Real Quantifier Elimination in the RegularChains Library

Changbo Chen[1] and Marc Moreno Maza[2]

[1] Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences
[2] ORCCA, University of Western Ontario

August 9, 2014
ICMS 2014, Seoul, Korea

Outline

Outline

Quantifier Elimination

- Input: a prenex formula $PF := (Q_{k+1}x_{k+1} \cdots Q_n x_n)\, F(x_1, \ldots, x_n)$
  - $F(x_1, \ldots, x_n)$: a quantifier free formula over $\mathbb{R}$
  - each Q_i is either $\exists$ or $\forall$.
- Output : a quantifier free formula $SF(x_1, \ldots, x_k)$ such that $SF \Leftrightarrow PF$ holds for all $x_1, \ldots, x_k \in \mathbb{R}$.

Quantifier Elimination (QE)

$(\exists x)(\forall y)\ (ax^2 + bx + c) - (ay^2 + by + c) \geq 0$, where $a, b, c, x, y \in \mathbb{R}$,

for which QE yields

$$(a < 0) \vee (a = b = 0).$$

Quantifier Free Formula (QFF)

$$\neg(y - x^2 > 0 \wedge z^3 - x = 0) \vee (z + xy \geq 0 \wedge x^2 + y^3 \neq 0)$$

Applications of QE

- Geometry theorem proving,
- Stability and bifurcation analysis of dynamical systems (biological systems),
- Control system design,
- Verification of hybrid systems,
- Program verification,
- Nonlinear optimization,
- Automatic parallelization,
- ...

Outline

The RegularChains library in MAPLE

Design goals

- Solving polynomial systems over $\mathbb{Q}$ and $\mathbb{F}_p$, including **parametric** systems and **semi-algebraic** systems.
- Offering tools to manipulate their solutions.
- Organized around the concept of a **regular chain**, accommodating all types of solving and providing space-and-time efficiency.

Features

- Use of types for algebraic structures: `polynomial_ring`, `regular_chain`, `constructible_set`, `quantifier_free_formula`, `regular_semi_algebraic_system`, ...
- Top level commands: `PolynomialRing`, `Triangularize`, `RealTriangularize` `SamplePoints`, ...
- Tool kits: `AlgebraicGeometryTools`, `ConstructibleSetTools`, `MatrixTools`, `ParametricSystemTools`, `FastArithmeticTools`, `SemiAlgebraicSetTools`, ...

The RegularChains library in Maple

Design goals

- Solving polynomial systems over $\mathbb{Q}$ and $\mathbb{F}_p$, including **parametric** systems and **semi-algebraic** systems.
- Offering tools to manipulate their solutions.
- Organized around the concept of a **regular chain**, accommodating all types of solving and providing space-and-time efficiency.

Features

- Use of types for algebraic structures: `polynomial_ring`, `regular_chain`, `constructible_set`, `quantifier_free_formula`, `regular_semi_algebraic_system`, ...
- Top level commands: `PolynomialRing`, `Triangularize`, `RealTriangularize` `SamplePoints`, ...
- Tool kits: `AlgebraicGeometryTools`, `ConstructibleSetTools`, `MatrixTools`, `ParametricSystemTools`, `FastArithmeticTools`, `SemiAlgebraicSetTools`, ...

Solving for the real solutions of polynomial systems

Classical tools

- Isolating the real solutions of zero-dimensional polynomial systems: `SemiAlgebraicSetTools:-RealRootIsolate`
- Real root classification of parametric polynomial systems: `ParametricSystemTools:-RealRootClassification`
- Cylindrical algebraic decomposition of polynomial systems: `SemiAlgebraicSetTools:-CylindricalAlgebraicDecompose`

New tools

- Triangular decomposition of semi-algebraic systems: `RealTriangularize`
- Sampling all connected components of a semi-algebraic system: `SamplePoints`
- Set-theoretical operations on semi-algebraic sets: `SemiAlgebraicSetTools:-Difference`

Solving for the real solutions of polynomial systems

Classical tools

- Isolating the real solutions of zero-dimensional polynomial systems:
  `SemiAlgebraicSetTools:-RealRootIsolate`
- Real root classification of parametric polynomial systems:
  `ParametricSystemTools:-RealRootClassification`
- Cylindrical algebraic decomposition of polynomial systems:
  `SemiAlgebraicSetTools:-CylindricalAlgebraicDecompose`

New tools

- Triangular decomposition of semi-algebraic systems:
  `RealTriangularize`
- Sampling all connected components of a semi-algebraic system:
  `SamplePoints`
- Set-theoretical operations on semi-algebraic sets:
  `SemiAlgebraicSetTools:-Difference`

Outline

The user interface of the QE procedure

We have developed the interface of our QE procedure based on the Logic package of MAPLE. The following MAPLE session shows how to use our procedure.

Example (Davenport-Heintz)

The interface:

```
> f := &E([c]), &A([b, a]), ((a=d) &and (b=c))
            &or ((a=c) &and (b=1)) &implies (a^2=b):
> QuantifierElimination(f);
            false
```

since this actually yields `(d - 1 = 0) &or (d + 1 = 0)`.

The default output of QuantifierElimination is quantifier free formula.

```
> R := PolynomialRing([x, a, b, c]);
  f := &E([x]), a*x^2+b*x+c=0;
  out := QuantifierElimination(f, R);
```

$$R := polynomial\_ring$$

$$f := \&E([x]), x^2\, a + x\, b + c = 0$$

$$out := ((4\, a\, c - b^2 < 0 \text{ \&or } 4\, a\, c - b^2 = 0 \text{ \&and } a < 0) \text{ \&or } 4\, a\, c - b^2 = 0 \text{ \&and } 0 < a) \text{ \&or } (4\, a\, c - b^2 = 0 \text{ \&and } a = 0) \text{ \&and } c = 0$$

Output of QuantifierElimination in extended Tarski formula (I)

```
> f := &E([x]), a*x^2+b*x+c=0;
  out := QuantifierElimination(f,'output'='rootof');
```

$$f := \&E([x]),\ a\,x^2 + b\,x + c = 0$$

$$out := \left(\left(\left(\left(a < 0 \ \&\text{and}\ \frac{1}{4}\,\frac{b^2}{a} \le c \ \&\text{or}\ a = 0 \ \&\text{and}\ b < 0\right) \&\text{or}\ (a = 0 \ \&\text{and}\ b = 0)\ \&\text{and}\ c = 0\right) \&\text{or}\ a = 0 \ \&\text{and}\ 0 < b\right) \&\text{or}\ 0 < a \ \&\text{and}\ c \le \frac{1}{4}\,\frac{b^2}{a}$$

Output of QuantifierElimination in extended Tarski formula (II)

```
> f := &E([y]), y^2+x^2=2;
  out := QuantifierElimination(f, output=rootof);
```

$$f := \&E([y]), x^2 + y^2 = 2$$

$$out := -\sqrt{2} \le x \ \&\text{and}\ x \le \sqrt{2}$$

```
> f := &E([y]), y^4+x^4=2;
  out := QuantifierElimination(f, output=rootof);
```

$$f := \&E([y]), x^4 + y^4 = 2$$

$$out := RootOf\left(\_Z^4 - 2, index = real_1\right) \le x \ \&\text{and}\ x \le RootOf\left(\_Z^4 - 2, index = real_2\right)$$

```
> evalf(op(1, out)); evalf(op(2, out));
```

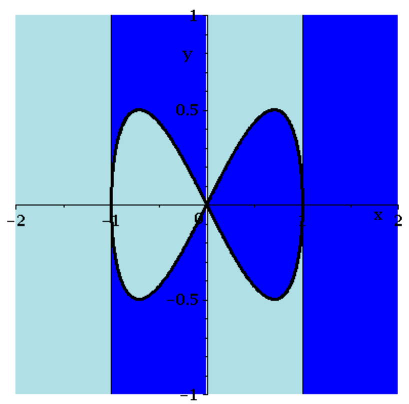
$$-1.189207115 \le x$$

$$x \le 1.189207115$$

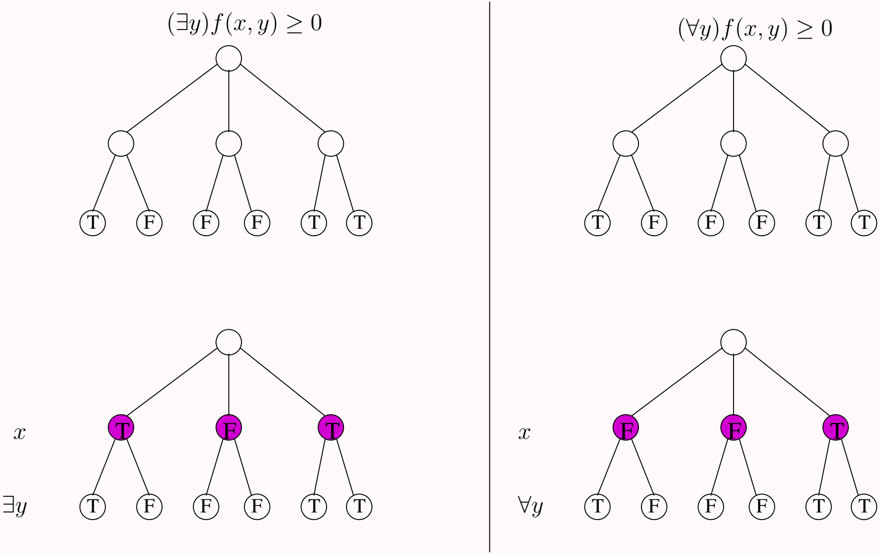
Outline

Cylindrical Algebraic Decomposition (CAD) of $\mathbb{R}^n$

- A CAD of $\mathbb{R}^n$ is a partition $\mathcal{C}$ of $\mathbb{R}^n$ s. t. each cell in $\mathcal{C}$ is a connected semi-algebraic set of $\mathbb{R}^n$ and all cells are cylindrically arranged.
- Two subsets A, B of $\mathbb{R}^n$ are cylindrically arranged if for any $1 \leq k < n$, the projections of A and B on $\mathbb{R}^k$ are equal or disjoint.
- Each cell can be described by a triangular system and all the cell descriptions can be organized as a tree data-structure.



Why CAD supports QE : The main idea



CAD based on regular chains (RC-CAD)

Motivation: potential drawback of Collins' projection-lifting scheme

- The projection operator is a function defined independently of the input system.
- As a result, a strong projection operator (Collins-Hong operator) usually produces much more polynomials than needed.
- A weak projection operator (McCallum-Brown operator) may fail for non-generic cases.

Solution: make case discussion during projection

- Case discussion is common for algorithms computing triangular decomposition.
- At ISSAC'09, we (with B. Xia and L. Yang) introduced case discussion into CAD computation.
- The new method consists of two phases. The first phase computes a complex cylindrical tree (CCT). The second phase decomposes each cell of CCT into its real connected components.

CAD based on regular chains (RC-CAD)

Motivation: potential drawback of Collins' projection-lifting scheme

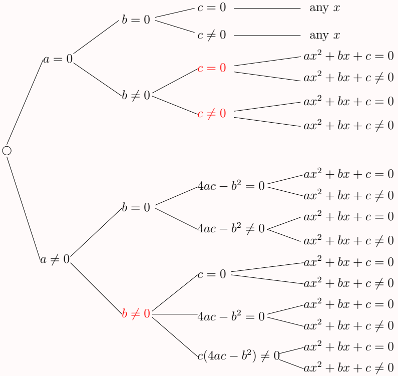
- The projection operator is a function defined independently of the input system.
- As a result, a strong projection operator (Collins-Hong operator) usually produces much more polynomials than needed.
- A weak projection operator (McCallum-Brown operator) may fail for non-generic cases.

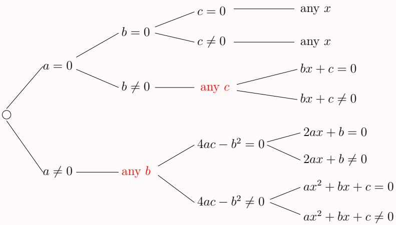
Solution: make case discussion during projection

- Case discussion is common for algorithms computing triangular decomposition.
- At ISSAC'09, we (with B. Xia and L. Yang) introduced case discussion into CAD computation.
- The new method consists of two phases. The first phase computes a complex cylindrical tree (CCT). The second phase decomposes each cell of CCT into its real connected components.

Illustrate PL-CAD and RC-CAD by parametric parabola example

Let $f := ax^2 + bx + c$. Suppose we like to compute a f-sign invariant CAD. The projection factors are $a, b, c, 4ac - b^2, ax^2 + bx + c$. Rethinking PL-CAD in terms of a *complex cylindrical tree*, we get the left tree.





Clearly, RC-CAD (see right tree) computes a smaller tree by *avoiding useless case distinction.*

QE by RC-CAD

Challenges for doing QE by RC-CAD

- RC-CAD has *no global projection factor set* associated to it.
- Instead, it is associated with a complex cylindrical tree. The polynomials in one path of a tree may not be sign invariant above cells derived from a different path of a tree.
- There is *no universal projection operator for RC-CAD.*
- Refining an existing CAD is not straightforward comparing to PL-CAD.

The solution (C. Chen & M., ISSAC 2014)

- Uses an operation introduced in ASCM 2012 (C. Chen & M.) for *refining a complex cylindrical tree* and,
- Adapts C. W. Brown's incremental method for creating *projection-definable* PL-CAD to RC-CAD;
- The approach works with truth-invariant CAD produced in ASCM 2012 and CASC 2014 (with R. Bradford, J. H. Davenport, M. England and D. J. Wilson) for making use of equational constraints.

QE by RC-CAD

Challenges for doing QE by RC-CAD

- RC-CAD has *no global projection factor set* associated to it.
- Instead, it is associated with a complex cylindrical tree. The polynomials in one path of a tree may not be sign invariant above cells derived from a different path of a tree.
- There is *no universal projection operator for RC-CAD.*
- Refining an existing CAD is not straightforward comparing to PL-CAD.

The solution (C. Chen & M., ISSAC 2014)

- Uses an operation introduced in ASCM 2012 (C. Chen & M.) for *refining a complex cylindrical tree* and,
- Adapts C. W. Brown's incremental method for creating *projection-definable* PL-CAD to RC-CAD;
- The approach works with truth-invariant CAD produced in ASCM 2012 and CASC 2014 (with R. Bradford, J. H. Davenport, M. England and D. J. Wilson) for making use of equational constraints.

QE by CAD based on regular chains (RC-QE) : The big picture

Algorithm: QuantifierElimination

- Input: A prenex formula $PF := (Q_{k+1}x_{k+1} \cdots Q_n x_n)FF(x_1, \ldots, x_n)$.
- Output: A solution formula of PF.

Description

1. Let F be the set of polynomials appearing in FF
2. $T :=$ CylindricalDecompose$F$//computes a complex cylindrical tree
3. $RT :=$ MakeSemiAlgebraic(T)//computes a CAD tree
4. AttachTruthValue(FF, RT)//evaluate the truth values of FF at each cell
5. PropagateTruthValue(PF, RT)//get the true values of PF
6. MakeProjectionDefinable(PF, RT)//refine RT until projection definable (not required if the output is allowed to be extended Tarski formula)
7. $SF :=$ GenerateSolutionFormula$_k(RT)$//generate QFF describing true cells in free space

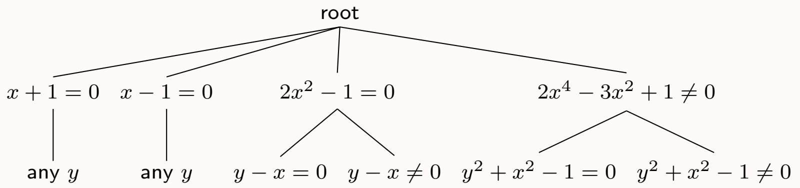
An example

This example illustrates the case that the polynomials in the initial CCT are not enough to express the solution set.

Consider the following QE problem:

$$(\exists y)\ (x^2 + y^2 - 1 = 0) \wedge (x + y < 0) \wedge (x > -1) \wedge (x < 1).$$

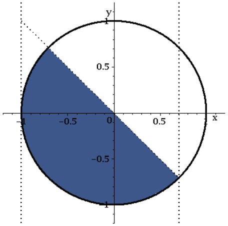
CylindricalDecompose($[x^2 + y^2 - 1 = 0, x + y \neq 0, x \neq -1, x \neq 1]$) computes the following CCT T:



- The CAD of $\mathbb{R}^1$ has the following cells, with blue ones being true cells.

$$(-\infty,-1),-1,(-1,-\frac{\sqrt{2}}{2}),-\frac{\sqrt{2}}{2},(-\frac{\sqrt{2}}{2},\frac{\sqrt{2}}{2}),\frac{\sqrt{2}}{2},(\frac{\sqrt{2}}{2},1),1,(1,+\infty).$$

- The true cells describe the projection of the blue region on the x-axis, which cannot be expressed by the signs of polynomials in the CCT.



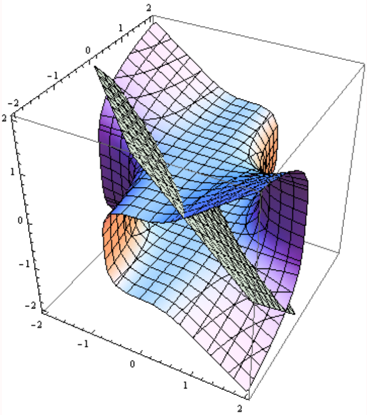
- The cells $-\frac{\sqrt{2}}{2}$ and $\frac{\sqrt{2}}{2}$ is called a conflicting pair, since they have opposite true values and all univariate polynomials in the tree have the same signs at them.
- They are derived from the path $\Gamma := [root, 2x_1^2-1=0]$ of T_1. Refine Γ w.r.t. $\text{diff}(2x_1^2-1,x)$ generates a projection definable CAD, from where we deduce the solution $(x_1<0 \wedge 0<x_1+1) \vee x_1=0 \vee (0<x_1 \wedge 2x_1^2<1)$.

An advanced example

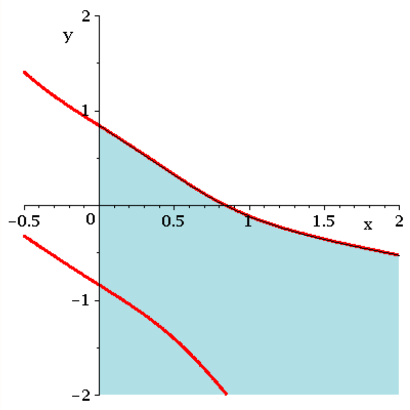
Let $f := 2z^4 + 2x^3y - 1$ and $h = x + y + z$.
Consider the following quantifier elimination problem.

$$\exists(z)(f < 0 \wedge h < 0).$$

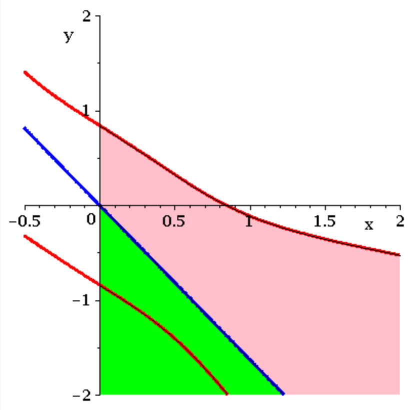
The plots of $f = 0$ and $h = 0$ are depicted in the following figure.



The solution set is the blue region in the following picture, where the red curve is the locus of $p := 2x^4 + 10x^3y + 12x^2y^2 + 8xy^3 + 2y^4 - 1$. The solution set is exactly the set of (x, y) such that $x > 0$ and $y < RealRoot_2(p, y)$. Apparently, this region cannot be described just by the sign of p.



To describe the blue region by a QFF, the derivative of p, namely $q := 10x^3 + 24x^2y + 24xy^2 + 8y^3$, is introduced. The locus of q is the blue curve.
Note that the blue region is the union of the green region ($x > 0 \wedge q < 0$), the blue curve ($x > 0 \wedge q = 0$) and the pink region ($x > 0 \wedge p < 0 \wedge q > 0$).



Outline

Benchmark examples

- The efficiency of the QE procedure directly benefits that of RC-CAD.
- It was shown in ASCM 12 that RC-CAD is competitive to the state of art CAD implementations.
- We illustrate the efficiency of the QE procedure by several examples.

Neither QEPCAD nor Mathematica can solve the examples blood-coagulation-2 and MontesS10 within 1-hour time limit.

Example (blood-coagulation-2)

It takes about 6 seconds.

```
f := &E([x, y, z]), (1/200*x*s*(1 - 1/400*x)
       + y*s*(1 - 1/400*x) - 35/2*x=0)
       &and (250*x*s*(1 - 1/600*y )*(z + 3/250) - 55/2*y=0)
       &and (500*(y + 1/20*x)*(1 - 1/700*z) - 5*z=0);
QuantifierElimination(f);
true
```

Example (MontesS10)

It takes about 26 seconds.

```
f := &E([c2,s2,c1,s1]),
      (r-c1+l*(s1*s2-c1*c2)=0) &and (z-s1-l*(s1*c2+s2*c1)=0)
      &and (s1^2+c1^2-1=0) &and (s2^2+c2^2-1=0);
QuantifierElimination(f);
```

```
      2    2    2
((((-r  - z  + l  - 2 l + 1 = 0) &or

   2    2    2                    2    2    2
((l  - r  - z  - 2 l < -1) &and (-r  - z  + l  + 2 l + 1 = 0))) &or

   2    2    2                        2    2    2
((l  - r  - z  - 2 l < -1) &and (0 < -r  - z  + l  + 2 l + 1))) &or

        2    2    2                  2    2    2
((0 < -r  - z  + l  - 2 l + 1) &and (l  - r  - z  + 2 l < -1))) &or

        2    2    2                     2    2    2
((0 < -r  - z  + l  - 2 l + 1) &and (-r  - z  + l  + 2 l + 1 = 0))
```

Consider a new example on algebraic surfaces.

Example (Sattel-Dattel-Zitrus)

It takes about 3 seconds while QEPCAD cannot solve it in 30 minutes.

```
Sattel := x^2+y^2*z+z^3;
Dattel := 3*x^2+3*y^2+z^2-1;
Zitrus := x^2+z^2-y^3*(y-1)^3;
f := &E([y, z]), (Sattel=0) &and (Dattel=0) &and (Zitrus<0);
QuantifierElimination(f);
```

The output is the inequality:

$$\begin{aligned}
&387420489\,x^{36}+473513931\,x^{34}+1615049199\,x^{32}\\
&-5422961745\,x^{30}+2179233963\,x^{28}-14860773459\,x^{26}\\
&+43317737551\,x^{24}-45925857657\,x^{22}+60356422059\,x^{20}\\
&-126478283472\,x^{18}+164389796305\,x^{16}-121571730573\,x^{14}\\
&+54842719755\,x^{12}-16059214980\,x^{10}+3210573925\,x^{8}\\
&-446456947\,x^{6}+43657673\,x^{4}-1631864\,x^{2}<40328.
\end{aligned}$$

Outline

Verification and synthesis of switched and hybrid dynamical systems (Sturm-Tiwari, ISSAC 2011)

A common problem studied in this field is to determine if a system remains in the safe state if it starts in an initial safe state. A typical approach to solve this problem is to find a certificate, or an invariant set, such that the following are satisfied simultaneously:

- the initial states satisfy the invariant set
- any states that satisfy the invariant set are safe
- the system dynamics cannot force the system to leave the invariant set

Finding such a certificate can be casted into a real quantifier elimination problem.

```
> phi1 := ( ( 74 <= x ) &and ( x <= 76 ) &and ( v = 0 )
  &implies ( -v^2 - a * (x-75)^2 + b >= 0 ) ):

> phi2 := ( ( -v^2 - a * (x-75)^2 + b >= 0 )
  &implies (( 80 >= x ) &and ( x >= 70 )) ):

> phi3 := ( ( -v^2 - a * (x-75)^2 + b = 0 )
  &implies (( -2*v - a * 2 * (x-75)* v >= 0 ) &or ( 2*v - a
  * 2 * (x-75)* v >= 0 )) ):

> phi := phi1 &and phi2 &and phi3:
> t0 := time():
  psi := QuantifierElimination(&A([x,v]),phi,output=rootof);
  t1 := time() - t0;
```

$$\psi := ((0 < a \ \&\text{and} \ a \le 1) \ \&\text{and} \ a \le b) \ \&\text{and} \ b \le \min\left(\frac{1}{a}, 25\,a\right)$$

$$t1 := 15.094$$

Figure: Solve a QE problem related to 1-D robot model

Outline

Summary and future work

- We have presented the command `QunatifierElimination` of the `RegularChains` library.
- The Maple library archive `RegularChains.mla` can be downloaded from `www.regularchains.org`
- The efficiency of `QunatifierElimination` is illustrated by examples.
- Our underlying algorithm algorithm benefit from RC-CAD and related optimizations like RC-TTICAD, early use of equational constraints, etc.
- An application to automatic parallelization of for-loop nests (suggested by A. Größlinger, M. Griebl, and C. Lengauer in JSC 2006) is discussed in our ISSAC 2014 paper.
- Further work is required to get simpler output QFF and partial cylindrical algebraic decompositions.