ISSN 1810-0198 Вестник ТГУ, т.17, вып.2, 2012

УДК 519.688

ОБ ОДНОМ МАТРИЧНОМ ПОДХОДЕ К ПОСТРОЕНИЮ БАЗИСОВ ГРЕБНЕРА

© **И. А. Борисов**

Ключевые слова: базисы Грёбнера, матричная редукция, система компьютерной алгебры Mathpar.

Мы рассматриваем матричный подход к построению базисов Грёбнера полиномиальных идеалов и новые структуры данных для эффективной организации вычислений в системе компьютерной алгебры Mathpar.

1 Введение

Базисы Гребнера полиномиальных идеалов имеют широкое применение. Существуют различные алгоритмы построения базисов Гребнера [1], [2]. Матричные алгоритмы и способы их распареллеливания приводятся в статьях [3], [4], [5].

В данной статье рассматриваются новые структуры данных, которые разработаны специально для матричного подхода к вычислению базисов полиномиальных идеалов.

В параграфе 2 вводятся основные определения. В параграфе 3 описывается матричный алгоритм Фужера `F4`. В параграфе 4 предлагаются структуры данных для хранения полиномов, которые приспособлены для матричных алгоритмов, используемых при вычислении полиномиальных базисов. В параграфе 5 приводится пример с использованием этих структур данных.

2 Базисы Гребнера

Определение 1. *Пусть R — коммутативное кольцо. Подмножество $I \subset R$ называется* идеалом, *если выполнены следующие условия [2]: 1) $0 \in I$; 2) если $a, b \in I$, то $a + b \in I$; 3) если $a \in I, b \in R$, то $a \cdot b \in I$.*

Если идеал I порождён полиномами $g_1, g_2, \ldots, g_n$, то множество $G = \{g_1, g_2, \ldots, g_n\}$ называется *базисом* (системой образующих) идеала I ($I = \langle G \rangle$).

Определение 2. *Идеал $I \subset k[x_1, \ldots, x_n]$ называется* мономиальным *($\langle x^\alpha : \alpha \in A \rangle$), если существует подмножество $A \subset Z^n_{\geqslant 0}$ такое, что I состоит из всех конечных сумм вида $\sum_{\alpha \in A} h_\alpha x^\alpha$, где $h_\alpha \in k[x_1, \ldots, x_n]$ [2].*

Определение 3. Мономиальным упорядочением *на* $k[x_1, \ldots, x_n]$ *называется бинарное отношение* $>$ *на множестве* $Z^n_{\geqslant 0}$, *обладающее следующими свойствами [2]: 1)* $>$ *является линейным упорядочением на* $Z^n_{\geqslant 0}$; *2) если* $\alpha > \beta$ *и* $\gamma \in Z^n_{\geqslant 0}$, *то* $\alpha+\gamma > \beta+\gamma$; *3)* $>$ *вполне упорядочивает* Z^n_0.

Относительно выбранного упорядочения будем пользоваться следующими обозначениями: 1) $HT(f)$ — *старший терм, т.е. старший моном без числового коэффициента*; 2) $HC(f)$ — *старший коэффициент*; 3) $HM(f)$ — *старший моном, т.е.* $HM(f) = HC(f)HT(f)$.

Определение 4. *Полином* f редуцируется *к полиному* h *по модулю* G, *если в* G *существует некоторый полином* g, *такой что* $HM(f) = HM(g) \cdot m$, *где* $m \in R$ *и* $h = f - g \cdot m$. Будем обозначать $f \underset{G}{\to} h$.

Определение 5. *Множество* $G \subset I$ *называется* базисом Гребнера *идеала* I, *если идеал, порождённый множеством старших мономов полиномов* G, *совпадает с идеалом, порожденным старшими мономами* I: $\langle HM(G)\rangle = \langle HM(I)\rangle$.

Для того, чтобы множество G из идеала I являлось базисом Гребнера, необходимо и достаточно, чтобы любой полином из I редуцировался по модулю G [3].

Б. Бухбергер доказал, что можно проверять редукцию не всех $f \in I$, а только его некоторого подмножества и сформулировал алгоритмическое определение базиса Гребнера [4].

Определение 6. *S-полиномом, порождённым* f *и* g, *называется полином вида:* $S(f,g) = \frac{LCM(HT(f),HG(g))}{LM(f)} \cdot f - \frac{LCM(HT(f),HT(g))}{LM(g)} \cdot g$.

Теорема 1 (Бухбергера). *Если для всех пар* $(f,g) \in G$ *их S-полиномы редуцируются к нулю по модулю* G, *то* G *является базисом Гребнера.*

Эта теорема лежит в основе алгоритма Бухбергера.

3 Матричный алгоритм вычисления базисов Гребнера

В 1999 г. Ж. К. Фужер представил эффективный матричный алгоритм построения базисов Гребнера F_4 [5].

Основные отличия от алгоритма Бухбергера: 1) Представление полиномов: вместо множества полиномов используется матрица коэффициентов этих полиномов. 2) Составление S-полиномов: алгоритм Фужера позволяет не вычислять S-полиномы полностью, а добавлять в матрицу их части для дальнейшего составления линейных комбинаций. 3) Редукция S-полиномов: редукция выполняется за один шаг с помощью приведения матрицы коэффициентов полиномов к ступенчатому виду. Чтобы эта операция была эквивалентна редукции, в матрицу добавляются строки, которые соответствуют полиномам базиса с некоторыми множителями.

Для построения матрицы необходимо задать мономиальное упорядочение и поставить в соответствие каждому моному каждого полинома коэффициент матрицы.

Пример соответствия матрицы множеству полиномов приведён на рис. 1.

На рис. 2, 3, 4 приведён алгоритм Фужера.

	z^3	z	y^2	yx^2
p_1	0	2	0	1
p_2	1	0	−3	0
p_3	−10	0	0	0

Рис. 1: Соответствие матрицы множеству полиномов $\{ p_1 = 2z + yx^2, p_2 = z^3 - 3y^2, p_3 = -10z \}$

function F4(F) ▷ F — исходный идеал
  $G = F$
  $P = \{(f, g) : f, g \in G, f \neq g\}$ ▷ Получение всех пар из G
  while $P \neq \varnothing$ **do**
    $Sel = Select(P)$ ▷ Выбор и удаление критических пар
    $LR = \{(left, right) : p, p \in Sel\}$ ▷ Получение левых и правых частей
S-полиномов пар
    $reduced =$ **reduction**(LR, G)
    for all $r \in reduced$ **do**
      if $HT(r) \notin HT(G)$ **then**
        for all $g \in G$ **do** ▷ Обновление списка пар
          $P = P \cup (r, g)$
        end for
      end if
    end for
  end while
  return G
end function

Рис. 2: Алгоритм F4

4 Структуры данных для матричного алгоритма

Существующая реализация содержит преобразование множества полиномов в матрицу коэффициентов и восстановление множества полиномов из матрицы коэффициентов и набора термов на каждой итерации вычисления базиса Гребнера. Данный подход является неэффективным, так как при каждой операции получения матрицы коэффициентов из множества полиномов выполняется выделение и заполнение двумерных массивов большого размера, и затем при восстановлении полиномов из матрицы производится поиск термов, нужных для получения каждого полинома, из списка всех термов всех полиномов базиса.

Для ускорения матричного алгоритма предлагается специальная структура данных, которая позволяет не совершать упомянутые выше преобразования, а производит все необходимые действия только над матрицей коэффициентов. Она описана в классе `PolynomialList`.

Класс `PolynomialList` содержит методы, требуемые на различных этапах исполнения матричного алгоритма. Основными операциями являются: создание объекта `PolynomialList`

```
function REDUCTION( (LR, G) )
    PP = symPP(LR, G)                                   ▷ препроцессинг
    reduced = row echelon form w.r.t. PP    ▷ преобразовываем полиномы в матрицу,
приводим ее к ступенчатому виду и восстанавливаем полиномы
    for all r ∈ reduced do
        if HT(r) ∉ PP then          ▷ возвращаем только редуцированные полиномы
            res = res ∪ r
        end if
    end for
    return res
end function
```

Рис. 3: Функция редукции

```
function SYMPP( LR, G )
    F = {t * f, (t, f) ∈ LR}
    Done = HT(F)
    while Done ≠ T(F) do
        Done = Done ∪ {t, t ∈ T(F)/Done}                ▷ выбираем терм t
        if t is top reducible by G then ▷ если t редуцируется каким-либо полиномом из
G
            r = reductor of t
            m = t / HT(r)
            res = res ∪ {m · r}
        end if
    end while
    return res
end function
```

Рис. 4: Функция построения матрицы

из массива полиномов, получение и выбор критических пар, умножение терма на полином и добавление полинома в существующий объект. Главные методы класса приведены на рис. 5, 6, 7.

Множество полиномов представляется следующим образом: все термы всех полиномов хранятся в сортированном списке (`AllTerms` типа `TermList`); сами полиномы — в строках двух двумерных массивов, один из которых содержит индексы термов в списке (`ColIndexes` — массив целых чисел), а другой хранит соответствующие коэффициенты (`Rows` — массив типа `Element`).

Термы помещаются в специальный объект класса `TermList`, который обеспечивает постоянное упорядочение термов по убыванию и обновление индексов существующих термов при добавлении новых.

Во время выполнения матричного алгоритма существуют три объекта `PolynomialList`. Первый объект используется для хранения текущего состояния базиса. Второй объект содержит левые и правые части S-полиномов критических пар. Третий объект служит для подготовки к редукции, в него помещаются выбранные на данном шаге критические

```
function PolynomialList.add(p)
    T = getAllTerms(p)                                          ▷ Получаем все термы.
    NewIndexes, UpdatedIndexes = AllTerms.addAll(T)   ▷ Добавляем термы в список
всех термов, получая индексы новых элементов и обновленные индексы.
    Rows.add(getAllCoeffs(p))                  ▷ Добавляем новую строку с коэффициентами
полинома.
    ColIndexes.add(NewIndexes)                 ▷ Добавляем новую строку с индексами термов.
    updateIndexes(UpdatedIndexes)              ▷ Обновляем изменившиеся индексы термов.
end function
```

Рис. 5: PolynomialList. Добавление полинома

```
function PolynomialList.multOnTerm(polIndex, t)
    for all currTerm = AllTerms[ColIndexes_i] do     ▷ Умножаем все термы полинома
на данный терм.
        NewIndex, UpdatedIndexes = AllTerms.add(currTerm * t)
        updateIndexes(UpdatedIndexes)
    end for
end function
```

Рис. 6: PolynomialList. Умножение полинома на терм

```
function TermList.add(t)
    newIndex = getInsertIndex(t)   ▷ Получаем индекс для вставки (двоичный поиск).
    insert(t, newIndex)
    updatedIndexes = updateIndexes(newIndex)          ▷ Обновляем индексы термов,
младших t.
    return (newIndex, updatedIndexes)
end function
```

Рис. 7: TermList. Добавление терма

пары и полиномы, полученные в результате выполнения препроцессинга.

5 Примеры

В примерах используется обратный лексикографический порядок $z > y > x$.

На рис. 8 приведена схема представления множества полиномов $\{ p_1 = 2z + yx^2, p_2 = z^3 - 3y^2, p_3 = -10z \}$ в виде объекта `PolynomialList`.

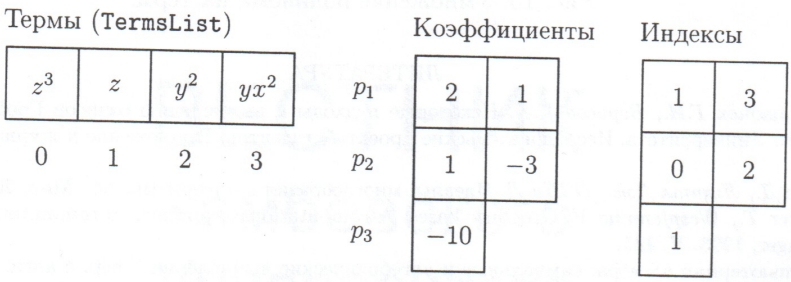

Рис. 8: Представление множества полиномов в виде `PolynomialList`

На рис. 9 показано состояние объекта после добавления нового полинома $p_4 = 5z^2 - 2y^2$, на рис. 10 — после умножения полинома p_3 на терм x^2.

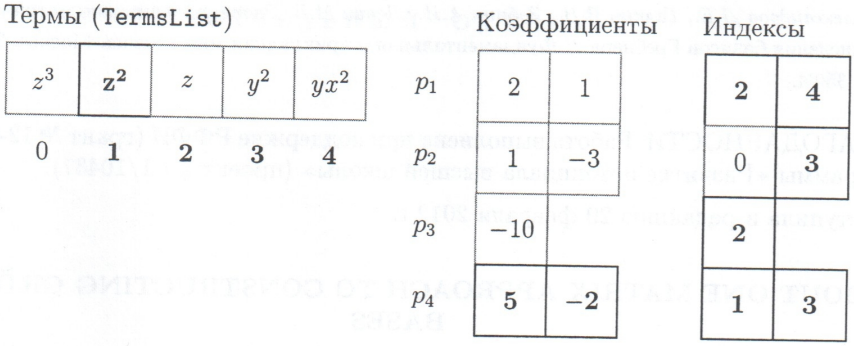

Рис. 9: Добавление полинома в объект `PolynomialList`

6 Заключение

В дальнейшем планируется развитие матричного алгоритма и его распараллеливание за счет матричных операций для исполнения на параллельных вычислительных машинах с распределенной памятью.

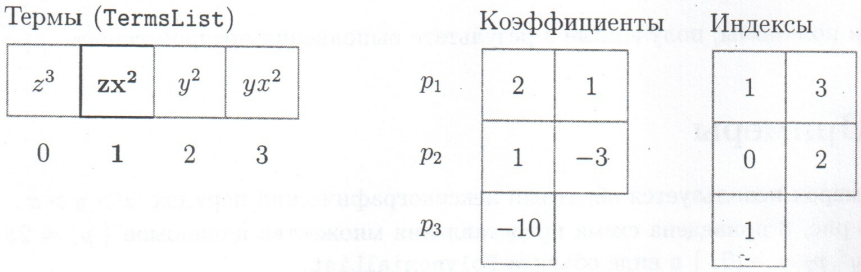

Рис. 10: Умножение полинома на терм

ЛИТЕРАТУРА

1. *Малашонок Г.И., Борисов И.А.* Некоторые подходы к вычислению базисов Гребнера // Вестник Тамбовского Университета. Исследовательские проекты студентов. Приложение к журналу. 2011. С. 191-194.

2. *Кокс Д., Литтл Дж., О'Ши Д.* Идеалы, многообразия и алгоритмы. М.: Мир, 2000.

3. *Becker T., Wespfenning V.* Groebner bases. A computational approach to commutative arlgebra. New York: Springer, 1993. V. 141.

4. Компьютерная алгебра: символьные и алгебраические вычисления / пер. с англ. *под ред. Б. Бухбергера, Дж. Колинза, Р. Лооса.* М.: Мир, 1986.

5. *Faugere J.-C.* A new efficient algorithm for computing Groebner bases (F4) // J. Pure Appl. Algebra. 1999. V. 139. No 1–3. P. 61–88.

6. *Малашонок Г.И., Старов М.В., Борисов И.А.* К параллельному вычислению базисов Гребнера // Вестник Тамбовского университета. Серия Естественные и технические науки. Тамбов. Т. 15. Вып. 6. 2010. С. 1755-1760.

7. *Старов М.В.* Реализация метода Фужера // Вестник Тамбовского университета. Серия Естественные и технические науки. Тамбов. Т. 14. Вып. 4. 2009. С. 802-803.

8. *Александров Д.Е., Галкин В.В., Зобнин А.И., Левин М.В.* Распараллеливание матричных алгоритмов вычисления базисов Гребнера // Фундаментальная и прикладная математика. Москва. Т. 14. Вып. 4. 2008. С. 35-64.

БЛАГОДАРНОСТИ: Работа выполнена при поддержке РФФИ (грант № 12-07-00755-а) и программы «Развитие потенциала высшей школы» (проект 2.1.1/10437).

Поступила в редакцию 20 февраля 2012 г.

ABOUT ONE MATRIX APPROACH TO CONSTRUCTING GRÖBNER BASES

© **Ivan Andreevich Borisov**

Tambov State University named after G.R. Derzhavin, Internatsionalnaya, 33, Tambov, 392000, Russia, Post-graduate Student of Mathematical Analysis Department, e-mail: ivan@iborisov.ru

Key words: Gröbner bases, matrix reduction, Mathpar computer algebra system.

We describe matrix approach to constructing Gröbner bases of polynomial ideals and data structures needed to efficient execution in computer algebra system Mathpar.