Simple signature-based Groebner basis algorithm

Galkin Vasily
Moscow State University
email: galkin-vv@yandex.ru

May 28, 2012

Abstract

This paper presents an algorithm for computing Groebner bases based upon labeled polynomials and ideas from the algorithm F5. The main highlights of this algorithm compared with analogues are simplicity both of the algorithm and of the its correctness proof achieved without loss of the efficiency. This leads to simple implementation which performance is in par with more complex analogues [1]

Consider polynomial ring $P = k[x_1, \ldots, x_n]$ over field k. Also assume that monoid of its monomials $\mathbb{T}$ has a monomial order$\prec$. A problem asking for a Gröbner basis can be stated for any ideal$(f_1, \ldots, f_l)$ in this ring. One of the approaches to the problem is using iterative method which computes every step a basis for ideal $(f_1, \ldots, f_i), i = 2 \ldots l$ based on the already computed for $(f_1, \ldots, f_{i-1})$ basis R_{i-1} and polynomial f_i. The algorithm described in this paper is designed to perform one step of such computation. So, the algorithm's input data consist of a some polynomial f and a polynomial set referred as $\{g_1, \ldots, g_m\}$ which is Gröbner basis of ideal $I_0 = (g_1, \ldots, g_m)$. After finishing the algorithm should give the resulting polynomial set R being a Gröbner basis of ideal $I = (g_1, \ldots, g_m, f)$. The special cases $f = 0 \Rightarrow I = I_0$ and $\exists i\, g_i \in k \Rightarrow I = P$ are not interesting from the computational point of view, so the further chapters assume that $f \neq 0, \forall i\, g_i \notin k$. The homogeneity of input polynomials is not required unlike the F5 algorithm described in [4].

Definitions

Consider the set$\mathbb{T}_0 = \mathbb{T} \cup \{0\}$ – the monomial monoid extended by zero. The order $\prec$ can be extended to $\mathbb{T}_0$ as $\prec_0$ with definition $\forall t \in \mathbb{T}\, t \succ_0 0$ which keeps the well-orderness property. The notion of division also can be extended to $\mathbb{T}_0$: $t_1 | t_2 \stackrel{\text{def}}{=} \exists t_3\, t_1 t_3 = t_2$. For polynomial $p \in P, p \neq 0$ the highest by $\prec$ monom and coefficient are written as $\mathrm{HM}(p) \in \mathbb{T}$ and $\mathrm{HC}(p) \in k$. For zero we define: $\mathrm{HM}(0) \stackrel{\text{def}}{=} 0 \in \mathbb{T}_0$, $\mathrm{HC}(0) \stackrel{\text{def}}{=} 0 \in k$. The least common multiple of $t_1, t_2 \in \mathbb{T}$ is written as $\mathrm{LCM}(t_1, t_2) \in \mathbb{T}$. In the following all definitions are given for fixed I_0 and f:

Definition 1. The *labeled polynomial* is a pair $h = (\sigma, p) \in \mathbb{T}_0 \times P$, that satisfies the correctness property: $\exists u \in P\ \mathrm{HM}(u) = \sigma, uf \equiv p \pmod{I_0}$. Some terminology is extended to labeled polynomials. The highest monomial is $\mathrm{HM}(h) \stackrel{\text{def}}{=} \mathrm{HM}(p)$ and coefficient is $\mathrm{HC}(h) \stackrel{\text{def}}{=} \mathrm{HC}(p)$. Additionally the *signature* is defined $\mathcal{S}(h) \stackrel{\text{def}}{=} \sigma$ and a notation is introduced for the polynomial – second element of pair: $\mathrm{poly}(h) \stackrel{\text{def}}{=} p$. The set of all labeled polynomials is written as $H \subset \mathbb{T}_0 \times P$. The trivial examples of labeled polynomials are $(1, f)$ and $(0, g)$ for $g \in I_0$. Another labeled polynomial example is $(\mathrm{HM}(g), 0)$ for $g \in I_0$. It satisfies correctness property because we can take u equal to g.

Lemma 2. *The product of $h \in H, t \in \mathbb{T}$ defined as $th \stackrel{\text{def}}{=} (t\sigma, tp) \in H$, is correct.*

The correctness property is checked by directly finding u for th.

[1] *Keywords:* Groebner basis, F5 algorithm, labeled polynomials

Definition 3. If the polynomials and monomial $h'_1, h_2 \in H, t \in \mathbb{T}$ satisfy $\mathcal{S}(h'_1) \succ_0 \mathcal{S}(th_2), \mathrm{HM}(h'_1) = \mathrm{HM}(th_2) \neq 0$, then exists a *signature-safe reduction* h'_1 *by* h_2, resulting in labeled polynomial $h_1 \in H$, equal to:

$$h_1 = (\mathcal{S}(h'_1), \mathrm{poly}(h'_1) + Kt\,\mathrm{poly}(h_2))\,,$$

where the $K \in k$ is selected in a way to perform cancellation of high coefficients, so we have $\mathrm{HM}(h_1) \prec_0 \mathrm{HM}(h'_1)$. Such reduction is equivalent to plain reduction with high term cancellation extended with requirement for reductor's signature being smaller than the signature of labeled polynomial being reduced. Like in previous case the correctness check is performed directly.

Let's introduce a partial order $<_{\mathrm{H}}$ on H:

$$h_1 = (\sigma_1, p_1) <_{\mathrm{H}} h_2 = (\sigma_2, p_2) \overset{\text{def}}{\Leftrightarrow} \mathrm{HM}(p_1)\sigma_2 \prec_0 \mathrm{HM}(p_2)\sigma_1.$$

The elements with zero signature or zero high monomial are extremums:

$$\forall \sigma_1, \sigma_2, p_1, p_2 \ (0, p_1) \not<_{\mathrm{H}} (\sigma_2, p_2)\,,\ (\sigma_1, 0) \not>_{\mathrm{H}} (\sigma_2, p_2)\,.$$

Lemma 4. *Let* $h_1, h_2 \in H, t \in \mathbb{T}$. *Then* $h_1 >_{\mathrm{H}} h_2 \Leftrightarrow h_1 >_{\mathrm{H}} th_2$.

Deduced from the fact that multiplying one of the compared labeled polynomials by t leads to multiplying by t both sides in the definition of$>_{\mathrm{H}}$.

Lemma 5. *Let* $h_1, h_2 \in H, \mathrm{HM}(h_1) | \mathrm{HM}(h_2), \mathrm{HM}(h_2) \neq 0$. *Then signature-safe reduction* h_2 *by* h_1 *is possible iff*$h_1 >_{\mathrm{H}} h_2$.

Deduced from the fact that claims of both sides are equivalent to $\mathcal{S}(h_2) \succ_0 \mathcal{S}(h_1)\frac{\mathrm{HM}(h_2)}{\mathrm{HM}(h_1)}$.

Lemma 6. *Let* $h_1 \in H$ *be a result of signature-safe reduction of* h'_1 *by some other polynomial. Then* $h_1 <_H h'_1$.

Deduced from equality $\mathcal{S}(h_1) = \mathcal{S}(h'_1)$ and decreasing HM during reduction: $\mathrm{HM}(h_1) \prec_0 \mathrm{HM}(h'_1)$.

Lemma 7. *Let* $h_1 <_H h_2$ *be labeled polynomials. Then for* $\forall h_3 \in H \setminus \{(0,0)\}$ *at least one of the following two inequalities holds:* $h_1 <_H h_3$ *or* $h_3 <_H h_2$.

The lemma clause gives inequality

$$\mathrm{HM}(h_1)\,\mathcal{S}(h_2) \prec_0 \mathrm{HM}(h_2)\,\mathcal{S}(h_1) \tag{1}$$

which shows $\mathrm{HM}(h_2) \neq 0, \mathcal{S}(h_1) \neq 0$. Therefore for the special case $\mathrm{HM}(h_3) = 0$ we get $h_3 <_{\mathrm{H}} h_2$and for the case $\mathcal{S}(h_3) = 0$ we get $h_1 <_{\mathrm{H}} h_3$. For remaining generic non-zero case the inequality (1) can be multiplied by non-zero monomial $\mathrm{HM}(h_3)\,\mathcal{S}(h_3)$:

$$\mathrm{HM}(h_3)\,\mathcal{S}(h_3)\,\mathrm{HM}(h_1)\,\mathcal{S}(h_2) \prec_0 \mathrm{HM}(h_3)\,\mathcal{S}(h_3)\,\mathrm{HM}(h_2)\,\mathcal{S}(h_1). \tag{2}$$

So, the element $\mathrm{HM}(h_3)^2\,\mathcal{S}(h_2)\,\mathcal{S}(h_1) \in \mathbb{T}_0$ need to be $\succ_0$ than left side or $\prec_0$ than right side of inequality (2), and gives after cancellation one of the inequalities from the lemma statement.

Algorithm

Input: polynomial set $\{g_1, \ldots, g_m\}$ being a Gröbner basis; polynomial f.

Variables: R and B – subsets of H; $(\sigma, p') \in H$ – current step's labeled polynomial before reduction; (σ, p) – the same after reduction; r, b – elements of R and B

Result: Gröbner basis of ideal$I = (g_1, \ldots, g_m, f)$

SimpleSignatureGroebner$(\{g_1, \ldots, g_m\}, f)$

1. $R \leftarrow \{(\mathrm{HM}(g_1), 0), (\mathrm{HM}(g_2), 0), \ldots, (\mathrm{HM}(g_m), 0), (0, g_1), (0, g_2), \ldots, (0, g_m)\}$
2. $B \leftarrow \{\}$
3. $(\sigma, p') \leftarrow (1, f)$
4. **do forever:**
   (a) $p \leftarrow$ReduceCheckingSignatures(σ, p', R)
   (b) $R \leftarrow R \cup \{(\sigma, p)\}$
   (c) **if** $p \neq 0$**:**
      i. **for** $\{r \in R \,|\, r <_{\mathbf{H}} (\sigma, p), \mathrm{HM}(r) \neq 0\}$**:**
         A. $B \leftarrow B \cup \{\frac{\mathrm{LCM}(\mathrm{HM}(r), \mathrm{HM}(p))}{\mathrm{HM}(r)} r\}$
      ii. **for** $\{r \in R \,|\, r >_{\mathrm{H}} (\sigma, p)\}$**:**
         A. $B \leftarrow B \cup \{\frac{\mathrm{LCM}(\mathrm{HM}(r), \mathrm{HM}(p))}{\mathrm{HM}(p)} (\sigma, p)\}$
   (d) $B \leftarrow B \setminus \{b \in B \,|\, \exists r \in R\, r <_{\mathrm{H}} b \wedge \mathcal{S}(r) | \mathcal{S}(b)\}$
   (e) **if** $B \neq \varnothing$**:** $(\sigma, p') \leftarrow$ element of B with $\prec$-minimal signature
   (f) **else: break**
5. **return** $\{\mathrm{poly}(r) \,|\, r \in R\}$

ReduceCheckingSignatures(σ, p, R)

1. **do while** $\exists r \in R\, r >_{\mathrm{H}} (\sigma, p) \wedge \mathrm{HM}(r) | \mathrm{HM}(p)$**:**
   (a) $p \leftarrow$signature-safe reduce p by $>_{\mathrm{H}}$-maximal element r from the set in cycle clause
2. **return** p

Lemma 8. *All pairs from $\mathbb{T}_0 \times P$ appeared in the algorithm are labeled polynomials from $H \setminus \{(0, 0)\}$.*

The elements created before entering main cycle are labeled polynomials mentioned above as examples. All other labeled polynomials in the algorithm are created either with multiplication by $t \in \mathbb{T}$ or with signature-safe reduction, so they satisfy the correctness property and belongs to H.

The clauses of cycles extending B enforces the absence in B elements with zero signature or zero highest monomial. So, σ never can be 0 and the only R elements with zero signatures are $(0, g_1), ..., (0, g_m)$. Any labeled polynomial added to R can have zero highest monomial but R does not contain zero polynomial with zero signature.

Algorithm termination

Lemma 9. *At the any moment during the algorithm execution any labeled polynomial from B can be signature-safe reduced by some element of R.*

Labeled polynomials are added to B in a way ensuring existence at least one possible signature-safe reductor. The pair $(\sigma, p) \in R$ is such reductor for polynomials added in first **for** cycle, and $r \in R$ – for the polynomials added in the second cycle.

Lemma 10. *Before reduction of polynomial p' – at the step4a of any algorithm iteration – the signatures of elements $\{r \in R \,|\, r <_H (\sigma, p')\}$ does not divide σ.*

This holds at the first algorithm iteration because $\sigma = 1$ and R does not contain elements with signatures dividing 1. This holds during next iterations because the existence such elements in R would lead to removal (σ, p') from B during previous iterations at the step 4d.

Lemma 11. *After reduction of p' to p – at the step 4b of any algorithm iteration – the highest monomials of elements $\{r \in R \,|\, r >_{\mathrm{H}} (\sigma, p)\}$ does not divide* $\mathrm{HM}(p)$.

The cycle in the ReduceCheckingSignatures(σ, p, R) stops only when it achieves p for which there is no such elements in R .

Lemma 12. *After reduction of p' to p – at the step 4b of any algorithm iteration – no one from R elements has simultaneously a highest monomial dividing* $\mathrm{HM}(p)$ *and a signature dividing σ.*

The lemma 9 ensures that $p'$is reduced at least once, so $(\sigma, p') >_{\mathrm{H}} (\sigma, p)$. Now, by lemma7 for $\forall r \in R$ we have $r >_{\mathrm{H}} (\sigma, p)$ or $r <_{\mathrm{H}} (\sigma, p')$. These inequalities allow to apply either lemma 10 or lemma 11.

Theorem 13. *The algorithmSimpleSignatureGroebner$(\{g_1, \ldots, g_m\}, f)$ terminates*

To prove the termination we need to show that all **do** cycles stops after finite number of executions. In the cycle inside ReduceCheckingSignatures(σ, p, R) with non-zero p during every iteration we have $\mathrm{HM}(p)$ decrease according to $\prec_0$, which is possible only finite number of times. When p becomes zero it stops immediately because of $<_{\mathrm{H}}$-minimality of $(\sigma, 0)$.

The set $R \subset \mathbb{T}_0 \times P$ is extended every step of the main algorithm cycle. It can be splitted to $R_{*0} \cup R_{0*} \cup R_{**}$, where $R_{*0} \subset \mathbb{T} \times \{0\}, R_{0*} \subset \{0\} \times P \setminus \{0\}, R_{**} \subset \mathbb{T} \times P \setminus \{0\}$. R_{0*} does never extend because $\sigma \neq 0$. For sets R_{*0} and R_{**} we apply a method based on idea of monoid ideal introduced in [8] as "monoideal". Consider the following two sets which are monoideals: $L_{*0} = (\{\sigma \,|\, (\sigma, 0) \in R_{*0}\}) \subset \mathbb{T}$ and $L_{**} = (\{(\sigma, t) \,|\, \exists (\sigma, p) \in R_{**}\, t = \mathrm{HM}(p)\}) \subset \mathbb{T} \times \mathbb{T}$. Lemma 12 shows that elements being added to R expand either L_{*0} or L_{**} in every cycle iteration. The monoids $\mathbb{T}$ and $\mathbb{T} \times \mathbb{T}$ are isomorphic to $\mathbb{N}^n$ and $\mathbb{N}^{2n}$, so the Dickson's lemma can be applied to their monoideals. It states exactly the needed fact – only finite number of expansions is possible for such monoideals.

Correctness of output

Definition 14. *S-representation of $h \in H$* over set $\{r_i\} \subset H$ is an expression $\mathrm{poly}(h) = \sum_j K_j t_j \,\mathrm{poly}(r_{i_j})$, $K_j \in k, t_j \in \mathbb{T}, i_j \in \mathbb{N}$, such that $\forall j\ \mathrm{HM}(h) \succcurlyeq_0 \mathrm{HM}(t_j r_{i_j}), \mathcal{S}(h) \succcurlyeq_0 \mathcal{S}(t_j r_{i_j})$.

Lemma 15. *Let* $\mathrm{poly}(h) = \sum_j K_j t_j \,\mathrm{poly}(r_{i_j})$ *be S-representation of h. Then at least one j satisfies* $\mathrm{HM}(h) = \mathrm{HM}(t_j r_{i_j})$.

To get a j satisfying the equality we can take a value which gives the $\succ$-maximum of $\mathrm{HM}(t_j r_{i_j})$.

The next definition extends the notation of S-basis from [1]:

Definition 16. We call a labeled polynomial set $R \subset H$ *S-basis* (correspondingly *S_σ-basis*), if all elements of H (correspondingly $\{h \in H \mid \mathcal{S}(h) \prec_0 \sigma\}$) have S-representation over R.

Lemma 17. *Let $\sigma \succ_0 0, R = \{r_i\}$ be S_σ-basis and $h_1, h_2 \in H, \mathcal{S}(h_i) = \sigma$ be labeled polynomials, that can't be signature-safe reduced by R elements. Then* $\mathrm{HM}(h_1) = \mathrm{HM}(h_2)$ *and h_1 has an S-representation over $R \cup \{h_2\}$.*

We have from the definition of H that $\exists u_i \in P\ \mathrm{HM}(u_i) = \sigma, u_i f \equiv \mathrm{poly}(h_i) \pmod{I_0}, i = 1, 2$. It means that there exists a linear combination of $\mathrm{poly}(h_i)$ having signature $\prec_0 \sigma$. This can be written as:

$$\exists K \in k, v \in P\ \mathrm{HM}(v) = \sigma' \prec_0 \sigma, vf \equiv \mathrm{poly}(h_1) - K\,\mathrm{poly}(h_2) \pmod{I_0},$$

or in the terminology of labeled polynomials: $(\sigma', p') = (\sigma', \mathrm{poly}(h_1) - K\,\mathrm{poly}(h_2)) \in H$. From the definition of S_σ-basis and the property $\sigma' \prec_0 \sigma$ we conclude: $\exists r_j \in R, t \in \mathbb{T}\ \mathcal{S}(t r_j) \preccurlyeq_0 \sigma', \mathrm{HM}(t r_j) = \mathrm{HM}(p')$. So $\mathrm{HM}(h_i) \neq \mathrm{HM}(p'), i = 1, 2$, because in the case of equality r_j would be signature-safe reductor for h_i. It is possible only if $\mathrm{HM}(h_i)$ are canceled while subtraction with k-coefficient, what means that $\mathrm{HM}(h_1) = \mathrm{HM}(h_2)$. S-representation of h_1 is constructed by adding $K\,\mathrm{poly}(h_2)$ to S-representation of (σ', p').

Theorem 18. *Every iteration of the algorithm after step 4d the following invariant holds: for* $\forall \sigma \in \mathbb{T}, \sigma \prec$ *signatures of elements of* B*, exists* $r_\sigma \in R, t_\sigma \in \mathbb{T} : \mathcal{S}(t_\sigma r_\sigma) = \sigma$ *such that* $t_\sigma r_\sigma$ *can't be signature-safe reduced by* R*.*

The set $R_\sigma = \{r \in R \mid \mathcal{S}(r) | \sigma\}$ is not empty, because contains the element r_0 added during the first algorithm iteration with $\mathcal{S}(r_0) = 1$. Let r_σ be $<_{\mathrm{H}}$-minimal element of the set; take $t_\sigma = \frac{\sigma}{\mathcal{S}(r_\sigma)}$. Suppose that $t_\sigma r_\sigma$ can be signature-safe reduced by some $r_1 \in R$. This gives that $r_1 >_{\mathrm{H}} r_\sigma$ and both sides of inequality are non-zero. It means that during the iteration which inserts in R the last of $\{r_\sigma, r_1\}$ the set B was extended by labeled polynomial $t' r_\sigma$, where $t' = \frac{\mathrm{LCM}(\mathrm{HM}(r_1), \mathrm{HM}(r_\sigma))}{\mathrm{HM}(r_\sigma)}$ and $t' | t_\sigma$. So we have $\mathcal{S}(t' r_\sigma) | \mathcal{S}(t_\sigma r_\sigma) = \sigma \Rightarrow \mathcal{S}(t' r_\sigma) \preccurlyeq \sigma \prec$ signatures of elements of B. This signatures inequality implies that $t' r_\sigma$ can't be element of B during the current iteration and was removed at the step 4d of some previous iteration, so $\exists r_2 \in R\, r_2 <_{\mathrm{H}} t' r_\sigma, \mathcal{S}(r_2) | \mathcal{S}(t' r_\sigma)$. This is impossible, because the existence of $r_2 <_{\mathrm{H}} r_\sigma, r_2 \in R_\sigma$ contradicts $<_{\mathrm{H}}$-minimality of r_σ.

Theorem 19. *Every iteration of the algorithm after step 4d the following invariant holds:* $\forall h \in H, \mathcal{S}(h) \prec$ *signatures of elements of* B *has S-representation over* R*.*

Suppose that invariant breaks during some algorithm iteration and take the $\prec_0$-minimal σ that has non-empty corresponding set $V_\sigma \stackrel{\text{def}}{=} \{h \in H \mid h$ breaks invariant$, \mathcal{S}(h) = \sigma\}$. Then R is S_σ-basis. $\forall g \in I_0\ (0, g)$ has S-representation over $\{(0, g_1), ..., (0, g_m)\} \subset R$, so $\sigma \succ_0 0$. Select v_σ – one of the V_σ elements with $\prec_0$-minimal HM. It can't be signature-safe reduced by R because the reduction result v_1 would be element ofV_σ with $\mathrm{HM}(v_1) \prec_0 \mathrm{HM}(v_\sigma)$. Take $w_\sigma \stackrel{\text{def}}{=} t_\sigma r_\sigma$ from the invariant of theorem 18 and apply lemma 17 to v_σ, w_σ and R. The lemma says that v_σ has S-representation over $R \cup \{w_\sigma\}$. All entries of w_σ in the representation can be replaced by $t_\sigma r_\sigma$ to acquire S-representation ofv_σ over R only. It's existence leads to contradiction.

Lemma 20. *If* R *is S-basis, then* $\{\mathrm{poly}(r) \mid r \in R\}$ *is a Gröbner basis of ideal* I*.*

For $\forall p \in I$ we can take some $h = (\sigma, p) \in H$ and apply lemma 15 to it.

Theorem 21. *SimpleSignatureGroebner*$(\{g_1, \ldots, g_m\}, f)$ *returns Gröbner basis*

At the moment of algorithm termination $B = \varnothing$ so by theorem 19 R is S-basis.

Comparison with other algorithms

The presented algorithm belongs to the family of the Gröbner basis algorithms using signatures and being at some degree a modification of F5 algorithm from [4]. One of the main modification directions of F5 is simplifying and clarifying the connected theory usually bound with some thoughts about extending the area of inputs the algorithm can be applied to. Investigations in this direction can be found in [6, 11, 10]. The other direction is improving efficiency of computations by introducing criteria to detect and don't perform some unnecessary computations. It is studied in [2, 5, 3] and allows to perform computations in a way that reduces to the end only polynomials that are either new S-basis entries or corresponds to extending monoideal of signatures known to be zero polynomial signatures – called *syzygy signatures.* Generalization simultaneously using all criteria described in algorithms TRB-MJ and SB [7, 9] achieve even more efficiency because all discardings are performed before any computational heavy operations like polynomial reduction or computation highest monomial of S-polynomial – so the non-trivial computations are never become unnecessary because their results are never discarded.

All mentioned algorithms including original F5 use discarding criteria of two types: syzygy-based criteria and rewrite-like criteria, with separate proof of correctness for each type. The other common idea used in the algorithms are S-polynomials: even the algorithms that does not deal with S-polynomials directly make heavy usage of them in the correctness proof.

This paper describes an algorithm computing minimal S-basis and discarding computations with efficiency identical to TRB-MJ, but using the only one discarding criteria in step 4d, which is based on $<_{\mathrm{H}}$-ordering of R. The efficiency of different reductor selection strategies in ReduceCheckingSignatures(σ, p, R) is open question. The method presented in this paper is based on the same ordering of R and is identical for homogeneous case with the methods used in original F5. The correctness proof is

given without the use of S-polynomials and is formulated in a way that allows to apply the clear algebraic interpretation of signature-based algorithms from [11] to the presented algorithm.

Algorithm simplification lead to simplification of programming its implementation and debugging. It is achieved by the smaller number of objects involved in computation and use of the same order for discarding criteria and reductor selection. Simplicity of implementation and the absence of complex data structures allows quick algorithm integration with any computer algebra system that can work with polynomials. The author's implementation linked below was written from scratch in a 8 hours what is a lot smaller than the time author spent implementing other algorithms with a similar tools.

The algorithm was implemented in C++ using low-level functions from computer algebra system Singular 3-1-4 and open source codes of C. Eder (one of the authors of [3]) for implementing F5-like algorithms in this system. The source is contained in "ssg" function in a file available at https://github.com/galkinvv/Singular-f5-like/blob/ssg/kernel/kstd2.cc

Comparison of SimpleSignatureGroebner implementation with other Gröbner basis algorithms implemented by C. Eder gives practical checks for the following theoretical facts:

- algorithm SimpleSignatureGroebner correctly computes Gröbner basis;
- the number of polynomials in the result set is not greater than the number of polynomials in a result of other incremental algorithms that compute S-basis;
- the execution time is not greater than execution time of other signature-based incremental algorithms.

References

[1] A. Arri and J. Perry. The f5 criterion revised. *ArXiv e-prints*, December 2010.

[2] C. Eder and J. Perry. F5c: a variant of faugère's f5 algorithm with reduced gröbner bases. *ArXiv e-prints*, June 2009.

[3] C. Eder and J. Perry. Signature-based algorithms to compute gröbner bases. *ArXiv e-prints*, January 2011.

[4] Jean Charles Faugère. A new efficient algorithm for computing gröbner bases without reduction to zero (f5). In *Proceedings of the 2002 international symposium on Symbolic and algebraic computation*, ISSAC '02, pages 75–83, New York, NY, USA, 2002. ACM.

[5] Shuhong Gao, Yinhua Guan, and Frank Volny, IV. A new incremental algorithm for computing gröbner bases. In *Proceedings of the 2010 International Symposium on Symbolic and Algebraic Computation*, ISSAC '10, pages 13–19, New York, NY, USA, 2010. ACM.

[6] O. German. Proof of the faugère criterion for the f5 algorithm. *Mathematical Notes*, 88(4):502–510, 2010.

[7] L. Huang. A new conception for computing gröbner basis and its applications. *ArXiv e-prints*, December 2010.

[8] Martin Kreuzer and Lorenzo Robbiano. *Computational commutative algebra. 1.* Springer-Verlag, Berlin, 2000.

[9] B. Roune and M. Stillman. Practical gröbner basis computation. 2012.

[10] Y. Sun and D. Wang. The f5 algorithm in buchberger's style. *ArXiv e-prints*, June 2010.

[11] A. I. Zobnin. Generalization of the f5 algorithm for calculating gröbner bases for polynomial ideals. *Program. Comput. Softw.*, 36(2):75–82, March 2010.