УДК 519.688

К ПАРАЛЛЕЛЬНОМУ ВЫЧИСЛЕНИЮ БАЗИСОВ ГРЕБНЕРА

© Г.И. Малашонок, М.В. Старов, И.А. Борисов

Ключевые слова: вычисление базисов Гребнера; параллельный алгоритм; модулярный метод; метод гомоморфных образов; кластерные вычисления.

Рассматривается параллельный алгоритм вычислений базисов Гребнера. Алгоритм основан на матричном алгоритме Ж.Ш. Фужера F4. Параллельную часть алгоритма составляет приведение разреженных матриц к ступенчатому виду. Используется модулярный алгоритм приведения к ступенчатому виду и его параллельная реализация. Приводятся результаты вычислительных экспериментов на кластере MVS100K Межведомственного Суперкомпьютерного Центра РАН.

1. Введение

Задача эффективного вычисления стандартных базисов полиномиальных идеалов является одной из центральных задач компьютерной алгебры и вычислительной алгебраической геометрии. Алгоритм вычисления стандартных базисов полиномиальных идеалов был предложен Б. Бухбергером ещё в середине 1960-х гг. [1]. Он же предложил их называть базисами Грёбнера, по имени своего научного руководителя, который поставил задачу вычисления стандартных базисов полиномиальных идеалов.

Базисы Гребнера позволяют искать решения систем нелинейных алгебраических уравнений от нескольких переменных, раскладывать на множители целые числа, решать задачу принадлежности полинома идеалу и другие алгоритмические задачи в полиномиальных идеалах.

Алгоритмы вычисления базисов Гребнера имеются во всех достаточно мощных системах компьютерной алгебры. Они применяется во многих прикладных задачах [2].

Мы будем развивать идеи, заложенные в алгоритме F4, который был предложен французским математиком Ж.Ш. Фужером в 1999 г. Алгоритм Фужера положил начало матричному подходу в алгоритмах вычисления базисов Гребнера. За десять лет он хорошо зарекомендовал себя на практике как очень продуктивный подход, позволяющий получать самые быстрые на сегодня алгоритмы.

Главное отличие алгоритма Фужера от традиционных методов заключается в том, что процесс редукции многочленов сводится к задаче приведения большой разреженной матрицы к ступенчатому виду. Методы работы с разреженными матрицами могут быть реализованы эффективно. Таким образом, происходит замена последовательной попарной редукции списка S-полиномов одновременной редукцией сразу всего списка за счет использования матричных алгоритмов. Подмножество S-полиномов может выбираться по-разному. Этот выбор для разных задач может по-разному влиять на эффективность алгоритма.

В настоящей работе рассматривается один из вариантов матричного подхода, который предназначен для параллельного вычисления базисов Гребнера [4–7]. Параллельное вычисление ступенчатого вида матрицы основано на методе гомоморфных образов, что сводит

решение вычислительной задачи в кольце Q к одновременному решению этой задачи на множестве конечных простых полей и последующему восстановлению решения в исходной области Q. Распараллеливание происходит за счет параллельных вычислений в каждом конечном поле. В следующем параграфе мы напомним известные сведения об алгоритмах вычисления базисов Гребнера.

2. Алгоритм Бухбергера

Бухбергер разработал алгоритм вычисления базиса Гребнера.

Будем рассматривать идеалы в кольце многочленов $\mathbb{Q}[x_1, \ldots, x_n]$. Будем называть термом выражение $x_1^{a_1} \ldots x_n^{a_n}$, где $a_i \geq 0$. Каждому терму можно сопоставить набор степеней переменных $(a_1 \ldots a_n)$, каждое из которых является целым неотрицательным числом. Множество всех термов n переменных будем обозначать T.

Обозначим через $\succ$ линейный порядок на множестве термов, удовлетворяющий следующему условию. Для произвольного множества термов t_1, t_2, t_3 из условия $t_1 \succ t_2$ следует $t_1 t_3 \succ t_2 t_3$.

Пусть f — ненулевой полином и G — множество полиномов из $F[x_1, \ldots, x_n]$.

1. $hT(f)$ — старший терм полинома f, $hT(G)$ — множество старших термов множества G.

2. $hM(f)$ — старший моном полинома f, $hM(G)$ — множество старших мономов множества G.

3. $hC(f)$ — коэффициент при старшем мономе полинома f, $hC(G)$ — коэффициенты при старших мономах множества G.

О п р е д е л е н и е 1. S-полиномом двух полиномов f и g будем называть комбинацию:

$$spol(f,g) = hC(g)\frac{lcm(f,g)}{hT(f)}f - hC(f)\frac{lcm(f,g)}{hT(g)}g.$$

О п р е д е л е н и е 2. Пусть G — конечное множество полиномов, T – множество термов. Если полином f может быть записан в виде

$$f = \sum_j t_j g_j + h, \quad t_j \in T, \quad g_j \in G,$$

при этом все старшие термы полиномов $t_j g_j$ различны и ни один терм полинома h не входит в множество $T \times hT(G)$, то будем записывать

$$f \to_G h$$

и будем говорить, что f редуцируется множеством G к полиному h. Если $h = 0$, то будем говорить, что f *редуцируется множеством* G.

О п р е д е л е н и е 3. Пусть I идеал, G – конечное множество полиномов. Множество G называется стандартным базисом, или базисом Гребнера, идеала I, если каждый полином из идеала I редуцируется множеством G, $\forall f \in I : f \to_G 0$.

Приведем критерий того, что порождающее множество идеала является базисом Гребнера этого идеала.

Т е о р е м а 1. (Критерий базиса Гребнера) *Пусть G – конечное множество, порождающее идеал I. Тогда множество $G = \{g_1, \ldots, g_n\}$ является базисом Гребнера идеала I в том и только в том случае, когда $spol(g_i, g_j) \to_G 0$ для всех пар $i \neq j$.*

Из этой теоремы непосредственно следует алгоритм построения базиса Гребнера. Он заключается в том, что пока множество полиномов F, заданное изначально и порождающее идеал, не является базисом Гребнера, нужно добавлять в него S-полиномы каждой пары полиномов, редуцируя предварительно эти S-полиномы по всему множеству F. В найденном базисе Гребнера можно редуцировать каждый полином по множеству всех остальных для минимизации числа полиномов.

Для усовершенствования этого алгоритма Бухбергер предложил использовать два критерия, благодаря которым можно не рассматривать пары, S-полином которых заведомо редуцируется к нулю.

Критерий 1: Пусть дано конечное множество $G = \{g_1, \ldots, g_n\}$, и пусть $g_i, g_j \in G$ таковы, что

$$lcm(hT(g_i), hT(g_j)) = hT(g_i)hT(g_j),$$

т. е. старшие термы полиномов g_i и g_j взаимно просты. Тогда $spol(g_i, g_j) \to_G 0$.

Критерий 2: Если для пары полиномов $g_1, g_2 \in G$ существует полином $g \in G$ такой, что $g_1 \neq g$, $g_2 \neq g$, $HT(g)$ является делителем $lcm(hT(g_1), hT(g2))$ и пары полиномов (g_1, g) и (g_2, g) уже рассмотрены и их S-полиномы уже включены в базу, то пару полиномов (g_1, g_2) можно не рассматривать, т. к. ее S-полином редуцируется к нулю.

3. Алгоритм Фужера F4

Основная идея алгоритма F4 заключается в замене последовательной редукции списка S-полиномов одновременной редукцией всего списка сразу за счет использования матричных алгоритмов. Шаг редукции множества S-полиномов сводится к задаче вычисления ступенчатого вида разреженных матриц большого размера. Строки этой матрицы образуют как редуцируемые полиномы из списка S-полиномов, так и редукторы, которые получаются из списка всех отобранных полиномов, домножением на подходящие термы. Элементами одной строки этой матриц являются коэффициенты одного полинома.

Конечно и здесь можно использовать два критерия Бухбергера, чтобы не рассматривать лишние пары и не искать их S-полиномы. Важным моментом при реализации алгоритма является выбор оптимальной стратегии отбора списка S-полиномов для редукции. Фужер предложил стратегию, согласно которой выбираются S-пары с наименьшей общей степенью – «нормальную стратегию».

Алгоритм F4.
Input: массив полиномов $F = \{f_1, \ldots, f_n\}$.
Output: массив полиномов G — базис Гребнера идеала $I = \langle f_1, \ldots, f_n \rangle$.
$G = F$;
B = **Update** (B, F, G);
while $B \neq 0$ **do**
  P = **SelectedPairs** (B)
  F = **Reduction** (P, G)
  $G = G \cup F$
  B = **Update** (B, F, G)
return G;

Метод $Update(B, F, G)$ строит новый список S-пар на основании текущего множества образующих G, множества новых полиномов F и текущего списка пар B, учитывая критерии Бухбергера. В начальный момент $G = F$ и список пар B пуст.

Метод $SelectedPairs(B)$ выбирает список пар из множества всех пар B, используя нормальную стратегию. Отобранные пары называются критическими парами.

Reduction (P, G)
Input: множество пар P и список редукторов G.
Output: массив полиномов F.
F = **SymbolicPreprocessing** (P, G);
M = **PolynomsToMatrix** (F);
$\widetilde{M}$ = **MatrixToEchelonForm** (M);
$\widetilde{F}$ = **MatrixToPolynoms** $(\widetilde{M})$;
$res = \varnothing$;
for $(f$ **in** $\widetilde{F})$
  if $($ **hT** $(f) \notin$ **hT** $(F))$ **then**
    $res = res \cup f$;
return res;

Метод $PolynomsToMatrix(F)$ строит матрицу из списка полиномов F.
Метод $MatrixToEchelonForm(M)$ приводит матрицу M к ступенчатому виду.
Метод $MatrixToPolynoms(M)$ преобразует матрицу M в массив полиномов.

SymbolicPreprocessing
Input: множество пар P и список редукторов G.
Output: массив полиномов F.
for $(p$ **in** $P)$
  $res = res \cup$ **lcm** $(p)/$ **hT**(**left** $(p))*$ **left** (p);
  $res = res \cup$ **lcm** $(p)/$ **hT**(**right** $(p))*$ **right** (p);
$done = \varnothing$;
$terms = \varnothing$;
updateTermsWithoutHT $(res, terms, done)$;
while $(terms \neq \varnothing)$
  $term =$ **first** $(terms)$;
  $terms = terms/term$;
  $done = done \cup term$;
  for $(g$ **in** $G)$
    if isReduction $(term, g)$ **then**
      $p = term/$ **hT** $(g) * g$;
      updateTermsWithoutHT $(p, terms, done)$;
      $res = res \cup p$;
      break;
    end if
return res;

Метод $updateTermsWithoutHT(pols, terms, done)$ добавляет все нестаршие термы полиномов $pols$ в список термов $terms$, которые отсутствуют в массивах термов $terms$ и $done$.

Распараллеливание этого алгоритма осуществляется за счет распараллеливания задачи нахождения ступенчатого вида одной матрицы, поскольку вычисление ступенчатого вида матрицы занимает основную часть времени работы алгоритма.

4. Параллельный алгоритм вычисления ступенчатого вида матрицы

Параллельное вычисление ступенчатого вида матрицы основано на методе гомоморфных образов. Метод гомоморфных образов сводит решение вычислительной задачи в коль-

це R к одновременному решению этой задачи на множестве факторколец и последующему восстановлению решения в исходной области R. Распараллеливание происходило за счет параллельных вычислений по каждому модулю. Для вычисления в конечном поле ступенчатой формы матрицы используется блочно-рекурсивный алгоритм [10].

Количество простых модулей, необходимое для каждой из получаемых матриц, оценивается неравенством Адамара [11]. При восстановлении полученной матрицы ступенчатого вида в кольце целых чисел используются только те модули, для которых были получены ступенчатые формы, имеющие одинаковые позиции ведущих элементов. Модули берутся с небольшим запасом, так чтобы призведение всех использованых модулей удовлетворяло оценке Адамара.

За номер процессора отвечает переменная $rank$.

За количество процессоров отвечает переменная $size$.

Для вычисления количества модулей и самих модулей используется метод **getPrimeNumbers**$(M, rank)$. Этот метод возвращает массив целых чисел на каждом процессоре.

Метод **EchelonForm**(M , $modul$) считает ступенчатый вид матрицы M по модулю $modul$.

Метод **DataGather**($echelonForms$, $recoveryData$, i) разбивает множество всех ненулевых элементов в ступенчатой матрице $echelonForms$ на $size$ равных частей, посылает всем процессорам их части множества ненулевых элементов. При этом в процессор с номером $rank$ передается часть массива под номером $rank$. В этом же методе происходит прием в область $recoveryData$ со всех процессоров соответствующих частей множества ненулевых элементов, отосланных в данный процессор.

Метод **RecoveryNewton**($recoveryData$, $moduls$) восстанавливает с помощью китайской теоремы об остатках массив целых чисел $recoveryData$ по модулям $moduls$.

Метод **DataGatherv**($data$, $result$) собирает на нулевом процессоре со всех процессоров соответствующие части множества ненулевых элементов ступенчатой матрицы (массивы $data$) в массив $result$.

Input: $M \in Z^{n \times m}$ — данная матрица;
Output: Ступенчатый вид матрицы M;

```
echelonForms = ∅
moduls =getPrimeNumbers (M, rank);
for ( modul in moduls )
        echelonForms = echelonForms∪ EchelonForm( M , modul );
DataGather( echelonForms , recoveryData );
data = RecoveryNewton( recoveryData , moduls );
DataGatherv( data , result );
```

5. Эксперименты с параллельным алгоритмом вычисления ступенчатого виды матрицы

Эксперимент проводился на кластере МВС-100К в МСЦ РАН. В эксперименте использовались плотные матрицы размера 512×512 с 5-разрядными коэффициентами. Результаты экспериментов приведены в табл. 1.

Таблица 1

Время вычисления ступенчатого вида матрицы с использованием алгоритма, основанного на методе гомоморфных образов

Количество процессоров k	1	2	4	8	16	32	256
Время t_k**, с**	1780.4	898.5	454.4	234.2	121.0	61.8	15.4
Эффективность α_k		98.1%	97.3%	94.3%	91.4%	89.7%	44.9%

В таблице 1 приводятся время вычисления ступенчатого вида матрицы и эффективность распараллеливания для разного числа вычислительных узлов. Эффективность принимается равной 100%, если при увеличении числа процессоров в t раз время вычисления уменьшается в t раз. Эффективность равна нулю, когда с ростом числа процессоров время вычисления не изменяется. Как показывают эксперименты, результаты которых приведены в табл. 1, эффективность распараллеливания вычислений находится в пределах от 44 до 98% для матриц данного размера и при изменении числа процессоров от 1 до 256.

ЛИТЕРАТУРА

1. Компьютерная алгебра: Символьные и алгебраические вычисления / пер. с англ. под ред. Б. Бухбергера, Дж. Коллинза, Р. Лооса. М.: Мир, 1986.

2. *Becker T., Weispfenning V.* Grobner Bases. A Computational Approach to Commutative Algebra. New York: Springer, 1993. V. 141.

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

4. Алгоритмы компьютерной алгебры. Ч. 1: Учебное пособие / Г.И. Малашонок, О.Н. Переславцева, О.А. Сажнева, М.В. Старов; Федеральное агенство по образованию, Тамб. гос. ун-т им. Г.Р. Державина. Тамбов: Издательский дом ТГУ им. Г.Р. Державина, 2008.

5. *Старов М.В.* О реализации алгоритма Фужера вычисления базисов Гребнера // International conference Polynomial Computer Algebra. St.Petersburg, 2008. С. 66.

6. *Малашонок Г.И., Старов М.В.* Об экспериментах с алгоритмами Фужера F4 // Mathematical Modeling and Computation Physics (MMCP'2009): Book of Abstracts of the International Conference. Dubna: JINR, 2009. P. 135.

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

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

9. *Gebauer R., Moller H.M.* On an Installation of Buchberger's Algorithm // Journal of Symbolic Computation 1988. V. 6. P. 275–286.

10. *Malaschonok G.I.* On computation of kernel of operator acting in a module // Tambov University Reports. Series Natural and Technical Sciences. Tambov, 2008. V. 13. Issue. 1. P. 129-131.

11. *Кнут Д.* Искусство программирования для ЭВМ. Т. 2. М.: Мир, 1977.

БЛАГОДАРНОСТИ: Работа выполнена при финансовой поддержке АВЦП «Развитие научного потенциала высшей школы (2009-2010 годы)» (проект № 2.1.1/1853); темплана 1.12.09.

Поступила в редакцию 28 августа 2010 г.

Malaschonok G. I., Starov M. V., Borisov I. A. On the parallel computation of Groebner bases.

There are considered one parallel version of Groebner bases calculations, based on matrix algorithm of J. Faugere (F4). We suggest to use the parallel computations only for getting the echelon form of matrices. We describe the modular parallel version of the algorithm for matrix echelon form computation. We discuss the results of computing experiments for calculation of Groebner bases with parallel algorithm on the cluster MVS100k of Joint Super Computer Center RAS.

Key words: calculation of Groebner bases; parallel algorithm; modular method, method of homomorphic images; cluster computations.