FGb: a library for computing Gröbner bases

Jean-Charles Faugère

SALSA Project - INRIA (Centre Paris-Rocquencourt)
UPMC, Univ Paris 06
CNRS, UMR 7606, LIP6
4, place Jussieu F-75252 PARIS CEDEX 05, France
Jean-Charles.Faugere@inria.fr

Abstract. FGb is a high-performance, portable, C library for computing Gröbner bases over the integers and over finite fields. FGb provides high quality implementations of state-of-the-art algorithms (F_4 and F_5) for computing Gröbner bases. Currently, it is one of the best implementation of these algorithms, in terms of both speed and robustness. For instance, FGb has been used to break several cryptosystems.

1 Introduction - Polynomial System Solving - Gröbner Bases

Solving efficiently polynomial system of equations is a fundamental problem in Computer Algebra with many applications. Let $\mathbb{K}$ be a field and $\mathbb{L} \supset \mathbb{K}$. The problem is:

$$\text{Find } z = (z_1, \ldots, z_n) \in \mathbb{L}^n, \text{ such that } \begin{cases} f_1(z_1, \ldots, z_n) = 0 \\ \cdots \\ f_m(z_1, \ldots, z_n) = 0 \end{cases} \text{ where } f_i \in \mathbb{K}[x_1, \ldots, x_n]$$

To solve this problem several methods have been proposed : semi-numerical methods (homotopies), heuristics (for instance SAT solvers when $\mathbb{K} = \mathbb{L} = \mathbb{F}_2$), probabilistic (geometrical resolutions [13]) or exact methods. Among the exact methods one can cite Gröbner bases, Triangular sets methods or Resultant based techniques. In this paper we restrict ourselves to Gröbner basis computation [1]. It is beyond the scope of this paper to explain, in full generality, why a Gröbner basis can be used to solve a polynomial system, but in finite fields (the case $\mathbb{K} = \mathbb{L} = \mathbb{F}_p$) univariate polynomials can be computed (such polynomials can be obtained in a Gröbner basis for an appropriate elimination ordering), and, then, the solutions can be explicitly computed by factoring these polynomials in $\mathbb{F}_p[X]$.

2 Goal and architecture of the library

The purpose of the `FGb` library [7] is twofold. First of all, the main goal is to provide efficient implementations of state-of-the-art algorithms for computing Gröbner bases: actually, from a research point of view, it is mandatory to have such an implementation to demonstrate the *practical efficiency* of new algorithms. Secondly, in conjunction with other software, the FGb library has been used in various applications (Robotics, Signal Theory, Biology, Computational Geometry, ...) and more recently to a wide range of problems in Cryptology (for instance, `FGb` was explicitly used in [2, 8, 9, 4, 5] to break several cryptosystems). Historically, the `Gb` library (191 420 lines of C++ code) has first been written to implement "classical" algorithms (Buchberger's algorithm[1], FGLM [12], NormalForms, Hilbert functions, . . .). The current iteration of the project – the FGb library (206 052 lines of C code) – was restarted from scratch to demonstrate the practical efficiency of a family of new algorithms (F_4[10], matrix-F_5[6], F_5[11], SAGBI-F_5[6],. . .). All these algorithms have in common that they rely heavily on linear algebra. Hence, whereas efficient internal representation for distributed multivariate polynomials was a key component in Gb, the critical part in FGb is the linear algebra package. The design of the library is somewhat modular: for instance, it is easy to add a new field of coefficients $\mathbb{K}$ (FGb provides already 19 different fields); it is even possible to replace the existing linear algebra package by another one (see section section 4). Even if a small portion of the code has been written in assembler code the library is *portable*; to date the library is available on several architectures: Windows (32 and 64 bits), Linux (32 and 64 bits), Mac (Universal ppc/intel 32/64 bits) and Sun Solaris.

3 Maple interface - C library mode

The FGb/Gb library has no friendly interface but it can be called from any C code. In a partnership with MapleSoft, the library has been *dynamically linked* with the kernel of the Computer Algebra system Maple. Users can use the power of expressivity of a general Computer Algebra System to generate the polynomial equations while keeping the efficiency of a dedicated library. The library is shipped with all recent versions of Maple and fully integrated with high level functions in Maple (for instance the universal solve function in Maple can call the FGb library if needed). It is also possible to call directly the internal package (and hence have access to more options for expert users):

```
    |\^/|     Maple 13 (X86 64 LINUX)
._|\|   |/|_. Copyright (c) Maplesoft, a division of Waterloo Maple Inc.
 \  MAPLE  /  All rights reserved. Maple is a trademark of
 <____ ____>  Waterloo Maple Inc.
      |       Type ? for help.
> with(fgbrs):
> # the output of fgb_gbasis is a list [[lc1,lt1,p1],...]
    where lc1*lt1 is the leading monomial of p1.
> fgb_gbasis([x-y^2,x^2-y],[x,y]);
                            2   2            2   2
                      [[1, y , y  - x], [1, x , x  - y]]
> # Same computation in GF(65521) for an elimination ordering x>>y
> fgb_gbasis([x-y^2,x^2-y],[x],[y],65521);
                      4   4                              2
                 [[1, y , y  + 65520 y], [1, x, x + 65520 y ]]
```

More generally, it is easy to call FGb from any C program (API and sample program available [7]): for instance, Coq can use FGb to proof some polynomial equations (the link was done by L. Pottier).

4 New High Performance linear algebra package - Benchmarks.

In [3], we present a recent new dedicated linear algebra package written by S. Lachartre for computing Gaussian elimination of matrices coming from Gröbner bases computations. This library is also written in C and contains new algorithms to compute Gaussian elimination as well as specific internal representation of matrices (namely sparse triangular blocks, sparse rectangular blocks and hybrid rectangular blocks). In order to demonstrate the efficiency of this combination of software we give some computational results for a well known benchmark: the Katsura problem. For instance, for a medium size problem such as Katsura 15, it takes 849.7 sec on a PC with 8 cores to compute a DRL Gröbner basis modulo $p < 2^{16}$; this is 88 faster than Magma (V2-16-1).

	Kat11	Kat12	Kat 13	Kat 14	Kat 15	Kat 16
Magma	19.5	151.2	1091.4	9460.35	74862.9	NA
FGb F_4	40.6	342.6	2550.65			
FGb F_5			191.7	1881.3	12130.8	103110.0
New linalg + F_4	**2.85**	**19.45**	**149.6**			
New linalg + F_5			**27.6**	**180.7**	**849.7**	**5687.3**

Benchmark: Katsura n modulo 65521 - PC with 8 cores.

References

1. Buchberger B. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal.* PhD thesis, Innsbruck, 1965.
2. J.-C. Faugère and A. Joux. Algebraic Cryptanalysis of Hidden Field Equation (HFE) Cryptosystems Using Gröbner bases. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *LNCS*, pages 44–60. Springer, 2003.
3. J.-C. Faugère and S. Lachartre. Parallel Gaussian Elimination for Gröbner bases computations in finite fields. In M. Moreno-Maza and J.L. Roch, editors, *ACM proceedings of The International Workshop on Parallel and Symbolic Computation (PASCO)*, pages 1–10, LIG, July 2010. ACM.
4. J.-C. Faugère, F. Levy-dit Vehel, and L. Perret. Cryptanalysis of Minrank. In David Wagner, editor, *Advances in Cryptology CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 280–296, Santa-Barbara, USA, 2008. Springer-Verlag.
5. J.-C. Faugère, A. Otmani, L. Perret, and J.-P. Tillich. Algebraic Cryptanalysis of McEliece Variants with Compact Keys. In *Proceedings of Eurocrypt 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 279–298. Springer Verlag, 2010.
6. J.-C. Faugère and S. Rahmany. Solving systems of polynomial equations with symmetries using SAGBI-Gröbner bases. In *ISSAC '09: Proceedings of the 2009 international symposium on Symbolic and algebraic computation*, pages 151–158, New York, NY, USA, 2009. ACM.
7. J.C. Faugère. `FGb` library for comptuing Gröbner bases. available at `http://www-salsa.lip6.fr/ jcf/Software/FGb/`.
8. Jean-Charles Faugère and Ludovic Perret. Cryptanalysis of 2R– schemes. In Cynthia Dwork, editor, *Advances in Cryptology CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 357–372. Springer-Verlag, 2006.
9. Jean-Charles Faugère and Ludovic Perret. Polynomial Equivalence Problems: Algorithmic and Theoretical Aspects. In Serge Vaudenay, editor, *EuroCrypt 2006 Advances in Cryptology*, volume 4004 of *Lecture Notes in Computer Science*. Springer-Verlag, 6 2006.
10. Faugère J.C. A new efficient algorithm for computing Gröbner bases (F4). *Journal of Pure and Applied Algebra*, 139(1–3):61–88, June 1999.
11. Faugère J.C. A new efficient algorithm for computing Gröbner bases without reduction to zero F5. In T. Mora, editor, *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation*, pages 75–83. ACM Press, July 2002.
12. Faugère, J.C., Gianni, P., Lazard, D. and Mora T. Efficient Computation of Zero-Dimensional Gröbner Basis by Change of Ordering. 16(4):329–344, October 1993.
13. G. Lecerf. `Kronecker` Magma package for solving polynomial systems by means of geometric resolutions. available at `http://www.math.uvsq.fr/ lecerf/software/`.