Cylindrical Algebraic Decomposition in Coq

MAP 2010 - Logroño 13-16 November 2010

Assia Mahboubi

INRIA Microsoft Research Joint Centre (France)
INRIA Saclay – Île-de-France
École Polytechnique, Palaiseau

November 12th 2010

This work has been partially funded by the FORMATH project, nr. 243847, of the FET program within the 7th Framework program of the European Commission.

Motivations

An example from J. Avigad (Types Summer School 2005)

Heuristic procedures for the reals

Remember the example: verify

$$(1 + \frac{\varepsilon}{3(C^* + 3)}) \cdot real(n) < Kx$$

using the following hypotheses:

$$real(n) \leq (K/2)x$$
$$0 < C^*$$
$$0 < \varepsilon < 1$$

Idea: work backwards, applying obvious monotonicity rules.


```
(* Finished transaction in 163. secs *)
```

Reflexion, large scale, again

- Starting from a goal to be proved on an object `t : T`:

```
(P t): Prop
```

- Implement a Coq decision procedure `f : T -> bool`
- Prove a correctness lemma:

```
Lemma f_correct: forall x : T, (f x)= true -> (P x)
```

- Apply the correctness lemma to the goal:

```
apply (f_correct t).
```

- This reduces the goal to proving that:

```
(f t)= true
```

- The goal will be proved if the execution of the program `f` on the value `t` actually produces the value `true`

Cylindrical Algebraic Decomposition

- So far we have seen an elementary yet intractable decision algorithm;
- One of our motivation is to enhance the automation of the formalization process;
- We cannot hope anything reasonable from an algorithm with a complexity which is a tower of exponential;
- Collin's cylindrical algebraic decomposition is the best known and implemented algorithm:
  - QEPCAD
  - Redlog
  - Axiom
  - Mathematica
  - (Coq)

Work plan

- Implementation of a cylindrical algebraic decomposition algorithm in the Coq system;
- Formal proof of correctness of the algorithm.

Remarks:

- Both the program and the correctness proof are challenging
- This provides a complete formally guaranteed program
- Yet efficient decision might benefit from other approaches

Cylindrical Algebraic Decomposition

A cylindrical algebraic decomposition (CAD) of R^n is a sequence $\mathcal{S}_1, \ldots, \mathcal{S}_n$ such that:

- $\forall i, \mathcal{S}$ is a partition of R^i (of level i);
- Each cell of level 1 is a point or an interval;
- For each cell S of level i, $S \times R$ is the disjoint union of cells of level $i+1$
- For each cell S of level i, there exists $\xi_1, \ldots, \xi_n$ algebraic functions such that a cell of level $i+1$ above S is:
  - either the graph of a semi-algebraic function
  - or a band between the graph of two algebraic functions

Cylindrical Algebraic Decomposition

Let $\mathcal{P} \subset \mathbb{Q}[X_1, \ldots, X_n]$ a finite family of polynomials, and R a real closed field:

- A finite partition of R is adapted to $\mathcal{P}$ if on each cell, each $P \in \mathcal{P}$ has a constant sign.
- There exists a CAD $\mathcal{S}_1, \ldots, \mathcal{S}_n$ such that $\mathcal{S}_n$ is adapted to $\mathcal{P}$.
- Moreover we can compute a sample of points for $\mathcal{S}_n$.

Cylindrical Algebraic Decomposition

$$\mathbb{R}[X_1, \ldots, X_n] \qquad\qquad \mathbb{R}[X_1, \ldots, X_{n-1}]$$

$$\mathcal{P} = P_1, \ldots, P_s \xrightarrow{\text{projection}} \mathcal{Q} = Q_1, \ldots, Q_t$$

$$\text{CAD and signs for } \mathcal{P} \xleftarrow[\text{lifting}]{} \text{CAD and signs for } \mathcal{Q}$$

$$\mathbb{R}^n \qquad\qquad \mathbb{R}^{n-1}$$

Cylindrical Algebraic decomposition for $X + Y$

Remarks

- The polynomials studied are the same for all the cells at a same level.
- There is much more than one point per connected components.
- The tree structure is used for deciding prenex formulas.

Deciding $\exists X, \forall Y, X + Y$

Deciding $\exists X, \forall Y, XY$

Description of the algorithm: key issues

- Lifting phase: re-compose a CAD for R^{k+1} from a CAD for R^k.

  One dimensional real root isolation
- Projection phase: determine the family of polynomials each partition should be adapted to.

  Control over the growth of the number of cells.

In all what follows we focus on archimedean discrete real closed fields.

Base case: real root isolation

- Input : $\mathcal{P} \subset \mathbb{Q}[X]$ a finite family of polynomials;
- Problem: partition the real line R into cells over which every $P \in \mathcal{P}$ has a constant sign;
- Solution: determine and isolate all the roots of the polynomials in $\mathcal{P}$

We consider only square free polynomials: $P \wedge P'$ has the same roots as P, but they are simple.

Base case: real root isolation

A root isolation for P is a list of disjoint open intervals and points, such that:

- Every root of P belongs to an element of the list;
- Every element of the list contains a single root of P.

The method:

- Bound the problem using Cauchy bounds:
$$\forall x, \sum_{i \leq n} a_i x^i = 0 \Rightarrow |x| \leq |a_n|^{-1} * \sum_{i \leq n} |a_i|$$
- Process by dichotomy using Descartes' law of sign based tools.

Base case: real root isolation

If the family $\mathcal{P}$ has a single element P.

We can program a test $T(P, a, b)$ such that:

- If $T(P, a, b) = 0$ then P has no root in $]a, b[$
- If $T(P, a, b) = 1$ then P has a single root in $]a, b[$
- Else the situation is unknown
- But if a and b are close enough we are in one of the two first cases.

A simple dichotomy process isolates the roots of P.
A sample of points is easy to compute.

Base case: real root isolation

If the family $\mathcal{P}$ has $n+1$ elements.

- Bound globally the interval of interest;
- Compute I isolating the roots of $P_1, \ldots, P_n$;
- Outside intervals delimited by elements of I, compute an isolation of the roots of P_{n+1};
- On each $]A, B[$ in I, containing a root of P_i:
  - determine the sign of P_{n+1} at A and B (evaluation)
  - determine the sign of P_{n+1} at the unique root of P_i contained in $]A, B[$ (gcd and refinement)
  - determine all the signs of P_{n+1} on the interval $]A, B[$
- Add the new sample points to the previous list

The base case of CAD is addressed, up to programming the test.

Bernstein polynomials

Let $c, d \in \mathbb{Q}$, $p \in \mathbb{N}$.
The Bernstein basis of degree p is defined by:

$$B_{p,i}(c,d) = \binom{p}{i} \frac{(X-c)^i (d-X)^{p-i}}{(d-c)^p}$$

The are obtained from the standard basis by translation, dilatation, and reverse of the coefficients.

Bernstein polynomials

- The test $T(P, a, b)$ for P of degree d is implemented by the number of sign changes in the list of coefficients of P in $B_d(a, b)$:

  Descartes' law of signs

- The dichotomy process means recomputing Bernstein coefficients on sub-intervals:

  de Casteljau algorithm

- The archemedeanity is crucial to termination.

Analysis of the method

The base case deals with polynomials in $\mathbb{Q}[X]$.
But the whole process is still valid if polynomials are in $D[X]$ where D a sub-ring of R containing $\mathbb{Q}$ for which:

- The sign of any $d \in D$ is computable;
- The sign of $P(q)$ where q is rational and $P \in D[X]$ is computable.

Lifting phase

- Suppose we have constructed a CAD $\mathcal{S}_k$ of R^k;
- We need to construct a CAD $\mathcal{S}_{k+1}$ of R^{k+1} for the family $\mathcal{P} \subset \mathbb{Q}[X_1, \ldots, X_k, X_{k+1}]$

Lifting phase

- Cells of level $k+1$ are pieces of the cylinder $S \times R$;
- For each cell $S \in \mathcal{S}_k$, we have computed a sample point x^S;
- The cells obtained as pieces of $S \in \mathcal{S}_k$ are described by a root isolation of the univariate family $\mathcal{P}(x^S) \subset \mathbb{Q}[x_1^S, \ldots x_k^S, X_{k+1}]$

Lifting phase

This lifting phase is addressed by our remark on the base case if $D := \mathbb{Q}[x_1^S, \ldots x_k^S]$ satisfies the conditions:

- The sign of elements in $\mathbb{Q}[x_1^S, \ldots x_k^S]$ can be computed;
- The sign of $P(x_1^S, \ldots x_k^S, r)$ with $P \in \mathbb{Q}[X_1, \ldots, X_k, X_{k+1}]$ and $r \in \mathbb{Q}$ can be computed.

Algebraic numbers

Cells which are thin in at least one dimension make algebraic number arithmetic unavoidable.

- Algebraic numbers in $\mathbb{Q}^n_{\text{alg}}$ are represented by triangular encodings.
- Required computations are obtained by recursive use of root isolation.
- Triangular encodings include a bounding box which can be used for testing sign conditions by interval arithmetic computations.

Projection operator

Given $\mathcal{P} \subset \mathbb{Q}[X_1, \ldots, X_k, X_{k+1}]$, a family $\mathcal{Q} \subset \mathbb{Q}[X_1, \ldots, X_k]$ is a good projection if on any partition on R^k adapted to $\mathcal{Q}$, and any cell S of such a partition:

- For every $P \in \mathcal{P}$, the degree and the number of roots of $P(x, X_{k+1})$ is independent from the choice of $x \in S$
- For every $P_1, P_2 \in \mathcal{P}$, the degree of $gcd(P_1(x), P_2(x))$ is independent from the choice of $x \in S$.

Multivariate gcds

The last condition demands investigating gcds for univariate polynomials with coefficients in a ring, like $\mathbb{Q}[X][Y]$:

- In an Euclidean ring, the gcd of two elements are computed using the Euclidean algorithm.
- But if R is an Euclidean ring, there is no reason $R[X]$ should be Euclidean as well.
- Be we can still use pseudo-Euclidean division, and pseudo-gcd.

Multivariate gcds

A pseudo Euclidean division corrects the default of divisibility so that the Euclidean division can still be tractable inside the ring:

$$cA = B * Q + R$$

with $d(A) = n, d(B) = m$, $lcoef(A) = \alpha$, $lcoef(B) = \beta$.

Multivariate gcds

A pseudo Euclidean division corrects the default of divisibility so that the Euclidean division can still be tractable inside the ring:

$$cA = B * Q + R$$

with $d(A) = n, d(B) = m$, $lcoef(A) = \alpha$, $lcoef(B) = \beta$.
There are many suitable choices for c:

- The Jacobi factor $c := b_m^{n-m+1}$ is always sufficient.
  But leads to an exponential growth in the size of the coefficients.
- We can always simplify polynomials by their content.
  But this leads to recursive gcd computations.
- A trade-off is not easy to find but crucial in our case.

Multivariate gcds

The control over gcd computation is obtained by allowing one more degree of freedom in the definition of each pseudo-division step:

$$cA = B * Q + dR$$

Such a chain is called a pseudo-remainder sequence.

The subresultant algorithm defines a pseudo-remainder sequence which achieves the best trade off (linear in the degrees and bitsize of coefficients).

The vanishing of leading coefficients of subresultant polynomial control the degree of the gcd of A and B.

The actual projection operator

There are many way of achieving the projection. Here we project $\mathcal{P} \subset \mathbb{Q}[X_1, \ldots, X_k, X_{k+1}]$ on $\mathcal{P}$ by:

- Saturating $\mathcal{P}$ with relevant truncations
- Adding to $\mathcal{Q}$ all the possibly "vanishing in a row" head coefficients of polynomials in $\mathcal{P}$
- Adding to $\mathcal{Q}$ all the leading coefficients of subresultant polynomials of P and P' for $P \in \mathcal{P}$
- Adding to $\mathcal{Q}$ all the leading coefficients of subresultant polynomials of P_1 and P_2 for $P_1, P_2 \in \mathcal{P}$

The actual projection operator

Over a cell S of a partition adapted to $\mathcal{Q}$:

- Polynomials in $\mathcal{P}$ have a constant degree, since their leading coefficients do not cancel.
- Each polynomial of $\mathcal{P}$ have a constant number of root, since a change in the number of roots would imply a change in the multiplicity of a root (by continuity).
- The pairwise gcds also do not change their degree hence the behavior of common roots is stable over the S.

The complete algorithm

- The projection operator
- The isolation of roots of univariate polynomials in $D[X]$
- The recursive process
  - Base case uses the isolation process with $D = \mathbb{Q}$
  - Recursive case builds sign computations elements of $D(\alpha)$ for any $\alpha \in \mathbb{Q}_{alg}$ and root isolation for polynomials in $D(\alpha)[X]$

Conclusion

- This has been implemented in the Coq system.
- It's purely functional program, but could benefit from semi-imperative features.
- It should be completed by other incomplete, possibly certificate based decision methods (sums of squares...).
- The correctness proof is in progress.
- It will benefit from the recent developments in formalized mathematics (algebraic hierarchy, linear, multilinear algebra,...)