On Computation of Gröbner Bases over $\mathbb{F}_2$

V.P. Gerdt, M.V. Zinin
Laboratory of Information Technologies, JINR

Abstract

In this paper we consider a version of Janet division algorithm and its implementation in C++ oriented to computing degree-reverse-lexicographical Gröbner bases for polynomial ideals in the ring of multivariate polynomials over the finite field $\mathbb{F}_2$. We compare efficiency of the algorithm and its implementation with those for Buchberger's algorithm and with the Gröbner bases software built-in computer algebra systems Singular, CoCoA and with the library FGb available for Maple. As benchmarks for our comparison we use conversion to $\mathbb{F}_2$ of the some benchmarks widely used for the Gröbner bases software over Q. Polynomial systems over $\mathbb{F}_2$ are of interest in particular for simulation of quantum computation and in cryptoanalysis.

1 Implemented Algorithms

1. *Buchberger's algorithm.* For preliminary testing of data structures and for subsequent experimental comparison with our implementation of the involutive algorithm [1, 2] we implemented first Buchberger's algorithm [3]. Unlike the former, the latter algorithm examines all S-polynomials and by this reason its computational efficiency heavily depends on the use of criteria to avoid unnecessary S-polynomial reductions.

2. *Involutive algorithm.* This algorithm, designed by Gerdt and Blinkov [1], who exploited constructive ideas of completion to involution of differential systems, was implemented in its improved version [2] and for degree-reverse-lexicographic order. This particular being heuristically optimal for computation of Gröbner bases over Q is also best for homogeneous generating sets. To analyze solutions in $\mathbb{F}_2$ for polynomial systems over $\mathbb{F}_2$ that is important, in particular, for application to quantum computation [4] and cryptoanalysis [5] the pure lexicographical order is more appropriate. Development of the package with inclusion of the lexicographic order is planned as the next step.

2 Data structures

Both of the above implementation are to be special modules of the open source software Ginv whose current version 1.2 is available on the Web page `http://invo.jinr.ru`. By this reason most of the data structures were taken over from Ginv. Functionality of some other structures, for instance, Janet trees, was extended in such a way that interface was preserved. However, two most important data structures – monomial and polynomial – were significantly refined under specifical features of $\mathbb{F}_2$. In particular, the main data fields of the monomial class are now the monomial degree (integer number) and the set of exponents for variables (bit array of length 64 or 128 bits).

3 Algorithmic peculiarities of implementation

Since all operations over monomials and polynomials are performed over the finite field $\mathbb{F}_2$, any variable can have degree either 0 or 1. As a result any monomial order $\succ$ is not admissible. It is easy to see. Let m_1, m_2 be two different monomials satisfying $m_1 \succ m_2$, and let m_3 be the third monomial such that $m_3 = \mathrm{lcm}(m_1, m_2)$. Then we obtain $m_1 \cdot m_3 = m_2 \cdot m_3$ that contradicts the definition of admissible monomial order. This fact can be explicitly verified:

$$
\begin{aligned}
&m_1 := x_1^{i_1} \cdots x_n^{i_n},\ m_2 := x_1^{j_1} \cdots x_n^{j_n},\ m_1 \succ m_2\,,\\
&m_3 := \mathrm{lcm}(m_1, m_2) = x_1^{\max(i_1,j_1)} \cdots x_n^{\max(i_n,j_n)}\,,\\
&m_1 \cdot m_3 = x_1^{\max(i_1,\max(i_1,j_1))} \cdots x_n^{\max(i_n,\max(i_n,j_n))} = m_3\,,\\
&m_2 \cdot m_3 = x_1^{\max(j_1,\max(i_1,j_1))} \cdots x_n^{\max(j_n,\max(i_n,j_n))} = m_3\,.
\end{aligned}
$$

By this reason one has to take care of applicability of the Involutive algorithm.

And one more unusual feature: a basis consisting of a single polynomial may not be a Gröbner basis. We give a simple example.

$$\langle xy + x + 1 \rangle = \langle x + 1, y \rangle$$

If one multiplies polynomial $xy + x + 1$ by all polynomials in the bivariate ring – there are 15 of them – it becomes obvious that the Gröbner basis consists of two polynomials: $x + 1$ and y.

4 Role of criteria

We implemented four involutive criteria [2] for detection of some zero-redundant prolongations as well as equivalent to them two Buchberger's criteria [3] for the case of Buchberger's algorithm. In so doing we adopted the involutive criteria to computation over field $\mathbb{F}_2$. We observe experimentally rather high efficiency of applying the criteria in Buchberger's algorithm. In most cases it achieves 96–100%, i.e. the criteria do not apply for at most 4 from every 100 zero-redundant S-polynomials. As to the Involutive algorithm, the efficiency of criteria is somewhat lower. As a rule it is about 60–85%. With all this going on, and for the benchmarks we used, most often the first criterion (see [2]) was applied.

5 Comparison with other Gröbner bases software

We did comparison of running time for our two implementations with some other computer algebra systems and packages implementing computation of Gröbner bases over $\mathbb{F}_2$, namely, with CoCoA 4.6 [6], Singular 3.0.2 [7] and FGb 1.34 library [8] for Maple. The timings for the standard serial benchmarks *eco*, *katsura*, *redcyclic* and *redeco* (see [9]) are shown in Figures 1 – 4, respectively.

These timings were obtained on a 2xOpteron-242 (1.6 Ghz) machine with 6Gb RAM running under Gentoo Linux 2005.1 and with gcc-4.1.0 compiler.

More detailed comparison together with description of the algorithms implemented is given in [10].

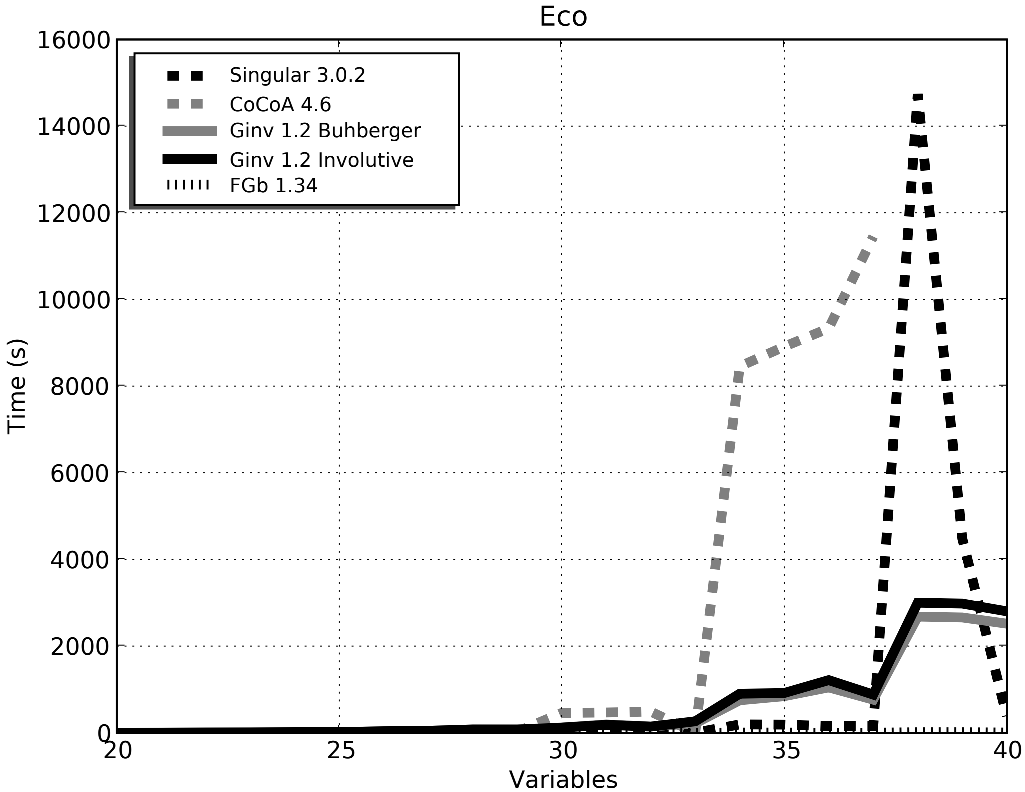

Figure 1: Timings for the *eco* benchmarks

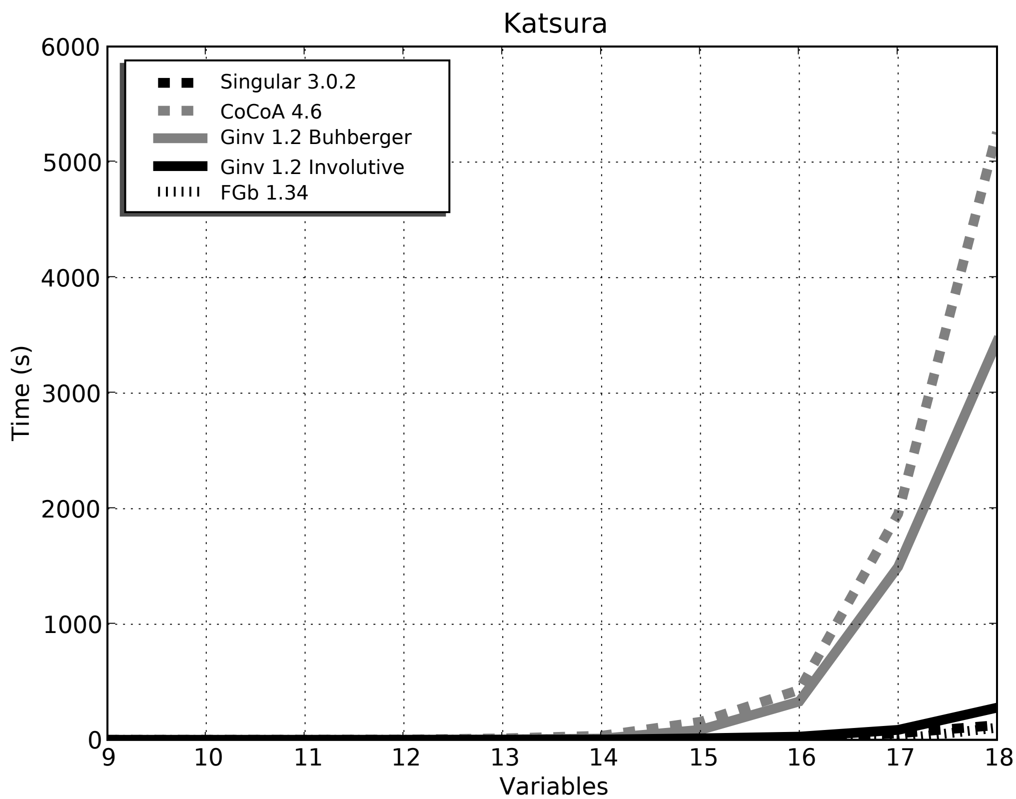

Figure 2: Timings for the *katsura* benchmarks

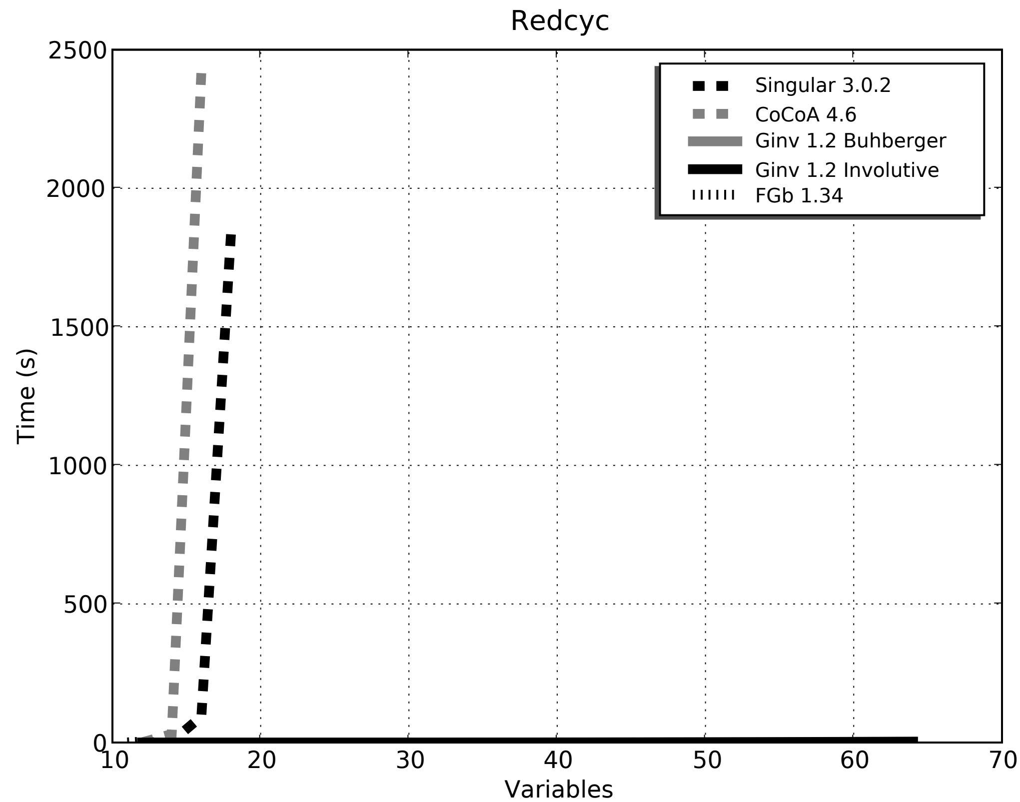

Figure 3: Timings for the *redcyclic* benchmarks

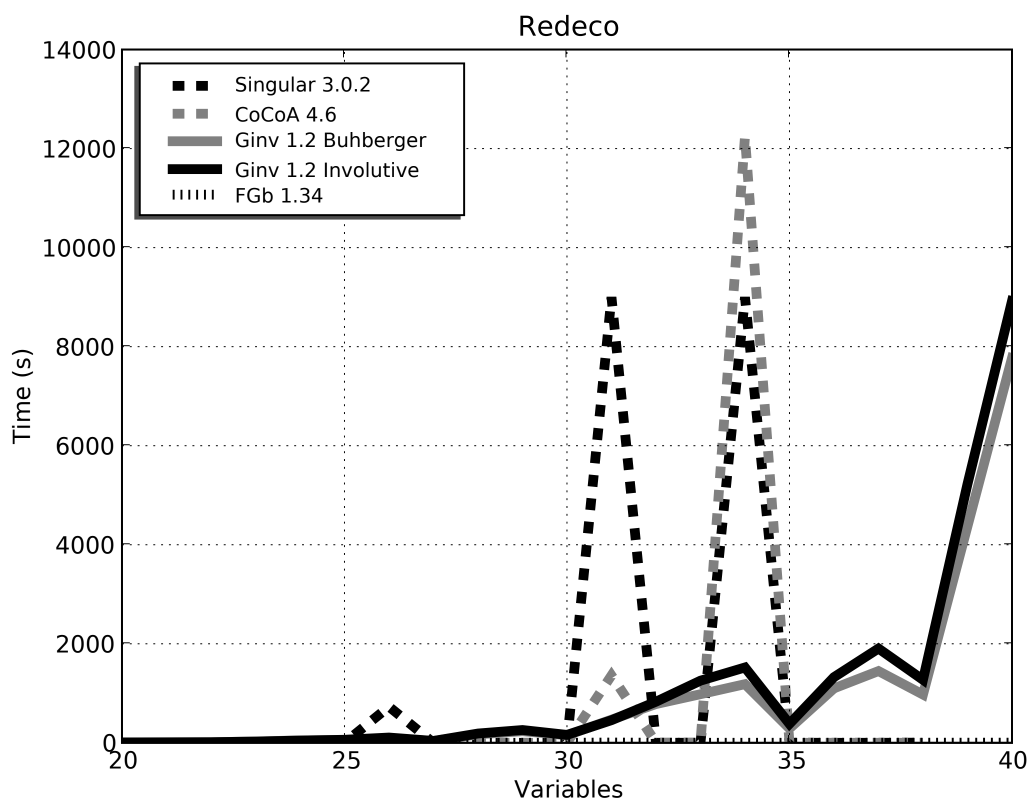

Figure 4: Timings for the *redeco* benchmarks

References

[1] Gerdt, V.P. and Blinkov,Yu.A.: Involutive Bases of Polynomial Ideals. Mathematics and Computers in Simulation 45 (1998) 519–542, arXiv:math.AC/9912027; Minimal Involutive Bases. Ibid., 543–560, arXiv:math.AC/9912029.

[2] Gerdt, V.P.: Involutive Algorithms for Computing Grobner Bases. In: "Computational Commutative and Non-Commutative Algebraic Geometry", S.Cojocaru, G.Pfister and V.Ufnarovski (Eds.), NATO Science Series, IOS Press, 2005, pp. 199-225. arXiv:math.AC/0501111.

[3] Buchberger, B.: Gröbner Bases: an Algorithmic Method in Polynomial Ideal Theory, In: *Recent Trends in Multidimensional System Theory*, N.K. Bose (ed.), Reidel, Dordrecht (1985) 184–232.

[4] Gerdt, V.P. and Severyanov, V.M.: An Algorithm for Constructing Polynomial Systems Whose Solution Space Characterizes Quantum Circuits. In: "Quantum Informatics 2005", Yu.I.Ozhigov (Ed.), SPIE Proceedings, Volume 6264, 2006.

[5] Faugère, J.C. and Joux, A.: Algebraic cryptanalysis of Hidden Field Equations (HFE) Using Gröbner Bases. LNCS 2729, Springer-Verlag, 2003, pp. 44–60.

[6] http://cocoa.dima.unige.it/

[7] http://www.singular.uni-kl.de

[8] http://fgbrs.lip6.fr/salsa/Software/

[9] http://www-sop.inria.fr/saga/POL

[10] Gerdt, V.P. and Zinin, M.V. On computation of Gröbner bases over F_2. Proceedings of the International Workshop on Computer Algebra and Differential Equations / CADE-2007 (Turku, Finnland, February 20-24, 2007), to appear.