Seminar for PhD student

Computational Algebra and Primary Decomposition

Elisa Palezzato

Unige - Dima

January 20, 2016

From Wikipedia

A **C**omputer **A**lgebra **S**ystem:

is a software program that allows computation over mathematical expressions in a way which is similar to the traditional manual computations of mathematicians and scientists.

The development of the computer algebra systems started in the second half of the 20th century and this discipline is called "computer algebra" or "symbolic computation".

Computer algebra systems may be divided in two classes:

- The specialized ones are devoted to a specific part of mathematics, such as number theory, group theory, etc. [Macaulay2, Singular, ...]
- General purpose computer algebra systems aim to be useful to a user working in any scientific field that requires manipulation of mathematical expressions. ["Matlab", Maple, Magma, ...]

Open Source

What is CoCoA?

History

The CoCoA project started in 1987 under the lead of L. Robbiano (Giovini & Niesi & Capani)
Aim: a *mathematician-friendly* software for

Computations in Commutative Algebra

especially **Gröbner bases**.

Present

It has evolved and has been rewritten, always maintaining this tradition, and now offers: (CoCoA-4, CoCoA-5: Abbott & Bigatti)

- an open source **C++ software library**: CoCoALib
- a **new interactive system**: CoCoA-5
- a prototype OpenMath-based **server**

What can I compute with CoCoA-5?

- Gröbner bases of ideals/modules, wide choice of term orderings
- special handling for ideals of points
- special handling for monomial ideals
- Hilbert series, *resolutions*, *Betti numbers*
- polynomial factorization
- basic exact linear algebra (`LinSolve`, `LinKer`, `eigenvectors`, `det`)
- approximate points: border bases, polynomial relations
- ...

New in the CoCoA-5 language

Even though all new, more robust and expressible CoCoA-5 language is mostly compatible with CoCoA-4

"Invisible multiplication" (`xy` for `x*y`) gave CoCoA-4 many constraints.
Now allowed only inside triple-`*`:

```
I := ideal(2*x^2*y -z, 3*x*z -5*y*z^3);
I := *** Ideal(2x^2y -z, 3xz-5yz^3) ***;
```

Rings and *functions* are now *first class values*:
can be assigned and passed as arguments

Better errors! I mean *error messages*!!!
make it easy to learn and to update CoCoA-4 code

From CoCoA-4 to CoCoA-5: new mathematics!

More rings: algebraic extensions, fraction fields ...

```
use R ::= QQ[a];  // "use" "::="  for special ring syntax: QQ[x] vs L[3]
K1 := NewFractionField(R);                  // K1 is QQ(a)
K2 := NewQuotientRing(R, ideal(a^2-2)); // K2 is QQ[a]/(a^2-2)
use P ::= K1[x,y,z];
f := x - (1/a)*x + y;            // viewed as  ((a -1)/a)*x +y  in P
```

Ring homomorphisms

```
phi   := CanonicalHom(R, K1);       //   phi: QQ[a] --> QQ(a)
psi   := CanonicalHom(R, K2);       //   psi: QQ[a] --> QQ[a]/(a^2-2)
theta := CanonicalHom(K1,P) (phi); // theta: QQ[a] --> QQ(a)[x,y,z]
use R; // polynomials are read as elements in R = QQ[a]
1/phi(a^2 + 2*a -1);                // gives  1/(a^2 +2*a -1)  in K1
1/psi(a^2 + 2*a -1);                // gives  ((2/7)*a -1/7)   in K2
1/theta(a^2 + 2*a -1);              // gives  1/(a^2 +2*a -1)  in P
```

More than CoCoA itself

Several ways of extending CoCoA-5 for your needs

- Write your own functions in CoCoA-5 language

```
define StrangeFunction(X)
  if   type(X) = INT then return 2^X;
  elif type(X) = MAT then return det(X);
  endif;
  return X;
enddefine;
```

- Collect some functions into a new **CoCoA-5 package**

- Write the new functions in C++ inside CoCoALib,
  and then make them "visible" to CoCoA-5
  (the new interpreter makes this last step really easy!)

What am I working on?

I am studying primary decomposition, from the Computational Algebraic point of view.
This study started with Emmy Nother, who studied and proved the principal instrument for primary decomposition, and her Ph.D. student Grete Hermann, who introduced the first algorithm.

Figure : Emmy Noether and Grete Hermann

A bit of math, some definitions:

Let R be a commutative ring with 1.
An ideal Q is **Primary** if $xy \in Q \Rightarrow x \in Q$ or $y^t \in Q$ where $t \in \mathbb{N}$.

A **Primary Decomposition** of an ideal I of R is an expression of I as a finite intersection of primary ideals:

$$I = Q_1 \cap \ldots \cap Q_s.$$

This decomposition is minimal if:

(i) $\sqrt{Q_i}$ are distinct,

(ii) $Q_i \not\supseteq \bigcap_{j \neq i} Q_j$.

Let K be a field and $P = K[x_1, \ldots, x_n]$ a polynomial ring.
An ideal I of P is **zero-dimensional** if $\dim_K(P/I) < +\infty$.

General case

A **moltiplication endomorphism** is a map,

$$m_r: \quad P/I \to P/I$$
$$x \mapsto rx$$

Given an endomorphism φ of P/I, its **minimal polynomial** is a monic generator of the ideal $\{p(z) \in K[z] : p(\varphi) = 0\}$, and we call it μ_φ.

Theorem

Let I be a zero-dimesional ideal such that $I = Q_1 \cap \ldots \cap Q_s$ then a generic $l \in P/I$ linear is such that

$$Q_i = I + (f_i(l)^{\alpha_i})$$

with $\mu_{m_l} = f_1^{\alpha_1} \cdot \ldots \cdot f_s^{\alpha_s}$ the irreducible factorization of the minimal polynomial of the multiplication endomorphism.

Primary Decomposition on a finite field

Let $P = K[x_1, \ldots, x_n]$ be a polynomial ring on a finite field K, and let I be a zero-dimensional ideal.
The **Frobenius endomorphism**:
$Frob\colon P/I \to P/I$ is define by $x \mapsto x^p$.

Let $FrobSp = \{g \in P/I : g^p - g = 0\}$ be the **Frobenius space**, namely the invariant subspace of the Frobenius endomorphism.

Theorem

Let I be a zero-dimensional ideal such that $I = Q_1 \cap \ldots \cap Q_s$ is its primary decomposition, then:

- $\dim_K(FrobSp) = s$;
- $g \in FrobSp \Leftrightarrow \mu_{m_g}$ it breaks up into distinct linear factors.

Partial decomposition

From the Theorem we know that the minimal polynomial is factored in linear factors. Then the Q_i components are:

$$Q_i = I + (g - \lambda) \text{ with } \lambda \in K.$$

We have also the following facts:

- For every $g \in FrobSp$ we have $d = \deg(\mu_{m_g}(z)) \leq \min\{\text{card}(K), s\}$.
- If $s > 1$ every $g \in FrobSp \setminus K$ is a **partial splitting**.

In this way we can break up the ideal in $\text{card}(K)$ components.
It is enough repeat the process a finite number of times to get all the components of the primary decomposition.

Example

Let P be the ring $\mathbb{Z}_2[x, y]$ and $I = (y^2 + y,\ x^2y,\ x^3 + y + 1)$ the ideal.
A quotient basis of P/I is $QB = [1, y, x, xy, x^2]$ and ... (*Example*)

Partial decomposition

From the Theorem we know that the minimal polynomial is factored in linear factors. Then the Q_i components are:

$$Q_i = I + (g - \lambda) \text{ with } \lambda \in K.$$

We have also the following facts:

- For every $g \in FrobSp$ we have $d = \deg(\mu_{m_g}(z)) \leq \min\{\text{card}(K), s\}$.
- If $s > 1$ every $g \in FrobSp \setminus K$ is a **partial splitting**.

In this way we can break up the ideal in $\text{card}(K)$ components.
It is enough repeat the process a finite number of times to get all the components of the primary decomposition.

Example

Let P be the ring $\mathbb{Z}_2[x, y]$ and $I = (y^2 + y,\ x^2y,\ x^3 + y + 1)$ the ideal.
A quotient basis of P/I is $QB = [1, y, x, xy, x^2]$ and ... (*Example*)

Guidelines of the algorithm

PrimaryDecomposition0

* check that the ideal I is zero dimensional;
* **find the splitting** r **(partial o total,** $\mathrm{char}(K) = 0$ **or** p**)**;
* **compute** μ_{m_r}**, the minimal polynomial** r **in** P/I;
* **factorization of** $\mu_{m_r} = f_1^{\alpha_1} \cdot \ldots \cdot f_s^{\alpha_s}$;
* definition of $Q_i = I + (f_i(r)^{\alpha_i})$,

$$I = Q_1 \cap \cdots \cap Q_s$$

* **certification primariety** Q_i

What to use it for?

Let $L = K[x_1, \ldots, x_n]/\mathcal{M}$ be algebraic extention, where $\mathcal{M}$ is a maximal ideal, and K equal to $\mathbb{Q}$ or $\mathbb{F}_p$. Let $f(z) \in L[z]$ be a polynomial, the factorization of $f(z)$ could be compute through the Primary Decomposition of the ideal $\mathcal{M} + (f(z))$ in the polynomial ring $K[x_1, \ldots, x_n, z]$.

Example

```
Use ZZ/(2)[x,z];
I := ideal(x^3+x+1, z^5+z+1);
Q := PrimaryDecomposition0(I);
/**/ indent(Q, 2);
record[  IsCertified := true,
         PrDec_I := [   ideal(z^2 +z +1, x^3 +x +1),
                        ideal(x^2 +z +1, z*x +1, z^2 +z +x),
                        ideal(x^2 +z +x +1, z*x +z +x, z^2 +x +1),
                        ideal(z +x +1, x^3 +x +1)]] /**/
```

$$f(z) = (z + \overline{x^2 + 1})(z + \overline{x + 1})(z^2 + z + \overline{1})(z + \overline{x^2 + x + 1})$$

Next steps...

- Factorization of algebraic extensions.
- Certification of Primary Decomposition.
- Compute Minimal Polynomial.
- Compute Radical Ideal.
- Study of isomorphisms of representations for $\mathbb{F}_q$.
- Explicit solutions for polynomial equation systems.

CoCoA-5

want you