' \in \mathrm{I}(\mathcal{Q})$

The Linar Ranking Function problem [over $\mathbb{Z}$]

Given a loop $\mathcal{Q}$ (as constraints), find a LRF (satisfying (B),(D)) for $\mathcal{Q}$ [for $\mathrm{I}(\mathcal{Q})$]

LRF by Linear Programming (LP)

$$x \in \mathcal{Q} \Rightarrow \vec{\lambda} \cdot \vec{x} + \lambda_0 \geq 0 \quad \text{(B)}$$

and $\vec{\lambda} \cdot \vec{x} - \vec{\lambda} \cdot \vec{x}' \geq 1$ (D)

linear constraints

linear constraints

LRF by Linear Programming (LP)

- Sohn and van Gelder (1991)
- Feautrier (1992)
- Colón and Sipma (2001)
- Podelski and Rybalchenko (2004)
- Mesnard and Serebrenik (2008)

Logic programming

Farkas' Lemma

parallelization

imperative programs

see also survey by Bagnara et al. (2012)

The "Farkas based" solution

How to draw an implication from inequalities

$\mu_1 \cdot$	$a_1 x$	$+$	$b_1 y$	$\geq$	c_1
$\mu_2 \cdot$	$a_2 x$	$+$	$b_2 y$	$\geq$	c_2

Farkas' lemma says that all implications are formed this way

The "Farkas based" solution

transition polyhedron $\mathcal{Q}$ - $A'' \vec{x}'' \leq c''$

should imply (B)+(D) - $\vec{\lambda} \cdot \vec{x} + \lambda_0 \geq 0$

$$\vec{\lambda} \cdot \vec{x} - \vec{\lambda} \cdot \vec{x}' \geq 1$$

The “Farkas based” solution

transition polyhedron $\mathcal{Q}$ - $$-A'' \vec{x}'' \geq -c''$$

should imply (B)+(D) - $$\vec{\lambda} \cdot \vec{x} + \lambda_0 \geq 0$$

$$\vec{\lambda} \cdot \vec{x} - \vec{\lambda} \cdot \vec{x}' \geq 1$$

We just have to find the Farkas multipliers

- which is an LP problem
- every solution of which yields a LRF
- it specifies the set of all LRFs for $\mathcal{Q}$

Multipath loops

- An LRF has to be valid for all the paths
- A conjunction of LP problems is an LP problem

Solved!

The “Farkas based” solution is:

- polynomial time
- complete, in some sense

Completeness

- We call a method **complete** if it is guaranteed to find a LRF, when one exists
- The LP based methods are complete

  over the rationals
- (B)+(D) have to hold over $\mathcal{Q}$, not just $\mathrm{I}(\mathcal{Q})$

Understanding the integer LRF problem

(B. & Genaim, POPL '13)

- The decision problem is coNP-complete
- we show:
  - coNP hardness
  - inclusion in coNP
  - synthesis algorithm in exponential time
  - PTIME-solvable classes (DBMs? octagons? TVPI?)

Examples: Integer vs Rational

```
while (x ≥ y , x+y ≥ 1) do
          y := y+1-2x
```

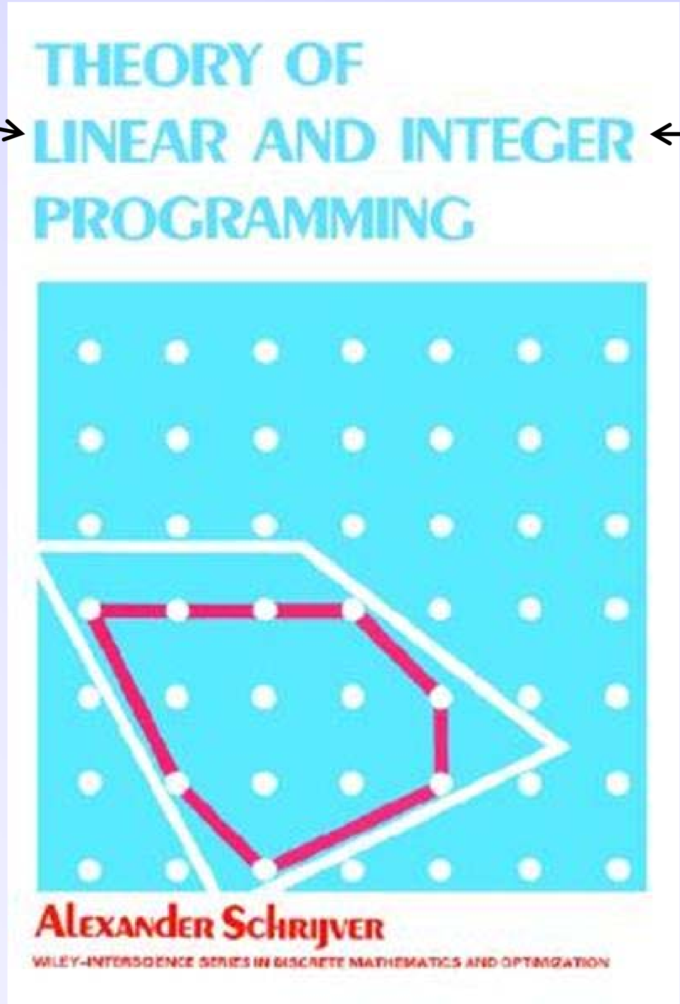
- Here $f(x,y) = x+y$ is a ranking function
  - for integers but not for rationals! (consider $(\frac{1}{2},\frac{1}{2})$)

```
while (x ≥ y , x+y ≥ 1, 4x ≥ 1) do
          y := y+1-2x
```

- Here $f(x,y) = 2(x+y)$ is a ranking function over the rationals, while for integers $x+y$ is valid

The source of hardness

Easy
(PTIME)



hard
(NPC)

Hardness proof

- NP-hard problem:

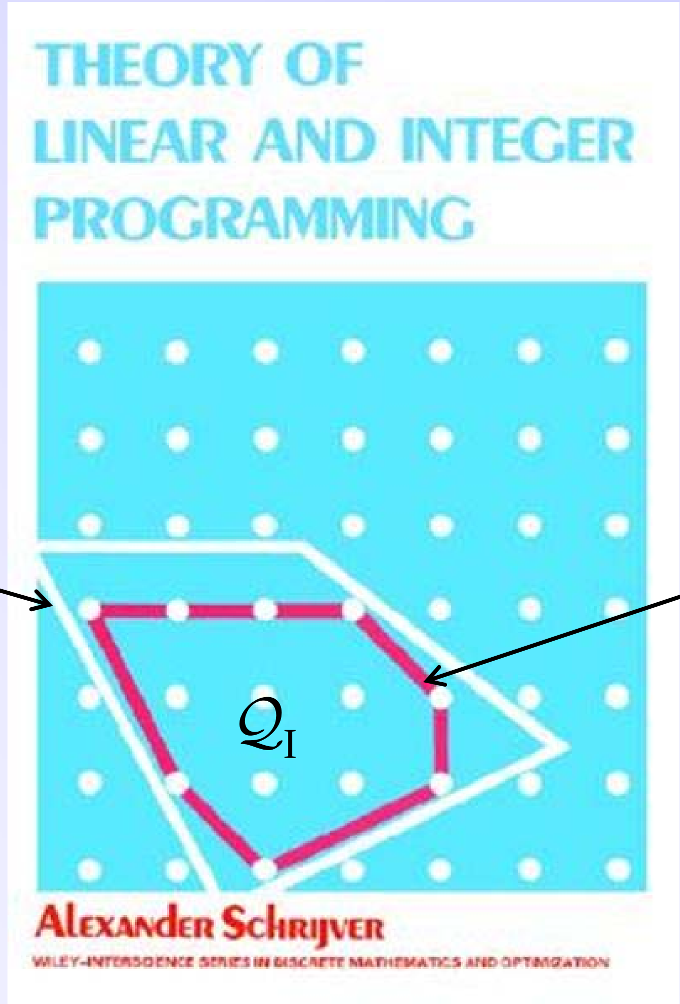
  Given constraints: $B\vec{x} \leq b$, is there any integer solution?

- Reduction to termination problem:

$$\text{while } (B\vec{x} - \vec{z} \leq b \;,\; \vec{z} \geq 0 \,) \;\text{ do}$$
$$\vec{x}' = \vec{x} \;\;,\;\; \vec{z'} = 0$$

This loop has a LRF $\Leftrightarrow$
there is no integer solution to $B\vec{x} \leq b$

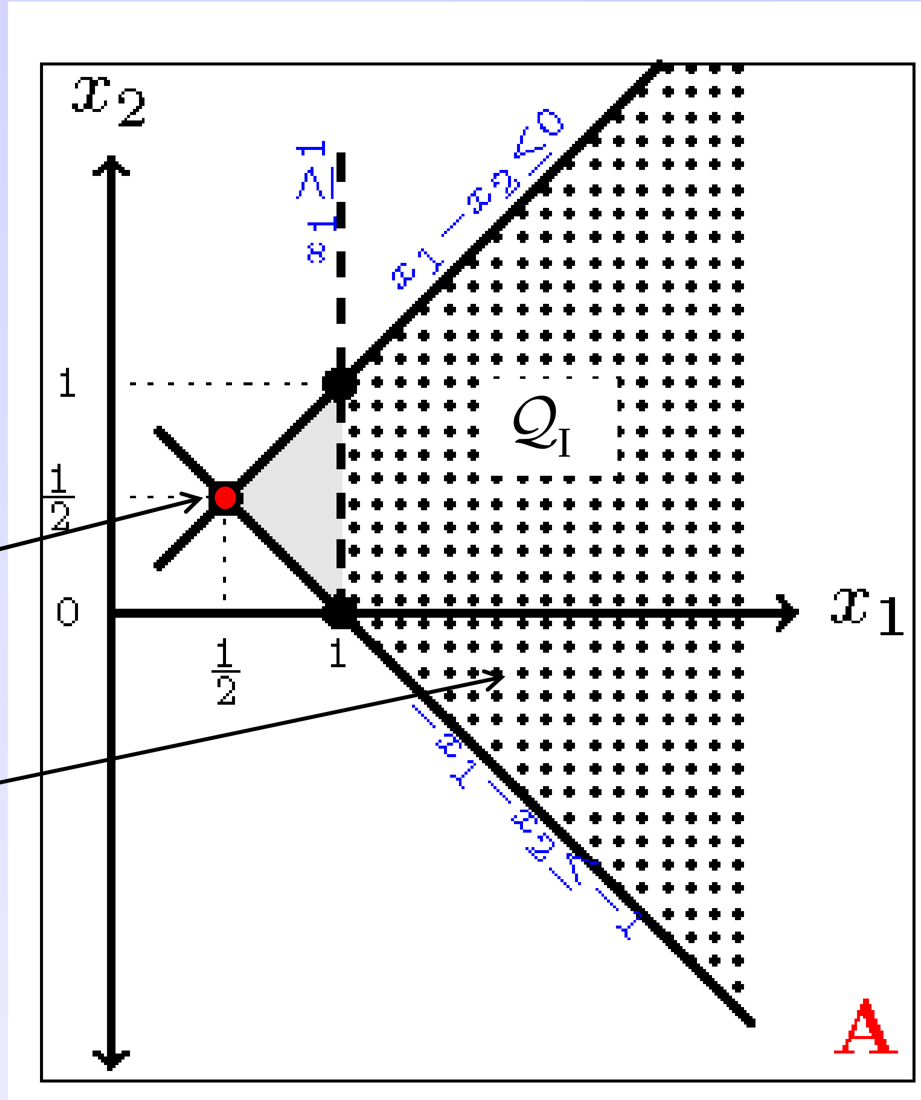
Clue to solution



polyehdron $\mathcal{Q}$

integer hull
= convex
hull of $\mathrm{I}(\mathcal{Q})$

while ($x_2 \le x_1$, $x_1+x_2 \ge 1$) do
$x_2' = x_2+1-2x_1$, $x_1' = x_1$



obnoxious point

$\mathcal{Q}_{\mathrm{I}}$ has a LRF

Solution

- If $\mathrm{I}(\mathcal{Q})$ has a LRF, $\mathcal{Q}_\mathrm{I}$ has one.
- Compute a representation of $\mathcal{Q}_\mathrm{I}$
- Find LRF over rationals
- Complete solution, exponential complexity

Inclusion in coNP

THM

The LRF existence problem over integers is coNP-complete.

⇔ There are polynomially-checkable witnesses to **non-existence** of a LRF

Witnesses

Consider a candidate function $f(\vec{x}) = \vec{\lambda} \cdot \vec{x} + \lambda_0$

Point $\vec{x}'' = (\vec{x}\ \vec{x}') \in Q$ **witnesses against** f if f fails to satisfy (B) or (D)

$$\vec{\lambda} \cdot \vec{x} + \lambda_0 \geq 0 \qquad \text{(B)}$$

$$\vec{\lambda} \cdot \vec{x} - \vec{\lambda} \cdot \vec{x}' \geq 1 \qquad \text{(D)}$$

Let $W(\vec{x}'') = \{\vec{\lambda} | \vec{x}'' \text{ witnesses against } \vec{\lambda}\}$

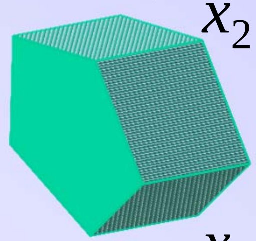
$$W(X) = \cup_{x \in X} W(\vec{x}'')$$

no LRF $\Leftrightarrow \exists X . W(X) = \mathbb{Q}^{n+1}$

space of coefficient vectors

Where to look for witnesses?

1. If $\mathcal{Q}$ is bounded:



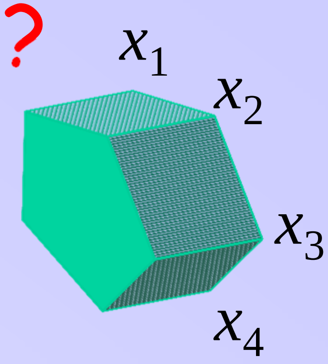
$$\mathcal{Q} = \text{conv. hull}(x_1, \dots, x_V)$$

If $f(\vec{x})$ is not a RF, it must fail on one of the vertices.

(If all vertices satisfy (B),(D), then so does any convex combination.)

- no LRF $\Leftrightarrow W(\{x_1, \dots, x_V\}) = \mathbb{Q}^{n+1}$

Where to look for witnesses?



no LRF $\Leftrightarrow WS(x_1, \dots, x_V) = \mathbb{Q}^{n+1}$

$\Leftrightarrow$ (B),(D) fail on $x_1, \dots, x_V$

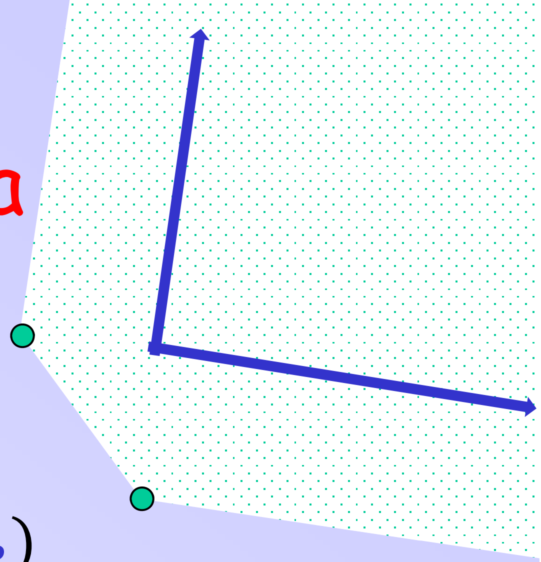
A corollary of Farkas' Lemma (found in Schrijver):

"if a set of linear constraints on $\mathbb{Q}^{n+1}$ has no solution, there is a subset of $n+2$ of them that doesn't "

- We have a **small** witness set
- The bit-size of the witnesses is polynomial

(theorem on relation between bit-size of constraints and of vertices.)

Unbounded polyhedra



Every polyhedron can be represented as

$$\mathcal{Q} = \text{conv.hull}(x_1, \ldots, x_V) + \text{cone}(y_1, \ldots, y_R)$$

A ray y is added to a point $\mathrm{x} \in \text{conv.hull}(x_1, \ldots, x_V)$ to form points $\mathrm{x} + a\mathrm{y}, \quad a \geq 0$

y witnesses against f if f fails to satisfy (B') or (D')

$\vec{\lambda} \cdot \vec{y} \geq 0$	(B')	$\vec{\lambda} \cdot \vec{y} - \vec{\lambda} \cdot \vec{y}' \geq 0$	(D')

Inclusion in coNP

Witnesses can be found among the vertices and rays

$\Rightarrow$

There are witness sets of polynomial cardinality and bit-size, and they are polynomially checkable

Cases we can solve in polynomial time

Basic observation:

Our problem becomes tractable if either

$\mathcal{Q}$ is an integral polyhedron ($\mathcal{Q}_I = \mathcal{Q}$) [CKRW '10]

or we have a specialized procedure to compute $\mathcal{Q}_I$

Totally Unimodular matrices

- Matrix A is totally unimodular if each subdeterminant of A is in {0,$\pm$1}
- A is TUM $\Rightarrow$

  the polyhedron $\{x | Ax \leq b\}$ is integral

Example:

difference-bound constraints

yield TUM constraint matrices

$$x_i - x_j \leq d$$
$$\pm x_i \leq d$$

Difference-bound constraints are also special cases of:

- Octagons

$$\pm x_i \pm x_j \leq d$$
$$\pm x_i \leq d$$

- Two-variable per inequality (TVPI) constraints

$$a x_i + b x_j \leq d$$

Octagons can be non-integral and their integer hull can be hard to compute

Using two-dimensional polyhedra

Harvey (1999) shows how to compute in PTIME the integer hull of any **two-dimensional** polyhedron

More than 2 variables? Try to **decompose** the constraint set into independent constraint-sets

```
while (4x >= 1 & y>=1) do
        5x' <= 2x + 1,
        5x' > 2x – 4,
        y = y'
```

More in our paper

Outline

So far: **linear** ranking functions

- complete solutions, algorithms and complexity

Next: **lexicographic linear** ranking functions

- For each type:
  - review one major technique
  - proceed to recent work (B. & Genaim)

LLRFs

(Lexicographic Linear Ranking Functions)

- Very natural, at least for multipath-loops (and more general control-flow graphs)
- Stricly extend the class of loops (compared to LRFs)
- Known since the dawn of times [Turing 1948]
- Widely used in the termination literature
- Deduction from linear constraints begins with Colón and Sipma (2002); Bradley, Manna and Sipma ('05)

The "Farkas based" solution

Alias et al. [ADFG 2010] :

- Program is a control-flow graph with linear constraints (for simplicity: multipath loop), variables $x_1, \dots, x_n$
- Algorithm generates a LLRF, $\langle \rho_1, \rho_2, \dots, \rho_d \rangle$ (dimension)
- Has polynomial time complexity – based on LP
  - Applies the Farkas technique d times, where $d \leq \#\text{of paths}$ (for a multipath loop)
- Used to bound the no. of transitions
- Guaranteed to find the smallest dimension

- Alias et al. prove completeness

over the rationals

- **Our plan:**
  Solve the integer-restricted problem
- **Our results:**
  - decision problem is coNP-complete
  - synthesis algorithm guarantees smallest dimension
  - we find LLRFs for some single-path loops having no LRF – a bit of a surprise
    - more so because [ADFG] proves it cannot happen
    - the reason is a difference in definitions

$$while(x_1 \geq 0, x_2 \geq 0, x_3 \geq -x_1)\ do$$
$$x'_2 = x_2 - x_1, \quad x'_3 = x_3 + x_1 - 2$$

$\rho(x_1, x_2, x_3) = \langle x_2, x_3 \rangle$

- Think about integers
  - Case 1: x_2 descends

    Case 2: x_2 does not descend

- In [ADFG] this is not allowed, since they would require x_3 to be non-negative at all times
- In our definition, if the 1st component descends, we do not care about the 2nd

The algorithm at a glance

Definition: a **quasi-ranking** function for $\mathcal{Q}$ is

$f: \mathbb{Q}^n \mapsto \mathbb{Q}$ such that

$$\vec{x}'' \in \mathcal{Q} \Rightarrow f(\vec{x}) \geq 0$$

$$\vec{x}'' \in \mathcal{Q} \Rightarrow f(\vec{x}) \geq f(\vec{x}')$$

Note that $f(\vec{x}) = 0$ satisfies it

A function is non-trivial if

$$\exists \vec{x}'' \in \mathcal{Q}\,.\; f(\vec{x}) > f(\vec{x}')$$

To build a LLRF for $\mathcal{Q}$:

- Find a non-trivial quasi-ranking function ρ_1
  (based on the Farkas method for LRF)
- Compute $\mathcal{Q}' = \mathcal{Q} \wedge \{\rho_1(\vec{x}) = \rho_1(\vec{x}')\}$
  This gives the transitions where ρ_1 does **not** descend.
- If $\mathcal{Q}' \neq \emptyset$, proceed recursively.
  - as in last example

$$while(x_1 \geq 0, x_2 \geq 0, x_3 \geq -x_1) \;\; do$$

$$x_2' = x_2 - x_1, \;\; x_3' = x_3 + x_1 - 2$$

Properties of the algorithm:

- Maximum recursion depth is n – because components are linearly independent
- It takes polynomial time (over the rationals)
- To get completeness over the integers, compute the integer hull first
  - PTIME special cases apply
- Optimality of the dimension follows from a judicious selection of the quasi-LRF

LLRFs and the number of steps

[ADFG 2010] use the LLRF ρ to bound the number of steps. The idea:

Since ρ always decreases, the number of steps is bounded by the number of distinct values of $\rho(\vec{x})$

Let $\mathcal{P}$ be the state space (polyhedral invariant computation is used to find it)

$\rho(\mathcal{P})$ is a d-dimensional polyhedron

The number of integer points in $\rho(\mathcal{P})$ is estimated.

Example

Bubble Sort

loop $(0 \le i < n \wedge 0 \le j < n)$

$$i' = i + 1,\ j' = 0,\ n' = n$$

$$|\quad i' = i,\ j' = j + 1,\ n' = n$$

- $\rho(i,j) = \langle n-i, n-j \rangle$ is a ranking function
- $\mathcal{P} = \{0 \le i < n \wedge 0 \le j < n \wedge n = n^{init}\}$
- $\rho(\mathcal{P}) = [1, n^{\mathrm{init}}] \times [1, n^{\mathrm{init}}]$ - quadratic

Examples

Single-path example:

$$while(x_1 \geq 0, x_2 \geq 0, x_3 \geq -x_1)\ do$$
$$x'_2 = x_2 - x_1, \quad x'_3 = x_3 + x_1 - 2$$

- ρ(i,j) = $\langle x_2, x_3 \rangle$ is a ranking function
- $\mathcal{P} = \{0 \leq x_2 \leq x_2^{init},\ x_1 \geq 0, -x_1^{init} \leq x_3 \leq x_3^{init} + x_2^{init}\}$
- ρ($\mathcal{P}$) = $[0, x_2^{init}] \times [-x_1^{init}, x_3^{init} + x_2^{init}]$ - quadratic
- We proved that for certain loops (including all SLC loops with LLRF) a **linear** bound can be found
- In the above loop, $O(\max(x_2^{init}, x_3^{init}))$

Summary

- (Lexicographic) Linear ranking functions are a useful tool and there are solid theoretical results.
- The decision problems are PTIME over rationals, coNP-complete over integers.
- The PTIME solution is sound for the integers, and for certain classes of constraints, it is complete.
- Synthesis algorithms have corresponding efficiency

Bibliography

- Please see our POPL paper and TR !

Demo program - iRankFinder

iRankFinder

Home | Analyzer | Examples | Help | Download

Enter the loop in the corresponding text area, set the required options, and click on Find LRF. The result will be displayed at the bottom of this page.

```
# Example 3.20 - 1st loop

!vars
x1 x2

!pvars
x1' x2'

!path
-x1+x2 <= 0
-x1-x2 <= -1
x1'=x1
x2'=x2-2*x1+1
```

- (•) LINRF(Z)
- ( ) LINRF(Q)

[ ] PTIME	Check PTIME cases
[ ] OCTAGONS	For octagons, compute colusre instead of the integer hull
[x] INTHULL	Compute the integer hull if PTIME cases fail
[ ] iRank	Use the algorithm based on the generator representation (Section 3.4)
[x] VERBOSE	Verbose mode

[Find LRF] [Clear Input] [Clear Output]

Result

```
Not applied yet ...
```