зультаты записываются в соответствующие поля результатов, флаг этой дочерней вершины принимает значение (− 2), меняются счетчики флагов, а процессоры распределяются процедурой *distribFreeProc*.

В этом же треке потока $Stream_{out}$ происходит обработка всех запросов на прием сообщений, учитывается изменение всех счетчиков флагов и проверяется завершенность вычислений всего текущего пучка.

Если пучок вычислен, а за ним следует Легкий Лист, то этот лист вычисляется процедурой *lightLeaf* в этом же потоке $Stream_{out}$. После чего начинается процесс вычисления следующего пучка процедурой *computNewSheaf* и используется весь список свободных процессоров.

Если же вычисление пучка не окончено, то распределяются свободные процессоры процедурой *distribFreeProc*. После этого трек потока $Stream_{out}$ завершается.

Работа выполнена при частичной поддержке грантов РФФИ (проект 08-07-97507) и программы "Развитие потенциала высшей школы"(проект 2.1.1/1853).

Список литературы

1. *Коновалов Н.А., Крюков В.А., Михайлов С.Н., Погребцов Л.А.* Fortran-DVM - язык разработки мобильных параллельных программ. // Программирование. 1995. № 1. 17. С. 49-54.

2. *Берзигияров П.К.* Программирование на типовых алгоритмических структурах с массивным параллелизмом. Вычислительные методы и программирование. 2001. Т.2. С. 1-16.

3. *Berzigyarov P.K.* Static Pipelines for Divide-and-Conquer Functions: transformation and Analysis. Preprint IVTAN, Moscow, 1995. N 8-391.

4. *Campbell D.K.G.* Towards the Classification of Algorithmic Skeletons. Technical Report YCS 276. Department of Computer Science, University of York. York, 1996.

5. *Cole M.I.* Algorithmic Skeletons: Structured Management of Parallel Computation. Massachusetts, Cambridge. Boston: MIT Press, 1989.

6. *Gamma E., Helm R., Johnson R., Vlissides J.* Design Patterns: Elements of Reusable Object-Oriented Software. Reading-Amsterdam-Tokyo: Addison-Wesley, 1995.

7. *Singh A., Schaeffer J., Szafron D.* Views on template-based parallel programming // Proc. of CASCON'96. Toronto, 1996. С. 1-12.

8. *Siu S.* Openness and Extensibility in Design-Pattern-Based Parallel Programming Systems. Master of Applied Science Thesis. University of Waterloo. Ontario. Canada. Waterloo, 1996.

9. *Siu S., De Simone M., Goswami D., Singh A.* Design patterns for parallel programming // Parallel and Distributed Processing Techniques and Applications. California. Pasadena, 1996. С. 230-240.

10. *MacDonald S.* Design patterns in enterprise // Proc. of CASCON'96. Toronto, 1996. С. 1-10.

11. *MacDonald S., Szafron D., Schaeffer J., Bromling S.* Generating parallel program frameworks from parallel design patterns // Proc. of EuroPar'2000. Berlin: Springer-Verlag, 2000. С. 95-104.

12. *Abramov S., Adamovitch A., Kovalenko M.* T-system: programming environment providing automatic dynamic parallelizing on IP-network of Unix-computers. Report on 4-th International Russian-Indian seminar and exibition. Sept. 15-25. Moscow, 1997.

13. *Малашонок Г.И., Валеев Ю.Д.* Организация параллельных вычислений в рекурсивных символьно-численных алгоритмах. Труды конференции ПаВТ'2008 (Санкт-Петербург). Челябинск: Изд-во ЮУрГУ, 2008. С. 153-165.

Поступила в редакцию 17 ноября 2008 г.

ВЫЧИСЛЕНИЕ ХАРАКТЕРИСТИЧЕСКОГО ПОЛИНОМА ДЛЯ ПОЛИНОМИАЛЬНЫХ МАТРИЦ

© **О.Н. Переславцева**

Ключевые слова: вычисление характеристического полинома, алгоритм Сейфуллина, алгоритм Данилевского, вычислительная сложность алгоритма.

Аннотация

Рассматриваются алгоритмы вычисления характеристических полиномов полиномиальных матриц с использованием Китайской теоремы об остатках (КТО). Приводится верхняя оценка степени и наибольшего коэффициента характеристического полинома полиномиальной матрицы от одной переменной. Обсуждаются результаты численных экспериментов для прямого алгоритма Сейфуллина и алгоритма Данилевского с применением КТО вычисления характеристического полинома.

1 Введение

Работа посвящена вычислению характеристических полиномов полиномиальных матриц от одной переменной. Теоретический анализ и сравнение алгоритмов вычисления характеристических полиномов числовых матриц были сделаны в работах [1, 2].

Среди прямых алгоритмов асимптотически лучшим будет алгоритм Сейфуллина [3], так как оценка числа бит-умножений данного алгоритма наименьшая. Это говорит о том, что в ходе алгоритма промежуточные числовые коэффициенты растут медленнее всех. Это будет верно и для полиномиальных матриц. Тогда алгоритм Сейфуллина будет лучшим среди прямых алгоритмов и для полиномиальных матриц.

Наименьшую сложность, выраженную в количестве операций над элементами матриц, имеет алгоритм Данилевского [4]. Поэтому, если применить к этому алгоритму китайскую теорему об остатках (КТО), он будет иметь асимптотически лучшее время по сравнению с другими алгоритмами с применением КТО.

Для сравнения алгоритма Сейфуллина и алгоритма Данилевского с применением КТО были написаны программы для вычисления характеристических полиномов для полиномиальных матриц от одной переменной.

2 Алгоритм Данилевского

Алгоритм Данилевского был применен совместно с китайской теоремы об остатках (КТО).

Пусть $A = (a_{ij}(x)),\ 1 \le i \le n,\ 1 \le j \le n$, данная матрица. Выберем k простых чисел $p_1, \ldots, p_k$ и перейдем к гомоморфным образам $\mathbb{Z}[x] \to \mathbb{Z}[x]/p_i = \mathbb{Z}_{p_i}[x]$. Будем вычислять k характеристических полиномов для полиномиальных матриц $M_i \in \mathbb{Z}_{p_i}^{n\times n}[x],\ 1 \le i \le k$.

Для вычисления характеристического полинома для матрицы $M_i,\ 1 \le i \le k$, возьмем m полиномов $x, x-1, \ldots, x-(m+1)$ и перейдем к гомоморфным образам $\mathbb{Z}[x]/p_i \to \mathbb{Z}_{p_i}$. Получим $k \cdot m$ матриц $M_{ij} \in \mathbb{Z}_{p_i}^{n\times n},\ 1 \le i \le k,\ 1 \le j \le m$.

Для каждой матрицы M_{ij} вычислим характеристический полином над полем $\mathbb{Z}_{p_i}$, применив алгоритм Данилевского. Получим $k \cdot m$ полиномов $f_{ij}(y),\ 1 \le i \le k,\ 1 \le j \le m$.

Каждый массив полиномов $\{f_{i1}(y), f_{i2}(y), \ldots, f_{im}(y)\}$ восстановим в полином $F_i(x,y)$ по полиномиальным модулям $x, x-1, \ldots, x-(m+1)$, $(1 \le i \le k)$. Т.е. $F_i(x,y)$ есть характеристический полином матрицы $M_i \in \mathbb{Z}_{p_i}^{n\times n}[x]$.

Затем восстановим коэффициенты полиномов $F_1(x,y), \ldots, F_k(x,y)$ по числовым модулям $p_1, \ldots, p_k$ в полином $F(x,y)$, который является характеристическим полиномом данной матрицы A.

3 Оценка коэффициентов характеристического полинома полиномиальной матрицы от одной переменной

Для программной реализации модулярного алгоритма вычисления характеристического полинома матрицы необходимо знать k и m.

Для полинома f введем норму $||f||$ – наибольший по модулю коэффициент этого полинома. Для полиномов f и g верно, что $||f+g|| \le ||f|| + ||g||$ и $||fg|| \le (min\{\deg f, \deg g\}+1)||f|| \cdot ||g||$.

Обозначим через $A = (a_{ij}(x)),\ 1 \le i \le n,\ 1 \le j \le n$, данную матрицу, $s = max\{\deg a_{ij}(x)\}$, $||A|| = max\{||a_{ij}||\} = a$ для $1 \le i \le n,\ 1 \le j \le n$.

Пусть характеристический полином имеет вид $F(x,y) = y^n + f_1(x)y^{n-1} + \cdots + f_n(x)$. Тогда $m = \max\{\deg f_1(x), \ldots, \deg f_n(x)\} + 1$.

Оценим m и $||F(x,y)||$.

Коэффициент $f_i(x)$ есть сумма всех определителей порядка $n-i$, опирающихся на главную диагональ. Поэтому $m \le n \cdot s + 1$.

Для получения оценки $||F(x,y)||$ воспользуемся алгоритмом Леверье-Фаддеева [5]:

$B_0 = E$;

$i = 1, \ldots, n$:

$\{A_i = AB_{i-1}$;

$f_i = (1/i)TrA_i$;

$B_i = A_i - f_iE\}$.

Для матрицы B_i будем рассматривать две нормы: $||B_i||_d$ – наибольший по модулю коэффициент полиномов матрицы B_i, стоящих на главной диагонали, $||B_i||_{nd}$ – наибольший по модулю коэффициент полиномов матрицы B_i, не стоящих на главной диагонали.

При $i = 1$: $||A_1|| \le a$, $||f_1|| \le na$, $||B_1||_{nd} \le a$, $||B_1||_d \le (n+1)a$.

Для $i > 1$: $\min\{\deg B_{i-1}, s\} = s$. Тогда

$||f_i|| \le (n/i)||A_i||$;

$||A_i|| \le (n-1)(s+1)a||B_{i-1}||_{nd} + (s+1)a||B_{i-1}||_d$;

$||B_{i-1}||_{nd} = ||A_{i-1}||$ и $||B_{i-1}||_d = ||A_{i-1}|| + ||f_{i-1}||$.

Следовательно, $||A_i|| \le n^{i-1}(s+1)^{i-1}a^i i$ и $||f_i|| \le n^i(s+1)^{i-1}a^i$ для $i > 1$.

Таким образом, получили верхнюю оценку для коэффициентов характеристического полинома полиномиальной матрицы от одной переменной:

$$\log_2 ||F(x,y)|| \le n(\log_2 n + \log_2 a + \log_2(s+1)) - \log_2(s+1).$$

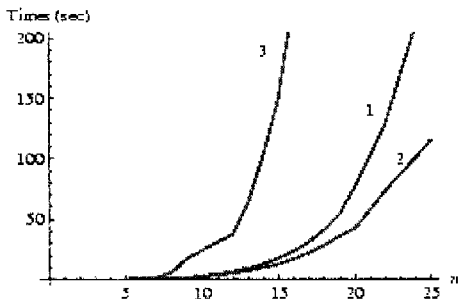
4 Вычислительные эксперименты

В экспериментах использовались плотные полиномиальные матрицы от одной переменной. Рассматривались плотные полиномы степени 15, коэффициенты полиномов длиной 20 бит выбирались случайным образом. Размерность матрицы менялась в пределах от 5 до 25.

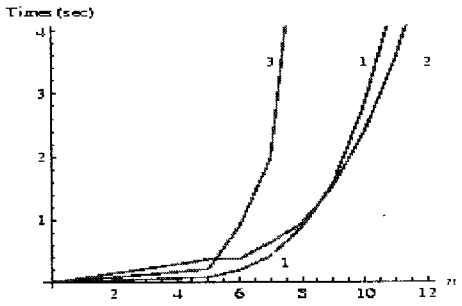
Рассматривались следующие алгоритмы:

1. прямой алгоритм Сейфуллина,
2. алгоритм Данилевского с применением КТО,
3. алгоритм, который реализован в системе Mathematica 5.1.

Результаты экспериментов приведены на рис. 1 и рис. 2.



$5 < n < 25$



$5 < n < 12$

Рис. 1, 2. Время вычисления характеристического полинома с использованием прямого алгоритма Сейфуллина (1), алгоритма Данилевского с применением КТО (2), алгоритма, реализованного в системе Mathematica 5.1 (3)

Алгоритм Данилевского с применением КТО выигрывает у остальных рассматриваемых алгоритмов. Например, при $n = 20$ он вычисляет характеристический полином быстрее в 1,8 раза, чем алгоритм Сейфуллина, и быстрее в 23 раза, чем алгоритм, который реализован в системе Mathematica 5.1.

В дальнейшем предполагается распараллелить алгоритм Данилевского с применением КТО для полиномиальных матриц и провести эксперименты на кластере ТГУ и кластере МСЦ.

Работа выполнена при частичной поддержке грантов РФФИ (проект 08-07-97507) и программы "Развитие потенциала высшей школы"(проект 2.1.1/1853).

Список литературы

1. *Икрамов Х.Д.* О конечных спектральных процедурах в линейной алгебре // Программирование. 1994. N 1. С. 56-69.

2. *Переславцева О.Н.* О вычислении коэффициентов характеристического полинома // Вычислительные методы и программирование. Издательство Московского университета, 2008. Т.9. N.2. С. 180-185.

3. *Сейфуллин Т.Р.* Вычисление определителя, присоединённой матрицы и характеристического полинома без деления // Кибернетика и системный анализ. 2002. N.5. С. 18-42.

4. *Данилевский А.М.* О численном решении векового уравнения // Матем. сб. 1937. Т.2(44). N1. С. 169-172.

5. *Фаддеев Д.К., Фаддеева В.Н.* Вычислительные методы линейной алгебры. М., Л.: Гос. изд. физ. мат. литературы, 1963.

Поступила в редакцию 17 ноября 2008 г.

РЕШЕНИЕ СИСТЕМ НЕОДНОРОДНЫХ И ОДНОРОДНЫХ ЛИНЕЙНЫХ УРАВНЕНИЙ НАД КОЛЬЦОМ ПОЛИНОМОВ

© **О.А. Сажнева**

Ключевые слова: система линейных неоднородных уравнений, p-адический метод, канонический базис целых решений однородной системы линейных уравнений над кольцом полиномов.

Аннотация

Рассматривается p-адический метод решения системы линейных неоднородных уравнений над кольцом полиномов. Приводится пример решения системы линейных неоднородных уравнений над кольцом полиномов. Описывается метод нахождения канонического базиса целых решений однородной системы линейных уравнений над кольцом полиномов. Приводится пример нахождения канонического базиса.

1 p-адический метод решения системы линейных неоднородных уравнений над кольцом полиномов

Существуют разные методы решения систем линейных неоднородных уравнений в области главных идеалов R. Сравнение по оценкам сложности этих методов показывает преимущество метода, предложенного в работе [1]. В этом методе выделяются два этапа. На первом этапе вычисляется базисное множество решений системы в поле частных области. На втором строится базис целых решений.

Для построения решения системы в поле частных области сначала находится решение в факторкольце R/p^k, где p - подходящий простой элемент кольца R, а степень k выбирается так, чтобы степень полинома p^k превосходила степень произведения числителя и знаменателя в каждой рациональной компоненте вектора решения.

Решение системы в кольце R/p^k осуществляется с помощью p-адического подъема [2]. Сначала найдем решение в R/p. Пусть $x_p = A_p^{-1} \cdot c_p$, где индекс p означает отображение $R \to R/p$. Чтобы найти решение в R/p^2, рассмотрим систему $A(xp + x_p) = c$. Так как $c' = c - Ax_p$ делится нацело на p, то достаточно решить систему $Ax = c'/p$ в R/p. Этот процесс продолжать до p^k.