Simple signature-based Groebner basis algorithm

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

14 мая 2012 г.

Аннотация

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]

Рассмотрим кольцо многочленов $P = k[x_1, \ldots, x_n]$ над полем k. Будем предполагать, что на моноиде его мономов $\mathbb{T}$ задан допустимый мономиальный порядок $\prec$. В этом кольце может быть поставлена задача вычисления базиса Грёбнера для произвольного идеала $(f_1, \ldots, f_l)$. Один из способов её решения инкрементальный: последовательно вычисляются базисы идеалов $(f_1, \ldots, f_i), i = 2 \ldots l$ на основе уже вычисленного для идеала $(f_1, \ldots, f_{i-1})$ базиса R_{i-1} и многочлена f_i. Представляемый алгоритм позволяет выполнить шаг такого вычисления. Таким образом, входные данные для алгоритма – это некоторый многочлен f и множество многочленов, обозначаемое $\{g_1, \ldots, g_m\}$, являющееся базисом Грёбнера идеала $I_0 = (g_1, \ldots, g_m)$. В качестве результата своей работы алгоритм должен построить множество многочленов R, являющееся базисом Грёбнера идеала $I = (g_1, \ldots, g_m, f)$. Поскольку случаи $f = 0 \Rightarrow I = I_0$ и $\exists i\, g_i \in k \Rightarrow I = P$ не представляют интереса, далее предполагается что $f \neq 0, \forall i\, g_i \notin k$. Заметим, что в отличии от алгоритма F5, описанного в [1], однородность многочленов не требуется.

Определения

Введём обозначения: $\mathbb{T}_0 = \mathbb{T} \cup \{0\}$ – расширенный нулём моноид мономов. Порядок $\prec$ продолжается с сохранением вполне упорядоченности на $\mathbb{T}_0$ как $\prec_0$ определением $\forall t \in \mathbb{T}\, t \succ_0 0$. Понятие делимости также расширяется на $\mathbb{T}_0$: $t_1 | t_2 \stackrel{\text{def}}{=} \exists t_3\, t_1 t_3 = t_2$. Для $p \in P, p \neq 0$ старшие по $\prec$

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

моном и коэффициент обозначим $\mathrm{HM}(p) \in \mathbb{T}$ и $\mathrm{HC}(p) \in k$. Для нуля – $\mathrm{HM}(0) \stackrel{\text{def}}{=} 0 \in \mathbb{T}_0$, $\mathrm{HC}(0) \stackrel{\text{def}}{=} 0 \in k$. Наименьшее общее кратное $t_1, t_2 \in \mathbb{T}$ обозначим $\mathrm{LCM}(t_1, t_2) \in \mathbb{T}$. Далее все определения даются для фиксированных I_0 и f:

Определение 1. *Отмеченным многочленом* называется пара $h = (\sigma, p) \in \mathbb{T}_0 \times P$, удовлетворяющая условию корректности: $\exists u \in P\ \mathrm{HM}(u) = \sigma, uf \equiv p \pmod{I_0}$. На отмеченные многочлены распространяются определения старшего монома $\mathrm{HM}(h) \stackrel{\text{def}}{=} \mathrm{HM}(p)$ и коэффициента $\mathrm{HC}(h) \stackrel{\text{def}}{=} \mathrm{HC}(p)$. Также определяются *сигнатура* $\mathcal{S}(h) \stackrel{\text{def}}{=} \sigma$ и вводится обозначение многочлена – второго элемента пары: $\mathrm{poly}(h) \stackrel{\text{def}}{=} p$. Множество отмеченных многочленов обозначается за $H \subset \mathbb{T}_0 \times P$. Тривиальными примерами отмеченных многочленов являются $(1, f)$ и $(0, g)$ для $g \in I_0$. Другим примером отмеченного многочлена является $(\mathrm{HM}(g), 0)$ для $g \in I_0$. Он корректен, поскольку в качестве u можно взять g.

Лемма 2. *Умножение для $h \in H, t \in \mathbb{T}$, заданное как $th \stackrel{\text{def}}{=} (t\sigma, tp) \in H$, корректно.*

Корректность определения проверяется явным нахождением u для h.

Определение 3. Если для некоторых $h'_1, h_2 \in H, t \in \mathbb{T}$ выполняется $\mathcal{S}(h'_1) \succ_0 \mathcal{S}(th_2), \mathrm{HM}(h'_1) = \mathrm{HM}(th_2) \neq 0$, то возможна *редукция h'_1 по h_2 с сохранением сигнатуры*, дающая в результате многочлен $h_1 \in H$, равный:

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

где коэффициент $K \in k$ взят так, чтобы при сложении сократились старшие мономы и выполнилось $\mathrm{HM}(h_1) \prec_0 \mathrm{HM}(h'_1)$. По сути такая редукция представляет из себя обычную редукцию многочлена с сокращением старшего монома, дополненную требованием того, что сигнатура редуктора меньше сигнатуры редуцируемого. Корректность проверяется как и выше.

Введём частичный порядок $<_{\mathrm{H}}$ на H:

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

Элементы с нулевой сигнатурой и старшим мономом оказываются экстремумами:

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

Лемма 4. *Пусть $h_1, h_2 \in H, t \in \mathbb{T}$. Тогда $h_1 >_{\mathrm{H}} h_2 \Leftrightarrow h_1 >_{\mathrm{H}} th_2$.*

Выводится из того, что умножение на t одного из сравниваемых отмеченных многочленов приводит к умножению на t обоих частей в определении $>_{\mathrm{H}}$.

Лемма 5. *Пусть $h_1, h_2 \in H, \mathrm{HM}(h_1) | \mathrm{HM}(h_2), \mathrm{HM}(h_2) \neq 0$. Тогда редукция h_2 по h_1 с сохранением сигнатуры возможна если и только если $h_1 >_{\mathrm{H}} h_2$.*

Следует из того что оба утверждения равносильны $\mathcal{S}(h_2) \succ_0 \mathcal{S}(h_1)\frac{\mathrm{HM}(h_2)}{\mathrm{HM}(h_1)}$.

Лемма 6. *Пусть $h_1 \in H$ – результат редукции h_1' с сохранением сигнатур по некоторому многочлену. Тогда $h_1 <_H h_1'$.*

Следует из $\mathcal{S}(h_1) = \mathcal{S}(h_1')$ и уменьшения НМ при редукции: $\mathrm{HM}(h_1) \prec_0 \mathrm{HM}(h_1')$.

Лемма 7. *Пусть $h_1 <_H h_2$ отмеченные многочлены. Тогда $\forall h_3 \in H \setminus \{(0,0)\}$ выполняется хотя бы одно из двух неравенств: $h_1 <_H h_3$ или $h_3 <_H h_2$.*

Из условия леммы известно, что

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

откуда $\mathrm{HM}(h_2) \neq 0, \mathcal{S}(h_1) \neq 0$. Поэтому, если $\mathrm{HM}(h_3) = 0$, имеем $h_3 <_\mathrm{H} h_2$, а если $\mathcal{S}(h_3) = 0$ – то $h_1 <_\mathrm{H} h_3$. Иначе можно домножить неравенство (1) на ненулевой элемент $\mathrm{HM}(h_3)\,\mathcal{S}(h_3)$:

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

Поэтому $\mathrm{HM}(h_3)^2\,\mathcal{S}(h_2)\,\mathcal{S}(h_1) \in \mathbb{T}_0$ будет или $\succ_0$ левой или $\prec_0$ правой части неравенства (2), и после сокращения даст эквивалентное утверждению леммы неравенство.

Алгоритм

Вход: многочлены $\{g_1, \ldots, g_m\}$, образующие базис Грёбнера; многочлен f.

Переменные: R и B – подмножества H; $(\sigma, p') \in H$ – отмеченный многочлен текущего шага до редукции; (σ, p) – он же после редукции; r, b – элементы R и B

Результат: базис Грёбнера идеала $I = (g_1, \ldots, g_m, f)$

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** $B \neq \varnothing$**:** $(\sigma, p') \leftarrow$ элемент B с $\prec$-минимальной сигнатурой

(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) $p \leftarrow$редуцировать p с сохранением сигнатуры по $>_{\mathrm{H}}$-максимальному элементу r среди указанных в условии цикла

2. **return** p

Лемма 8. *Все пары из* $\mathbb{T}_0 \times P$ *в алгоритме – элементы* $H \setminus \{(0, 0)\}$.

Элементы, формируемые до начала главного цикла, являются рассмотренными выше примерами отмеченных многочленов. Все остальные отмеченные многочлены в алгоритме формируются или умножением на $t \in \mathbb{T}$ или редукцией с сохранением сигнатуры, поэтому они корректны и лежат в H.

Условия циклов, расширяющих B, таковы, что в B нет ни нулевых сигнатур, ни нулевых старших мономов. Поэтому σ никогда не обращается в 0 и нулевые сигнатуры в R лишь у элементов $(0, g_1), ..., (0, g_m)$. Нулевой старший моном может быть у любого многочлена, добавляемого в R, а нулевых многочленов с одновременно нулевой сигнатурой в R нет.

Остановка алгоритма

Лемма 9. *В любой момент работы алгоритма любой отмеченный многочлен из* B *может быть редуцирован с сохранением сигнатуры по некоторому элементу* R.

Отмеченные многочлены добавляются в B таким образом, чтобы иметь хотя бы один подходящий редуктор. $(\sigma, p) \in R$ является таким редуктором при добавлении в первом цикле **for**, $r \in R$ – во втором.

Лемма 10. *До редукции многочлена* p'*, то есть на шаге 4а любой итерации алгоритма, сигнатуры элементов* $\{r \in R \,|\, r <_H (\sigma, p')\}$ *не делят* σ.

На первой итерации алгоритма это выполняется, поскольку $\sigma = 1$ и R не содержит элементы с сигнатурами, делящими 1. На последующих итерациях это выполнено, поскольку если бы в R существовали такие элементы, то (σ, p') был бы убран из B в предыдущей итерации на шаге 4d.

Лемма 11. *После редукции многочлена p' до p, на шаге 4b любой итерации алгоритма, старшие мономы элементов $\{r \in R \,|\, r >_{\mathrm{H}} (\sigma, p)\}$ не делят* $\mathrm{HM}(p)$.

Вытекает из того, что цикл в ReduceCheckingSignatures(σ, p, R) останавливается по достижении p, для которого такие элементы в R не существуют.

Лемма 12. *После редукции многочлена p' до p, на шаге 4b любой итерации алгоритма, элементы R не могут одновременно иметь старшие мономы, делящие* $\mathrm{HM}(p)$, *и сигнатуры, делящие σ.*

В силу леммы 9 будет произведена хотя бы одна редукция p', поэтому $(\sigma, p') >_{\mathrm{H}} (\sigma, p)$. Отсюда по лемме 7 для $\forall r \in R$ имеем $r >_{\mathrm{H}} (\sigma, p)$ или $r <_{\mathrm{H}} (\sigma, p')$. Выполнение одного из неравенств позволяет применить одну из лемм 10 и 11.

Теорема 13. *Алгоритм SimpleSignatureGroebner$(\{g_1, \ldots, g_m\}, f)$ останавливается*

Для доказательства остановки нужно показать, что все циклы **do** выполняются лишь конечное число раз. В ReduceCheckingSignatures(σ, p, R) при ненулевых p на каждой итерации $\mathrm{HM}(p)$ уменьшается по $\prec_0$, что возможно лишь конечное число раз. При обнулении p он завершится в силу $<_{\mathrm{H}}$-минимальности $(\sigma, 0)$.

На каждом шаге основного цикла пополняется множество $R \subset \mathbb{T}_0 \times P$. Оно может быть разбито как $R_{*0} \cup R_{0*} \cup R_{**}$, где $R_{*0} \subset \mathbb{T} \times \{0\}, R_{0*} \subset \{0\} \times P \setminus \{0\}, R_{**} \subset \mathbb{T} \times P \setminus \{0\}$. R_{0*} не пополняется в силу $\sigma \neq 0$. Для R_{*0} и R_{**} применим подход, основанный на понятии идеалов моноидов, предложенном в [2] как "monoid ideal". Рассмотрим следующие множества, являющиеся идеалами моноидов: $L_{*0} = (\{\sigma \,|\, (\sigma, 0) \in R_{*0}\}) \subset \mathbb{T}$ и $L_{**} = (\{(\sigma, t) \,|\, \exists (\sigma, p) \in R_{**}\, t = \mathrm{HM}(p)\}) \subset \mathbb{T} \times \mathbb{T}$. В силу леммы 12 добавляемые в R элементы расширяют на каждом шаге L_{*0} или L_{**}. Поскольку моноиды $\mathbb{T}$ и $\mathbb{T} \times \mathbb{T}$ изоморфны $\mathbb{N}^n$ и $\mathbb{N}^{2n}$, к их идеалам может быть применена лемма Диксона, которая и утверждает, что расширение может происходить лишь конечное число раз.

Корректность результата

Определение 14. *S-представлением $h \in H$* над множеством $\{r_i\} \subset H$ будем называть выражение $\mathrm{poly}(h) = \sum_j K_j t_j \,\mathrm{poly}(r_{i_j})$, $K_j \in k, t_j \in \mathbb{T}, i_j \in \mathbb{N}$, такое что $\forall j\ \mathrm{HM}(h) \succcurlyeq_0 \mathrm{HM}(t_j r_{i_j}), \mathcal{S}(h) \succcurlyeq_0 \mathcal{S}(t_j r_{i_j})$.

Лемма 15. *Пусть $\mathrm{poly}(h) = \sum_j K_j t_j \,\mathrm{poly}(r_{i_j})$ – S-представление для h. Тогда для хотя бы одного j достигается* $\mathrm{HM}(h) = \mathrm{HM}(t_j r_{i_j})$.

В качестве такого j можно взять то, на котором достигается $\succ$-максимум $\mathrm{HM}(t_j r_{i_j})$.

Следующее определение расширяет понятие S-базиса из работы [3]:

Определение 16. Назовём $R \subset H$ *S-базисом* (соответственно S_σ*-базисом*), если все элементыH (соответственно $\{h \in H \mid \mathcal{S}(h) \prec_0 \sigma\}$) имеют S-представление над R.

Лемма 17. *Пусть* $\sigma \succ_0 0, R = \{r_i\} - S_\sigma$*-базис и выбраны* $h_1, h_2 \in H, \mathcal{S}(h_i) = \sigma$*, которые не редуцируются по* R *с сохранением сигнатуры. Тогда* $\mathrm{HM}(h_1) = \mathrm{HM}(h_2)$ *и у* h_1 *есть S-представление над* $R \cup \{h_2\}$.

Из определения H имеем $\exists u_i \in P\ \mathrm{HM}(u_i) = \sigma, u_i f \equiv \mathrm{poly}(h_i) \pmod{I_0}, i = 1, 2$. Значит некоторой линейной комбинации $\mathrm{poly}(h_i)$ сопоставляется $\prec_0 \sigma$ сигнатура:

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

то есть $(\sigma', p') = (\sigma', \mathrm{poly}(h_1) - K\,\mathrm{poly}(h_2)) \in H$. Из определения S_σ-базиса и $\sigma' \prec_0 \sigma$ вытекает $\exists r_j \in R, t \in \mathbb{T}\ \mathcal{S}(tr_j) \preccurlyeq_0 \sigma', \mathrm{HM}(tr_j) = \mathrm{HM}(p')$. Отсюда $\mathrm{HM}(h_i) \neq \mathrm{HM}(p'), i = 1, 2$, иначе r_j редуцировало бы h_i с сохранением сигнатуры. Значит, $\mathrm{HM}(h_i)$ сокращаются при вычитании с k-коэффициентом, что даёт $\mathrm{HM}(h_1) = \mathrm{HM}(h_2)$. S-представление h_1 получается добавлением $K\,\mathrm{poly}(h_2)$ к S-представлению (σ', p').

Теорема 18. *На каждой итерации алгоритма после шага 4d выполнен инвариант: для* $\forall \sigma \in \mathbb{T}, \sigma \prec$ *сигнатур элементов* B*, найдутся* $r_\sigma \in R, t_\sigma \in \mathbb{T} : \mathcal{S}(t_\sigma r_\sigma) = \sigma$ *и* $t_\sigma r_\sigma$ *не редуцируется по* R *с сохранением сигнатуры.*

Множество $R_\sigma = \{r \in R \mid \mathcal{S}(r) | \sigma\}$ непусто, так как содержит добавленный на первой итерации элемент r_0 с $\mathcal{S}(r_0) = 1$. Обозначим за r_σ его $<_\mathrm{H}$-минимальный элемент; положим $t_\sigma = \frac{\sigma}{\mathcal{S}(r_\sigma)}$. Предположим, что $t_\sigma r_\sigma$ может быть редуцирован с сохранением сигнатуры относительно некоторого $r_1 \in R$. Отсюда следует, что $r_1 >_\mathrm{H} r_\sigma$, а также что они не нулевые. Значит на той же итерации, когда в R был добавлен последний из $\{r_\sigma, r_1\}$, в множество B был добавлен многочлен $t' r_\sigma$, где $t' = \frac{\mathrm{LCM}(\mathrm{HM}(r_1), \mathrm{HM}(r_\sigma))}{\mathrm{HM}(r_\sigma)}$, причём $t' | t_\sigma$. Отсюда $\mathcal{S}(t' r_\sigma) | \mathcal{S}(t_\sigma r_\sigma) = \sigma \Rightarrow \mathcal{S}(t' r_\sigma) \preccurlyeq \sigma \prec$ сигнатур элементов B. В силу этого неравенства на сигнатуры получается, что $t' r_\sigma$ уже не может быть элементом B, а значит был выкинут на шаге 4d одной из итераций, то есть $\exists r_2 \in R\, r_2 <_\mathrm{H} t' r_\sigma, \mathcal{S}(r_2) | \mathcal{S}(t' r_\sigma)$. Это невозможно, поскольку влечёт $r_2 <_\mathrm{H} r_\sigma, r_2 \in R_\sigma$, что противоречит $<_\mathrm{H}$-минимальности r_σ.

Теорема 19. *На каждой итерации алгоритма после шага 4d выполнен инвариант:* $\forall h \in H, \mathcal{S}(h) \prec$ *сигнатур элементов* B*, имеет S-представление над* R.

Предположим нарушение инварианта на какой-то итерации и рассмотрим $\prec_0$-минимальную σ, для которой непусто $V_\sigma \stackrel{\mathrm{def}}{=} \{h \in H \mid h$ нарушает инвариант, $\mathcal{S}(h) =$

$\sigma\}$. Тогда $R - \mathrm{S}_\sigma$-базис. $\forall g \in I_0$ $(0, g)$ имеют S-представления над $\{(0, g_1), ..., (0, g_m)\} \subset R$, поэтому $\sigma \succ_0 0$. Выберем v_σ – один из элементов V_σ с $\prec_0$-наименьшим НМ. Он не может быть редуцирован с сохранением сигнатуры по R, поскольку результат редукции v_1 был бы элементом V_σ с $\mathrm{HM}(v_1) \prec_0 \mathrm{HM}(v_\sigma)$. Возьмём $w_\sigma \stackrel{\text{def}}{=} t_\sigma r_\sigma$ из инварианта теоремы 18 и применим лемму 17 к v_σ, w_σ и R. Получим что v_σ имеет S-представление над $R \cup \{w_\sigma\}$. Вхождения w_σ в нём можно заменить на $t_\sigma r_\sigma$, получив представление v_σ над R, что приводит к противоречию.

Лемма 20. *Если R – S-базис, то $\{\mathrm{poly}(r) \,|\, r \in R\}$ является базисом Грёбнера идеала I.*

Для $\forall p \in I$ можно взять некоторый $h = (\sigma, p) \in H$ и применить лемму 15.

Теорема 21. *SimpleSignatureGroebner$(\{g_1, \ldots, g_m\}, f)$ возвращает базис Грёбнера*

К моменту остановки $B = \varnothing$, значит по теореме 19 R – S-базис.

Сравнение с аналогами

Представленный алгоритм принадлежит к семейству алгоритмов вычисления базисов Грёбнера, использующих сигнатуры, которые вычисляют S-базис и в той или иной степени являются модификациями алгоритма F5 из [1]. Одно из основных направлений его модификации – упрощение теоретических обоснований и расширение области применимости – представлено в [4,5,6]. Другое – повышение эффективности путём ввода дополнительных критериев отбрасывания некоторых вычислений – описывается в [7,8,9] и позволяет проводить вычисления так, чтобы до конца редуцировались лишь многочлены, являющиеся новыми элементами S-базиса или дающие новую сигнатуру нулевого многочлена, расширяющую идеал моноида, содержащий такие сигнатуры, называемые также *сигнатурами сизигий*. Обобщение с одновременным применением всех критериев в алгоритмах TRB-MJ и SB [10,11] позволяет добиться большей эффективности благодаря тому, что все отбрасывания применяются до проведения таких вычислительно трудоёмких операций, как редукция многочлена или подсчёт старшего монома S-пары, – в результате не оказывается, что результаты каких-то вычислений были отброшены.

Во всех упомянутых алгоритмах, включая немодифицированный F5, формулируется два типа критериев отброса: критерии, связанные с сизигиями, и критерии перезаписи, корректность каждого из которых доказывается независимо. Также, даже в алгоритмах, не вычисляющих S-полиномы явно, теоретическое обоснование корректности алгоритма на них опирается.

Данная работа описывает алгоритм вычисляющий минимальный S-базис и осуществляющий отброс вычислений не менее эффективно, чем в TRB-MJ, но использующий лишь единственный критерий отброса на шаге 4d,

основанный на $<_{\mathrm{H}}$-упорядочивании множества R. Вопрос наиболее эффективного способа выбора редуктора в ReduceCheckingSignatures(σ, p, R) является открытым. Представленный в этой работе способ выбора основан на всё том же упорядочении R и совпадает для случая однородных многочленов со способом выбора, применявшемся в алгоритме F5. Теоретическое обоснование сформулировано без S-полиномов и позволяет применять к нему простую алгебраическую интерпретацию из [5].

Упрощение формулировки алгоритма повлекло значительное уменьшение времени на его реализацию и отладку на компьютере по сравнению с аналогами, как за счёт меньшего количества множеств, так и за счёт общего для критериев отбрасывания и процедуры редукции порядка. Простота реализации и нетребовательность к структурам данных позволяет за небольшое время внедрять эффективную версию алгоритма в любую систему компьютерной алгебры. Реализация, упоминаемая ниже, была создана автором за 8 часов, что на порядок меньше, чем время, затраченное автором на экспериментальные реализации других алгоритмов в подобных условиях. Доказательство, основанное на инвариантах в терминах S-представлений, позволило сделать работу алгоритма более прозрачной с алгебраической точки зрения и потенциально расширяемым на объекты, обобщающие кольцо многочленов над полем.

Алгоритм был реализован на C++ с использованием функций ядра программного комплекса Singular 3-1-4 и открытых наработок Кристиана Эдера (одного из авторов [8]) по реализации F5-подобных алгоритмов на этом ядре. Исходный код реализации содержится в функции ssg файла, доступного по адресу https://github.com/galkinvv/Singular-f5-like/blob/ssg/kernel/kstd2.cc

Сравнение реализации SimpleSignatureGroebner с другими алгоритмами вычисления базисов Грёбнера, реализованных Кристианом Эдером подтвердили следующие соображения:

- алгоритм SimpleSignatureGroebner корректно вычисляет базис Грёбнера;
- результат содержит не большее число многочленов, чем результат других инкрементальных алгоритмов, возвращающих S-базис;
- время работы алгоритма оказывается не больше, чем у других инкрементальных алгоритмов, основанных на сигнатурах.