F5: A REDUCE Package for Signature-based Gröbner Basis Computation

Alexander Demin[1 [0000−0002−1679−8665]], Hamid Rahkooy[1 [0000−0001−5789−2976]], and Thomas Sturm[2,1,3 [0000−0002−8088−340X]]

[1] MPI Informatics, Saarland Informatics Campus, Germany
[2] CNRS, Inria, and the University of Lorraine, France
[3] Saarland University, Saarland Informatics Campus, Germany

{alexander.demin,hamid.rahkooy}@mpi-inf.mpg.de, thomas.sturm@cnrs.fr

1 Introduction

We report on work in progress on an implementation of the F5 algorithm [10] in REDUCE [16]. In 1965, Buchberger introduced Gröbner Bases, along with an algorithm for their computation [4,5]. The performance of the algorithm depends crucially on the number and efficiency of the necessary polynomial reductions. Heuristic algorithmic optimizations thus typically try to reduce the number of reductions [5]. In 1999, Faugère introduced his algorithm F4 [9], which exploits linear algebra techniques to perform reductions simultaneously. In 2002, F5 [10] followed, which initiated the area of signature-based Gröbner basis algorithms. F5 considers module of polynomials, and incrementally operates on the signature of the module element, resembling the Buchberger algorithm. F5 owes its efficiency to the *F5 criterion*, which avoids many zero reductions. For regular sequences of polynomials, F5 does not do any reduction to zero [10]. Other variants of signature based algorithms, F5C [8] and M5GB [13], have been proposed that combine fast reduction of F4 with *F5 criterion*.

REDUCE has historically offered an implementation of Buchberger's Algorithm due to Gebauer and Möller [11]. This dates back to the mid 1980s and received several improvements by various developers until the mid of the 1990s, including optimizations of variable orders [3], the sugar strategy [12], and the Gröbner Walk [1]. At its time, that implementation was considered state of the art, and, e.g., used as a reference by Becker et al. for their monograph on Gröbner Bases [2]. In 1999, Dolzmann et al. forked the Gröbner package within the REDUCE distribution and based their package CGB [15, Sect.20.14] for the computation of Comprehensive Gröbner Bases [18] on a simplified core of the existing code base. Around the same time F4 and F5 became popular, and the performance of the REDUCE implementations significantly fell behind other general purpose computer algebra systems. At the end of 2008, REDUCE became open source under a liberal Free-BSD license [16]. In spite of its age, dating back to the mid 1960s [14], REDUCE is still quite popular with a stable and even slightly growing user base. Since its launch at Source Forge in 2008 it had a total of 130k downloads, 13k of them in 2021. It has recently been recognized

with the Open Source Excellence award by Source Forge. Development is active with 6k SVN commits since 2008, 650 of them in 2021.

Our goal here is certainly not to compete with highly optimized implementations developed by teams of specialists in the field. Instead, our motivation is primarily a better support of various other REDUCE components that rely on Gröbner Bases:

1. The above mentioned CGB package for the computation of Comprehensive Gröbner Bases [18] should be modernized and based on F5.
2. The Redlog package [6] for interpreted first-order logic has a real quantifier elimination procedure based on Comprehensive Gröbner Bases [19].
3. Simplification of quantifier-free formulas over the reals throughout Redlog partly uses Gröbner Bases [7].

Nevertheless, our computational experiments have been surprisingly promising so far.

2 Some Implementation Details

We re-use the existing efficient infrastructure to the most possible extent. Specifically the existing REDUCE command `torder` for the term order is used internally. We have forked the existing distributive polynomials from the `dipoly` module aiming at using hash tables as described, e.g., for Maple [17].

We use the *position over term* order on modules, where we iteratively produce d-Gröbner Bases for $d = 0, 1, 2, \ldots$, successively for each signature index. Input polynomials are inter-reduced before assigning signatures. This slightly reduces the number of critical pairs. The F5 criterion is checked twice, once at the time of pair creation and once more when computing S-polynomials. Computation experiments have confirmed that the extra test at creation pays off. Moreover, since in the non-regular setting signatures of critical pair components may coincide, we introduce an additional check and discard such pairs. Currently, rational numbers are used as coefficients. An implementation of modular arithmetic is under way. Normal form computations are limited to top-reductions, which significantly speeds up the reduction.

3 Computational Experiments

Our first table compares the numbers of reductions and critical pairs with our `F5` package versus the REDUCE package `groebner`[4] [15, Sect. 20.28], which uses Buchberger's algorithm with the common criteria and the sugar strategy [5,12]: In all cases, the number of reductions performed by F5 remains much lower. Note that F5 considers even more critical pairs, which is common with signature-based algorithms.

[4] CSL REDUCE, SVN revision 6305

Table 1. Reductions performed by `F5` and `groebner`

Systems	Regular sequence	F5		groebner	
		critical pairs	reductions	critical pairs	reductions
Noon-3	yes	77	10	45	28
Noon-4	yes	618	31	317	99
Noon-5	yes	3805	81	2252	334
Katsura-3	yes	23	4	15	18
Katsura-4	yes	110	11	49	43
Katsura-5	yes	416	24	147	90
cyclic-6	no	45	5	30	15
cyclic-7	no	63	6	42	18
cyclic-8	no	84	7	56	21

Our second table compares the timings of our `F5` package with four other implementations of Gröbner Bases, where all considered computations use rational coefficients: REDUCE `groebner` as above; the Maple[5] command `Basis` with `method=fgb` for F4 and `method=buchberger`; the Singular[6] command `groebner`, which "heuristically chooses the best algorithm."

Table 2. Benchmark timings given in seconds (if not otherwise stated)

Systems	REDUCE `F5`	REDUCE `groebner`	Singular `groebner`	Maple	
				`buchberger`	`fgb`
Noon-6	0.5 s	1.5 s	0.01 s	10 s	0.25 s
Noon-7	5.3 s	420 s	0.9 s	78 s	1.3 s
Noon-8	78 s	507 s	11 s	2,700 s	8.5 s
Katsura-6	0.6 s	0.5 s	0.1 s	1.4 s	0.2 s
Katsura-7	120 s	5 s	1.2 s	13 s	0.4 s
Katsura-8	> 1 h	89 s	11 s	107 s	1.3 s
Cyclic-10	0.5 s	2.8 s	< 0.01 s	0.5 s	0.1 s
Cyclic-11	2.3 s	17.6 s	< 0.01 s	1.4 s	0.2 s
Cyclic-12	18 s	99 s	0.01 s	4.1 s	0.6 s
Henrion-5	4.7 s	12.8 s	0.9 s	2.8 s	1.9 s
Henrion-6	73 s	56 s	4.1 s	35 s	2.1 s

In comparison to REDUCE `groebner`, the computation times of REDUCE `F5` are better in most cases, which is compatible with the lower numbers of reductions observed in Table 1. At the moment we cannot compete yet with the relevant implementations in Singular and Maple.

We note that at the current development stage, the CSL code for `F5` is only byte-compiled. For the final production system the package will be profiled and

[5] Maple, 2021.0
[6] Singular v4.3.0

relevant parts of the code will be migrated to the kernel. Furthermore our polynomial reduction requires some technical improvements. All computations were carried out on an AMD EPYC 7702 Processor (64-Core), running 64-bit Linux.

References

1. Amrhein, B., Gloor, O., Küchlin, W.: Walking faster. In: Proc. DISCO '96, pp. 150—161. ACM (1996)
2. Becker, T., Weispfenning, V., Kredel, H.: Gröbner Bases, a Computational Approach to Commutative Algebra, GTM, vol. 141. Springer, New York (1993)
3. Boege, W., Gebauer, R., Kredel, H.: Some examples for solving systems of algebraic equations by calculating Groebner Bases. J. Symb. Comput. **1**, 83–98 (1986)
4. Buchberger, B.: Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal. Doctoral dissertation, Innsbruck, Austria (1965)
5. Buchberger, B.: Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Gleichungssystems. Aequationes Mathematicae **3**, 374–383 (1970)
6. Dolzmann, A., Sturm, T.: Redlog: Computer algebra meets computer logic. ACM SIGSAM Bulletin **31**(2), 2–9 (1997)
7. Dolzmann, A., Sturm, T.: Simplification of quantifier-free formulae over ordered fields. J. Symb. Comput. **24**(2), 209–231 (1997)
8. Eder, C., Perry, J.: F5c: A variant of Faugère's F5 algorithm with reduced Gröbner Bases. J. Symb. Comput. **45**(12), 1442–1458 (2010)
9. Faugére, J.C.: A new efficient algorithm for computing Gröbner Bases (F4). J. Pure Appl. Algebra **139**(1), 61–88 (1999)
10. Faugère, J.C.: A new efficient algorithm for computing Gröbner Bases without reduction to zero (F5). In: Proc. ISSAC 2002, pp. 75–83. ACM (2002)
11. Gebauer, R., Möller, H.M.: On an installation of Buchberger's algorithm. J. Symb. Comput. **6**(2–3), 275–286 (1988)
12. Giovini, A., Mora, T., Niesi, G., Robbiano, L., Traverso, C.: "One sugar cube, please" or selection strategies in the Buchberger algorithm. In: Proc. ISSAC 1991, pp. 49–54. ACM (1991)
13. Hauke, M.: Signature Gröbner Bases. A comprehensive survey and a new algorithmic approach. Master thesis, Graz University of Technology, Austria (2020)
14. Hearn, A.C.: REDUCE – A user-oriented system for algebraic simplification. ACM SIGSAM Bulletin **1**(6), 50–51 (1967)
15. Hearn, A.C., Schöpf, R.: REDUCE User's Manual: Free Version (2022), https://reduce-algebra.sourceforge.io/manual/manual.pdf
16. Hearn, A.C., et al.: REDUCE: A portable general-purpose computer algebra system (2008), https://sourceforge.net/projects/reduce-algebra/
17. Monagan, M., Pearce, R.: Sparse polynomial multiplication and division in Maple 14. ACM Communications in Computer Algebra **44**, 205–209 (2011)
18. Weispfenning, V.: Comprehensive Gröbner Bases. J. Symb. Comput. **14**(1), 1–29 (1992)
19. Weispfenning, V.: A new approach to quantifier elimination for real algebra. In: Quantifier Elimination and Cylindrical Algebraic Decomposition, pp. 376–392. Springer (1998)