The CoCoA project dates back to 1987 under the lead of L. Robbiano: the aim was to create a “mathematician”-friendly laboratory for studying Commutative Algebra, especially Gröbner bases. Since then, always maintaining this tradition, it has evolved and has been rewritten, and now offers: an open source C++ software library, CoCoALib [2]; a new interpreter for the interactive system, CoCoA-5 [1]; a prototype OpenMath-based server.

The openness and clean design of CoCoALib and CoCoA-5 are intended to offer different levels of usage, and to encourage external contributions.

CoCoA-5: Gröbner bases at ease

The easiest way to compute a Gröbner basis is with the interactive system CoCoA-5. This is a small sample:

Use QQ[x,y,z];
I := ideal(x^3 + x*y^2 - 2*z, ........ );
GBasis(I);

The new CoCoA-5 language, while mostly compatible with CoCoA-4, provides greater expressibility and a more solid mathematical basis. In particular it offers full flexibility for the field of coefficients: for example fraction fields and algebraic extensions of any ring, and even heuristically guaranteed floating point arithmetics with rational reconstruction.

CoCoALib: Gröbner bases in C++

We envisage researchers wishing to tackle a hard computation developing a prototype implementation in the convenient environment of CoCoA-5, and then translating the code using CoCoALib into C++ for better performance. To facilitate this conversion we have, whenever possible, used the same function names in both CoCoA-5 and CoCoALib, and we preferred traditional “functional” syntax in CoCoALib over object oriented “method dispatch” syntax (e.g. GBasis(I) rather than I.GBasis()). Our small sample before literally translates into:

ring P = NewPolyRing(RingQQ(), symbols("x", "y", "z"));
ideal I = ideal(ReadExpr(P, "x^3 + x*y^2 - 2*z"), ........ );
cout << GBasis(I);
Gröbner bases and beyond

Naturally, most of the source code in CoCoALib was written by us, but the design of the library was chosen to facilitate and encourage contributions.

The first kind of contribution is code written specifically to become part of CoCoALib: for example Janet and Pommaret bases by M. Albert and W. Seiler, and Border bases of ideals of approximate points by M. L. Torrente and C. Fassino. These alternatives for Gröbner Bases have interesting applications and the collaborations are still progressing.

The second kind of contribution consists in the integration of external (open source) code with CoCoALib: with our collaboration B. Roune first integrated some function from Frobby (for monomial ideals) and then C. Söger added Normaliz (for affine monoids or rational cones). There is also an experimental interface with GSL (GNU Scientific Library), and shortly with A. N. Jensen’s help we’ll link GFan (Gröbner fans and tropical varieties).

The mechanism we developed for adding other libraries is quite simple (see [3]). Moreover, thanks to the ingenious design of the interpreter, any function in CoCoALib (by authors, contributors, or integrated libraries) can readily be made “visible” in CoCoA-5 providing full and easy access to all new extensions.

Other extensions and contributions are in the form of CoCoA-5 functions/packages (primary decomposition for 0-dimensional ideals, simplicial complexes, ..), and, last but not least, feedback from CoCoA-5 and CoCoALib users!

References

