Простой итеративный алгоритм вычисления базисов Грёбнера, основанный на сигнатурах

19 апреля 2012 г.

Аннотация

Данная работа описывает алгоритм вычисления базисов Грёбнера, основанный на использовании отмеченных многочленов и идеях из алгоритма F5. Отличительными особенностями рассматриваемого алгоритма являются простота реализации и доказательств корректной работы.

1 Определения

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 представляет интереса, далее предполагается что $f \neq 0, \forall i\, g_i \notin k$.

Введём обозначения: $\mathbb{T}_0 = \mathbb{T} \cup \{0\}$ – моноид мономов, расширенный нулём. 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 мономов 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 $\prec$ моном и коэффициент обозначим $\mathrm{HM}(p) \in \mathbb{T}$ и $\mathrm{HC}(p) \in k$. Для нуля – $\mathrm{HM}(0) \stackrel{\text{def}}{=} 0 \in \mathbb{T}_0$, $\mathrm{HC}(0) \stackrel{\text{def}}{=} 0 \in k$. За $\mathrm{LCM}(t_1, t_2) \in \mathbb{T}$ обозначим наименьшее общее кратное $t_1, t_2 \in \mathbb{T}$. 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: $\exists u \in P : \mathrm{HM}(u) = \sigma, uf = p \mod I_0$. 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: $\mathrm{poly}(h) \stackrel{\mathrm{def}}{=} p$. Множество отмеченных многочленов обозначается за $H \subset \mathbb{T}_0 \times P$. Тривиальными примерами отмеченных многочленов являются $(1, f)$ и $(0, g)$ для $g \in I_0$.

Лемма 2. *Умножение для $h \in H, t \in \mathbb{T}$, заданное как $th \stackrel{\mathrm{def}}{=} (t\sigma, tp) \in H$, 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, чтоб 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 top-редукцию многочлена, дополненную требованием того, что сигнатура редуктора меньше сигнатуры редуцируемого. Корректность проверяется как и выше.

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

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

Элементы с нулевой сигнатурой и старшим мономом оказываются экстремумами: $\forall \sigma_1, \sigma_2, p_1, p_2\ (0, p_1) \not<_{\mathrm{gvw}} (\sigma_2, p_2)\,,\ (\sigma_1, 0) \not>_{\mathrm{gvw}} (\sigma_2, p_2)$.

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

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

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 $h_1 >_{\mathrm{gvw}} h_2$.*

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 $h_1 <_{\mathrm{gvw}} h'_1$.*

Следует из $\mathcal{S}(h_1) = \mathcal{S}(h'_1)$ и уменьшения HM при редукции: $\mathrm{HM}(h_1) \prec_0 \mathrm{HM}(h'_1)$.

Лемма 7. *Пусть $h_1 <_{\mathrm{gvw}} h_2$ отмеченные многочлены. Тогда для любого отмеченного многочлена $h_3 \neq (0, 0)$ выполняется по крайней мере одно из двух неравенств $h_1 <_{\mathrm{gvw}} h_3$ и $h_3 <_{\mathrm{gvw}} h_2$.*

Из условия леммы известно, что

$$\mathrm{HM}(h_1)\,\mathcal{S}(h_2) \prec_0 \mathrm{HM}(h_2)\,\mathcal{S}(h_1) \tag{1.1}$$

откуда $\mathrm{HM}(h_2) \neq 0, \mathcal{S}(h_1) \neq 0$. Поэтому, если $\mathrm{HM}(h_3) = 0$, имеем $h_3 <_{\mathrm{gvw}} h_2$, а если $\mathcal{S}(h_3) = 0$ – то $h_1 <_{\mathrm{gvw}} h_3$. Иначе можно домножить неравенство (1.1) на ненулевой элемент $\mathrm{HM}(h_3)\,\mathcal{S}(h_3)$:

$$\mathrm{HM}(h_3)\,\mathcal{S}(h_3)\,\mathrm{HM}(h_1)\,\mathcal{S}(h_2) \prec_0 \mathrm{HM}(h_3)\,\mathcal{S}(h_3)\,\mathrm{HM}(h_2)\,\mathcal{S}(h_1). \tag{1.2}$$

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 (1.2), и после сокращения даст эквивалентное утверждению леммы неравенство.

2 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$(\{g_1, \ldots, g_n\}, f)$

1. $R \leftarrow \{(0, g_1), (0, g_2), \ldots, (0, g_n)\}$
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** $\{r \in R | r <_{\text{gvw}} (\sigma, p), \text{HM}(r) \neq 0\}$**:**
         A. $B \leftarrow B \cup \{\frac{\text{LCM}(\text{HM}(r), \text{HM}(p))}{\text{HM}(r)} r\}$
      ii. **for** $\{r \in R | r >_{\text{gvw}} (\sigma, p)\}$**:**
         A. $B \leftarrow B \cup \{\frac{\text{LCM}(\text{HM}(r), \text{HM}(p))}{\text{HM}(p)} (\sigma, p)\}$
   (d) $B \leftarrow B \setminus \{b \in B | \exists r \in R,\, r <_{\text{gvw}} b \wedge \mathcal{S}(r) | \mathcal{S}(b)\}$
   (e) **if** $B \neq \emptyset$**:** $(\sigma, p') \leftarrow$ элемент B с $\prec$-минимальной сигнатурой
   (f) **else: break**
5. **return** $\{\text{poly}(r) | r \in R\}$

ReduceCheckingSignatures(σ, p, R)

1. **do while** $\exists r \in R | r >_{\text{gvw}} (\sigma, p) \wedge \text{HM}(r) | \text{HM}(p)$**:**
   (a) $p \leftarrow$редуцировать p с сохранением сигнатуры по $>_{\text{gvw}}$-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 тривиальных 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 $(0, g_1), (0, g_2), ..., (0, g_n)$. Нулевой старший моном может быть у любого многочлена, добавляемого в R, а нулевых многочленов с одновременно нулевой сигнатурой в R нет.

3 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, чтоб иметь хотя бы один подходящий редуктор. При добавлении в первом цикле **for** редуктором является $(\sigma, p) \in R$, во втором $r \in R$.

Лемма 10. *До редукции многочлена p', то есть на шаге 4a любой итерации алгоритма, сигнатуры элементов $\{r \in R | r <_{\text{gvw}} (\sigma, p')\}$ 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 $\{r \in R | r >_{\text{gvw}} (\sigma, p)\}$ не делят* $\text{HM}(p)$.

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* $\text{HM}(p)$ *и сигнатуры, делящие σ.*

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

Теорема 13. *Описанный алгоритм останавливается на любых входных данных*

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 $\text{HM}(p)$ уменьшается относительно $\prec_0$, что возможно лишь конечное число раз. При обнулении p он завершится в силу $<_{\text{gvw}}$-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 остальных рассмотрим идеалы моноидов: $L_{*0} = (\{\sigma | (\sigma, 0) \in R_{*0}\}) \subset \mathbb{T}$ и $L_{**} = (\{(\sigma, t) | \exists (\sigma, p) \in R_{**}, t = \text{HM}(p)\}) \subset \mathbb{T} \times \mathbb{T}$. В силу леммы 12 добавляемые элементы расширяют на каждом шаге L_{*0} или L_{**}, что по лемме Диксона может происходить лишь конечное число раз.

4 Корректность

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

Следующее определение расширяет понятие S-базиса из работы Arri and Perry [1]:

Определение 15. Множество отмеченных многочленов называется *$\mathcal{S}_\sigma$-базисом*, если относительно него любой допустимый отмеченный многочлен сигнатуры $\prec_0 \sigma$ имеет S-представление. Если S-представление имеет допустимый многочлен любой сигнатуры, то множество называется просто *$\mathcal{S}$-базисом.*

Лемма 16. *Если $R = \{r_i\}$ – S-базис, то R является базисом Грёбнера идеала I.*

Вытекает из того что для $p \in I$ можно взять некоторый допустимый $h = (\sigma, p)$ в S-представлении которого найдётся r_{i_j}, такой что $\mathrm{HM}(p) = t_j \,\mathrm{HM}(r_{i_j})$.

Для доказательства корректности алгоритма будем доказывать следующую теорему, из которой будет следовать корректность ответа в момент остановки алгоритма, поскольку при $B = \emptyset$ её инвариант станет выполняться для всех многочленов.

Теорема 17. *На каждой итерации алгоритма после шага 4d выполнен следующий инвариант: любой допустимый отмеченный многочлен, сигнатура которого $\prec$ чем сигнатуры всех элементов B имеет S-представление относительно R.*

Предположим, что утверждение теоремы неверно на определённой итерации алгоритма. Зафиксируем множества R и B на момент после шага 4d этой итерации, и все S-представления будем рассматривать относительно этого R. Пусть V – множество допустимых многочленов, для которых не выполняется инвариант, а $V_0 = \{v \in V | \, \mathcal{S}(v) = \sigma_0\}$ – подмножество с минимальной сигнатурой. Все элементы I_0 имеют S-представления относительно $\{(0, g_1), (0, g_2), ..., (0, g_n)\} \subset R$, поэтому $\sigma_0 \neq 0$. Выберем $v_0 = (\sigma_0, q_0) \in V_0$ – многочлен с наименьшим старшим мономом. Он не может быть редуцирован с сохранением сигнатуры относительно R, поскольку иначе результат его редукции v_1 являлся бы элементом V_0 с $\mathrm{HM}(v_1) \prec \mathrm{HM}(v_0)$.

Рассмотрим множество $\{r \in R | \, \mathcal{S}(r) | \sigma_0\}$. Поскольку $\sigma_0 \neq 0$, оно содержит добавленный на первой итерации элемент r с $\mathcal{S}(r) = 1$ и, значит, непусто. Рассмотрим его $<_{\mathrm{gvw}}$-минимальный элемент r_0 и обозначим $t_0 = \frac{\sigma_0}{\mathcal{S}(r_0)}$. Предположим, что отмеченный многочлен $t_0 r_0 = (\sigma_0, t_0 p_0)$ может быть редуцирован с сохранением сигнатуры относительно некоторого $r_1 \in R$. Отсюда следует, что $r_1 >_{\mathrm{gvw}} r_0$, а также что они не нулевые. Значит на той же итерации, когда в R был добавлен последний из $\{r_0, r_1\}$, в множество B был добавлен многочлен $t' r_0$, где $t' = \frac{\mathrm{LCM}(\mathrm{HM}(r_1), \mathrm{HM}(r_0))}{\mathrm{HM}(r_0)}$, причём $t' | t_0$. Отсюда $\mathcal{S}(t' r_0) | \, \mathcal{S}(t_0 r_0) = \sigma_0 \Rightarrow \mathcal{S}(t' r_0) \preccurlyeq \sigma_0$. Поскольку σ_0 – сигнатура v_0, для которого не выполняется инвариант, она $\prec$ чем сигнатуры элементов B – то есть $t' r_0$ был выкинут на шаге 4d одной из итераций, что влечёт существование $r_2 \in R, r_2 <_{\mathrm{gvw}} t' r_0, \mathcal{S}(r_2) | \, \mathcal{S}(t' r_0)$. Это невозможно, поскольку влечёт $r_2 <_{\mathrm{gvw}} r_0, \mathcal{S}(r_2) | \, \mathcal{S}(t_0 r_0) = \sigma_0$, что противоречит $<_{\mathrm{gvw}}$-минимальности r_0.

Таким образом показано, что $t_0 r_0 = (\sigma_0, t_0 p_0)$ и $v_0 = (\sigma_0, q_0)$ не могут быть редуцированы относительно R с сохранением сигнатуры, при этом первый из них имеет тривиальное S-представление $t_0 r_0$, а второй не имеет S-представления. Из допустимости следует, что их разность $w = (\sigma_1, q_0 - K t_0 p_0)$ с некоторым коэффициентом $K \in k$ есть допустимый многочлен сигнатуры $\sigma_1 \prec \sigma_0$. $w \notin V$ в силу малости сигнатуры и имеет S-представление $w = \sum_j K_j t_j r_{i_j}$. Значит $\exists j | \, \mathrm{HM}(t_j r_{i_j}) = \mathrm{HM}(w), \mathcal{S}(t_j r_{i_j}) \preccurlyeq \mathcal{S}(w) = \sigma_1 \prec \sigma_0$. Если бы $\mathrm{HM}(w) \in \{\mathrm{HM}(v_0), \mathrm{HM}(t_0 r_0)\}$, то r_{i_j} был бы сохраняющим сигнатуру редуктором для v_0 или $t_0 r_0$, что невозможно. Значит старшие мо-

номы сокращаются при вычитании, то есть НМ(v_0) = НМ($t_0 r_0$). Отсюда выводим S-представление: $v_0 = Kt_0 r_0 + \sum_j K_j K t_j r_{i_j}$. Противоречие.

4.1 Связь с аналогами

Представленный алгоритм связан с двумя уже известными алгоритмами, основанных на сигнатурах – алгоритм G2V из работы Gao et al. [4] и версию F5, опубликованную в работе Arri and Perry [1]. Оба они в определённом смысле являются модификациями простого F5, впервые представленного в работе Faugère [3], причём модификации направлены на сокращение числа редукций многочленов, занимающих большую часть времени в процессе вычисления базисов Грёбнера. Первый из них отличается от немодифицированной версии тем, что не использует явного вычисления S-полиномов до проведения редукции и вводит специальный критерий «super-topreducible» для отбрасывания некоторых многочленов после их редукции. Второй применяет расширенный критерий, отбрасывая до редукции те из посчитанных S-полиномов, которые не смогут удовлетворять критерию «primitve S-irreducble» после редукции. Эти методики позволяют ускорить работу алгоритма, однако в обоих случаях в определённый момент отбрасывается многочлен, полученный в процессе предыдущих вычислений путём редукции или создания S-полинома.

Практически во всех алгоритмах, основанных на сигнатурах, в том числе в исходном алгоритме F5 применяется и более эффективный тип критериев: критерии отбрасывания S-пар, не требующие вычислений с многочленами для своей проверки – их отличия в различных алгоритмах подробно разобраны в работе Eder and Perry [2]. Однако вопрос об их сравнительной эффективности остаётся неясным как с теоретической, так и с эмпирической точек зрения – на различных примерах большее или меньшее преимущество могут иметь различные подходы.

В алгоритме SimpleSignatureGroebner используется подход, который можно интерпретировать как объединение всего вышеуказанного: S-полиномы не вычисляются явно и их отбрасывание осуществляется на основе критерия, записываемого в точности как критерий второго алгоритма. При этом для проверки на шаге 4d элемента B достаточно знать лишь его сигнатуру и старший моном. Поскольку каждый элемент B получается как домноженный на моном m элемент $r \in R$ их можно хранить в виде пары (m, r), не выполняя без необходимости операции домножения всего многочлена на моном.Таким образом, для отброшенных элементов B не производится никаких операций, сложность которых пропорциональна длине многочлена. Заметим, что применяемое во втором алгоритме вычисление старшего монома S-пары в общем случае напротив имеет именно такую сложность. Таким образом, возможна реализация алгоритма SimpleSignatureGroebner, в которой все операции над многолченами производятся лишь для многочленов, попадающих в результирующее множество. Результирующее множество не является минимальным базисом Грёбнера, но при этом в определённом смысле минимально. Это минимальное множество, удовлетворяющее определению S-базиса.

Отсюда можно сделать вывод о некоторой полноте критериев отбрасывания: не может существовать критерия, который бы позволил убрать какие-либо операции редукции многочленов из алгоритма не добавив новых редукций и не лишив результат свойства быть базисом S-представлений. Из доказательств корректности многих алгоритмов, основанных на сигнатурах следует что их результат содержит такое

множество, поэтому представленный алгоритм является в указанном смысле оптимальным алгоритмом, находящим минимальный базис S-представлений для последовательности идеалов, расширяемых одним многочленом на каждом шаге.

Эта оптимальность не глобальна – могут существовать более эффективные модификации алгоритма, которые не просто убирают вычисления, а заменяют одни вычисления над полиномами другими. К примеру, вопрос наиболее эффективного способа выбора редуктора в процедуре Редуцировать_с_учётом_сигнатур является открытым. Представленный способ выбора, основанный на $<_{\text{gvw}}$-сравнении наиболее близок к способу выбора, применявшегося в оригинальном алгоритме F5.

Алгоритм был реализован на C++ с использованием функций ядра программного комплекса Singular 3-1-4 и открытых наработок Christian Eder 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 с другими алгоритмами вычисления базисов Грёбнера, реализованных Christian Eder подтвердили следующие теоретические соображения:

- Алгоритм SimpleSignatureGroebner корректно вычисляет базис Грёбнера
- Возвращаемое множество содержит не большее число многочленов, чем множество возвращаемое другими инкрементальными алгоритмами, про которые известно, что они возвращают S-представление
- Время работы алгоритма оказывается несколько меньше, чем у других инкрементальных алгоритмов, основанных на сигнатурах.

Список литературы

[1] Alberto Arri and John Perry. The f5 criterion revised. *J. Symb. Comput.*, 46(9): 1017–1029, sep 2011. ISSN 0747-7171.

[2] Christian Eder and John Edward Perry. Signature-based algorithms to compute gröbner bases. In *Proceedings of the 36th international symposium on Symbolic and algebraic computation*, ISSAC '11, pages 99–106, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0675-1.

[3] 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. ISBN 1-58113-484-3.

[4] Shuhong Gao, Yinhua Guan, and Frank Volny, IV. A new incremental algorithm for computing groebner bases. In *Proceedings of the 2010 International Symposium on Symbolic and Algebraic Computation*, ISSAC '10, pages 13–19, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0150-3.