ПРОГРАММИРОВАНИЕ, 2003, N№ 2, с. 8–22

КОМПЬЮТЕРНАЯ АЛГЕБРА

УДК 681.32

О РЕШЕНИИ СИСТЕМ ЛИНЕЙНЫХ УРАВНЕНИЙ p-АДИЧЕСКИМ МЕТОДОМ*

© **2003 г. Г. И. Малашонок**

Тамбовский государственный университет им. Г.Р. Державина

392622 Тамбов, ул. Интернациональная, 33

E-mail: malashonok@math-iu.tstu.ru

Поступила в редакцию 03.06.2002 г.

Предлагается новый алгоритм решения системы линейных уравнений $Ax = b$ в евклидовой области. В случае кольца целых чисел $\mathbb{Z}$ и при использовании стандартных алгоритмов умножения целых чисел и матриц оценка сложности этого алгоритма для решения системы с матрицей $A \in \mathbb{Z}^{n\times m}$, $m > n$, ранга n составляет $O^{\sim}(n^3 m \log^2 \|A\|)$, где $\|A\| = \max\limits_{i,j} |A_{i,j}|$. Лучший предыдущий алгоритм Г. Лабана и А. Шториоханна [1] при таких же условиях имеет оценку сложности $O^{\sim}(n^4 m \log^2 \|A\|)$. Однако теоретическая оценка в случае применения быстрых алгоритмов умножения чисел и матриц для последнего алгоритма составляет $O^{\sim}(n^{\beta} m \log \|A\|)$, что лучше, чем аналогичная оценка для нового алгоритма – $O^{\sim}(n^3 m \log \|A\|)$.

1. ВВЕДЕНИЕ

В этой статье рассматриваются две основные евклидовы области – кольцо целых чисел и кольцо полиномов над полем, и две задачи – решение систем линейных уравнений в области и решение систем линейных уравнений в поле частных области.

Рассмотрим задачу решения системы линейных уравнений с квадратной основной матрицей коэффициентов, определитель которой не равен нулю. Такую систему называют **определенной системой**, она имеет единственное решение в поле частных области.

Пусть все коэффициенты системы целочисленные, порядок основной матрицы коэффициентов равен n, и пусть для решения такой системы применяется алгоритм со сложностью $\theta(n^3)$ операций над коэффициентами[1].

Если использовать стандартную арифметику целых чисел, то сложность операции умножения будет пропорциональна произведению логарифмов сомножителей. Операндами на k-ом шаге, $k = 1, \ldots, n-1$, являются миноры k-го порядка, следовательно, их логарифмы растут как $O^{\sim}(k)$. В итоге сложность такого алгоритма будет расти как $O^{\sim}(n^5)$.

Модулярный метод, основанный на китайской теореме об остатках, позволяет избежать роста коэффициентов. Вычисления производятся в кольце вычетов по простому модулю, и роста коэффициентов не происходит. При этом для восстановления решения потребуется $O^{\sim}(n)$ модулей, поэтому нужно $O^{\sim}(n)$ раз решать систему. Следовательно, сложность такого алгоритма растет как $O^{\sim}(n^4)$.

Если же применять p-адический метод с линейным подъемом, то удается избежать роста не только коэффициентов, но и числа модулей. Сначала ищутся обратная матрица и вектор решения по модулю p, это потребует $\theta(n^3)$ операций по модулю p. Затем, умножая найденную обратную матрицу на пересчитанный столбец свободных членов, поднимается вектор решения до $p^{O^{\sim}(n)}$. При этом выполняется $O^{\sim}(n)$ шагов

*Работа частично поддержана грантом Минобразования России E02-2.0-98.

[1] Если f и g – функции со значениями в $\mathbb{R}$, то обозначаем:
$g = O(f)$, если $|g/f|$ ограничена на $+\infty$,
$g = O^{\sim}(f)$, если $g = O(f \log^c f)$ для некоторой постоянной c,
$g = \theta(f)$, если $|g/f|$ и $|f/g|$ ограничены на $+\infty$,
$g = o(f)$, если $\lim(g/f) = 0$.

подъема, на каждом шаге нужно $\theta(n^2)$ операций по модулю p для умножения матрицы на вектор. Затем восстанавливаются n рациональных чисел, составляющих вектор решения, для вычисления каждого из них требуется $\theta(n^2)$ операций. Таким образом, на каждом из этих этапов нужно $O^{\sim}(n^3)$ операций. Поэтому в целом сложность такого алгоритма $O^{\sim}(n^3)$.

Пусть теперь для решения определенной системы применяется алгоритм со сложностью $O(n^\beta)$ операций над коэффициентами ($\beta \approx \approx 2.376$), и применяется алгоритм FFT для умножения чисел [2]. Для умножения n-битных чисел требуется $O^{\sim}(n)$ бит операций, столько же операций требуется для вычисления НОД и для восстановления числа по взаимно простым остаткам. Отметим, что эти оценки сегодня можно рассматривать только как теоретические, в отличие от приведенных выше, которые легко реализуются практически.

Первый и второй алгоритмы, то есть алгоритмы со стандартной и модулярной арифметикой, в этом случае требуют $O^{\sim}(n^{\beta+1})$ бит-операций, а p-адический алгоритм по-прежнему требует $O^{\sim}(n^3)$ бит-операций.

Следовательно, для определенных систем асимптотически лучшим методом, как в теоретическом, так и в практическом смысле, является p-адический метод. Такой метод был предложен Дж. Диксоном (J. Dixon, 1982 [3]). Отметим, что для *небольших систем* лучшими являются алгоритмы со стандартной арифметикой, для систем *промежуточного* размера – модулярные алгоритмы, и только для *достаточно больших систем* на первое место выходит p-адический метод. Это связано, в первую очередь, с тем, что восстановление частного – это более сложная процедура, чем восстановление отдельно числителя и знаменателя.

В настоящей работе рассматриваются алгоритмы решения **неопределенных систем** линейных уравнений, вычисляющие общее решение таких систем.

Традиционный метод для решения линейных систем диофантовых уравнений основан на вычислении формы Смита матрицы коэффициентов системы [4–7].

Другой подход, развиваемый в последние годы, состоит в вычислении общего решения системы как суммы частного решения системы и общего решения соответствующей однородной системы. Лучшие результаты в этом направлении получены в работах [1, 8, 9]. Пусть $xA = 0$ – однородная диофантова система линейных уравнений, матрица $A \in \mathbb{Z}^{m\times n}$ имеет ранг n, $m > n$, $x \in \mathbb{Z}^{1\times m}$, U – унимодулярная матрица над $\mathbb{Z}$ такая, что $UA = H$, и H имеет верхнюю треугольную форму, тогда нетрудно видеть, что последние $m - n$ строк матрицы U составляют базис множества всех решений однородной системы $xA = 0$.

Такой алгоритм предлагается в работе [8]. Для вычисления общего решения однородной диофантовой системы $xA = 0$ предлагается алгоритм вычисления верхней треугольной формы для матрицы A. Он имеет сложность $O^{\sim}(n^2 m \boldsymbol{M}(n \log \|A\|))$. В работе [1] он улучшен до $O^{\sim}(n^{\beta-1} m \boldsymbol{M}(n \log \|A\|))$, при этом вычисляется эрмитова верхняя треугольная форма. В работе [9] предлагается вероятностный метод вычисления частного решения диофантовой системы линейных уравнений, имеющий ожидаемую сложность $O^{\sim}(n^3 \log^3 \|A\|)$ при использовании стандартных алгоритмов для умножения. Следовательно, основную часть в общей сложности такого алгоритма решения системы диофантовых уравнений составляет сложность задачи вычисления общего решения однородной системы. Здесь и далее $\|A\| = \max\limits_{i,j}(|A_{i,j}|)$, β – показатель степени в оценке сложности матричного умножения, предполагается, что две матрицы порядка n можно перемножить за $\theta(n^\beta)$ операций, $\theta(\boldsymbol{M}(t))$ – сложность умножения двух t-битных чисел. Для стандартных алгоритмов $\beta = 3$ и $\boldsymbol{M}(t) = t^2$. Лучшие теоретические результаты, известные сегодня, дают $\beta \approx 2.376$ и $\boldsymbol{M}(t) = t \log t \log\log t$ [2, 10].

Современная вычислительная техника не позволяет реализовать лучшие теоретические результаты. В связи с этим принято говорить отдельно о лучших теоретических и о лучших практических алгоритмах. Для стандартных алгоритмов умножения оценкой сложности алгоритма [1] для решения диофантовой системы линейных уравнений является $O^{\sim}(n^4 m \log^2 \|A\|)$, а для лучших алгоритмов умножения – $O^{\sim}(n^\beta m \log \|A\|)$.

Мы рассмотрим другой подход к решению неопределенных систем линейных уравнений – это детерминистский p-адический и вероятностный p-адический методы, предложенные в [11] и развитые в данной статье.

Основная идея состоит в вычислении базисного множества точек на плоскости всех решений системы в пространстве $\mathbb{Q}^n$ с помощью p-адического подъема. А затем для получения решений в области $\mathbb{Z}$ вычисляются целые точки на этой плоскости. При этом в детерминистском методе задача сводится к решению системы в факторкольце, а в вероятностном методе вычисляется одно целое решение, если идеал, порожденный знаменателями всех рациональных решений, является единичным, а затем строятся все остальные базисные целые решения.

Оценка сложности детерминистского алгоритма (см. предложение 3) для стандартных алгоритмов умножения матриц и чисел составляет $O^{\sim}(n^3 m \log^2 \|A\|)$, а для теоретически лучших алгоритмов умножения матриц и чисел – $O^{\sim}(n^3 m \log \|A\|)$, где матрица $A \in \mathbb{Z}^{n\times m}$ имеет ранг n. Сравнение этих оценок с оценками сложности алгоритма из [1] показывает, что для быстрых алгоритмов умножения оценки сложности нового алгоритма хуже в $n^{3-\beta}$ раз, а для стандартных алгоритмов умножения оценки для нового алгоритма лучше в n раз. Это позволяет считать алгоритм из [1] теоретически лучшим, а новый алгоритм – практически лучшим.

Еще одно преимущество нового алгоритма состоит в том, что коэффициенты в каждом вычисляемом векторе решения не превосходят по модулю величины $v = (\sqrt{n}\|A\|)^n$, оценки по Адамару наибольшего минора порядка n, в то время, как в алгоритме из [1] оценка для модуля коэффициентов составляет nv^2, а в алгоритме из [8] она еще больше.

Новые результаты, полученные в данной работе, сводятся к следующему.

- В развитие общего метода, изложенного в [11], приведены детерминистский p-адический и вероятностный p-адический алгоритмы (см. рис. 2–5).

- Получены уточненные оценки сложности этих алгоритмов для разных размеров системы и для разных вычислительных моделей.

- Оценка сложности в случае применения стандартных алгоритмов умножения понижена в n раз по сравнению с оценкой, полученной в [11]. Это достигается в результате применения модулярного алгоритма для вычисления произведения матриц в факторкольце.

Содержание остальных разделов следующее. Во втором разделе приводятся две вычислительные модели, которые используются в дальнейшем для получения оценок сложности алгоритмов. В третьем разделе излагаются p-адические алгоритмы решения систем линейных уравнений в поле частных (алгоритмы с рис. 2, 3). В четвертом разделе излагается вероятностный метод решения системы в коммутативной области (алгоритм с рис. 4). В пятом разделе излагается детерминистский метод решения системы в евклидовой области (алгоритм с рис. 5) и приводятся оценки сложности.

2. ВЫЧИСЛИТЕЛЬНАЯ МОДЕЛЬ

Пусть $F[x]$ – кольцо полиномов над полем F. Будем называть *нормой* элемента p в $F[x]\backslash\{0\}$ степень полинома и обозначать ее $\mathrm{Nr}(p)$, а нормой элемента в кольце целых чисел $\mathbb{Z}\backslash\{0\}$ – логарифм абсолютного значения этого элемента:

$$\mathrm{Nr}(p) = \begin{cases} \deg p & \text{для } F[x], \\ \log_w |p| & \text{для } \mathbb{Z}. \end{cases} \tag{1}$$

Логарифм удобно брать по основанию w, равному величине машинного слова, сегодня это обычно 2^{32} или 2^{64}. Если каждый коэффициент полинома хранится в одном машинном слове, то число машинных слов, которое необходимо для записи одного элемента p, в обоих случаях будет $\lfloor \mathrm{Nr}(p) \rfloor + 1$, и имеют место следующие свойства: $0 \le \mathrm{Nr}(p)$, $\mathrm{Nr}(pq) = \mathrm{Nr}(p) + \mathrm{Nr}(q)$. Определим также норму ненулевой матрицы $A = (a_{ij})$ как наибольшую из норм ее элементов: $\|A\| = \max\limits_{i,j} \mathrm{Nr}(a_{ij})$, и введем функцию ρ_A:

$$\rho_A = \begin{cases} \|A\| & \text{для } F[x], \\ \|A\| + \dfrac{1}{2}\log_w n & \text{для } \mathbb{Z}. \end{cases} \tag{2}$$

Здесь n – наименьшая размерность матрицы A. Функция ρ_A позволяет записать оценку для нормы определителя матрицы A порядка n:

$$\mathrm{Nr}(\det A) \le n\rho_A. \tag{3}$$

Для кольца $F[x]$ это неравенство очевидно, а для доказательства его в кольце $R = \mathbb{Z}$ нужно воспользоваться неравенством Адамара. Отметим также, что имеет место верхняя оценка для определителя любой подматрицы A_0 порядка k матрицы A: $\mathrm{Nr}(\det A_0) \le k\rho_A$. Будем оценивать алгоритмы в двух вычислительных моделях.

Лучшая современная модель (B-модель). *Лучший сегодня алгоритм [12] для мультипликативных операций над целыми N-разрядными числами имеет следующую оценку сложности:*

$$\Lambda_B = O(N \log N \log\log N), \tag{4}$$

выраженную в мультипликативных операциях над машинными словами, а лучший алгоритм [13] вычисления НОД для N-разрядных чисел имеет оценку сложности

$$\Phi_B = O(N \log^2 N \log\log N). \tag{5}$$

И такие же оценки имеют место для числа мультипликативных операций в поле F для полиномов степени N из $F[x]$ (см. подробное обсуждение в [2] и [10]).

Чтобы иметь оценку сложности для промежуточных алгоритмов, включая классический алгоритм, будем пользоваться следующей моделью.

Общая вычислительная модель (C-модель). *Будем полагать, что имеются алгоритмы для умножения и деления чисел и многочленов, которые не превосходят по норме элемента a, со сложностью*

$$\Lambda_C \le \eta \mathrm{Nr}^{\delta}(a) + o(\mathrm{Nr}^{\delta}(a)), \tag{6}$$

выраженной в числе мультипликативных операций над машинными словами для $\mathbb{Z}$ и в числе мультипликативных операций в поле F для $F[x]$. Здесь η и δ – постоянные, и $1 < \delta \le 2$. Будем полагать, что имеется алгоритм Евклида для вычисления НОД двух элементов a и b $(\mathrm{Nr}(b) \le \mathrm{Nr}(a))$ со сложностью

$$\Phi_C \le \mathrm{Nr}^2(a) \log_2 w + O(\mathrm{Nr}(a)), \tag{7}$$

здесь w – это величина машинного слова для $\mathbb{Z}$, и $w = 2$ для $F[x]$.

Оценка (6) для $\eta = 1$ и $\delta = 2$ дает оценку стандартного алгоритма умножения, для $\eta = 1$ и $\delta = \log_2 3$ – оценку алгоритма умножения Карацубы, когда a – степень числа 2.

Для $F[x]$ оценка (7) очевидна, а для $\mathbb{Z}$ она опирается на известную оценку [14]:
максимальное число делений в алгоритме Евклида для чисел a и b $(|b| \le |a|)$ не превосходит

$$\lfloor 2 \log_2 a \rfloor + 1. \tag{8}$$

Известно также (см. [15]), что при фиксированном b и произвольном a максимальное число делений в алгоритме Евклида превосходит

$$(\log_2(3+\sqrt{5}) - 1)^{-1} \log_2 b \quad (\approx 0.72 \log_2 b).$$

Численные эксперименты показывают, что среднее число делений в алгоритме Евклида для целых чисел равно $\dfrac{2}{1+\sqrt{5}} \log_2 b$, а частные обычно находятся в пределах одного машинного слова. Поэтому для вычисления частного достаточно, как правило, одной операции деления, а для нахождения остатка нужно столько операций умножения, сколько слов в записи числа. Подробное обсуждение можно найти, например, в работах [16, 17].

Оценки сложности для C-модели обозначаются нижним индексом C. Они даются для кольца $\mathbb{Z}$ в числе мультипликативных операций над *машинными словами*, а для кольца $F[x]$ – в числе мультипликативных операций в поле F. Оценки, которые будут получены для B-модели, обозначаются нижним индексом B.

3. РЕШЕНИЕ СИСТЕМЫ p-АДИЧЕСКИМ МЕТОДОМ В ПОЛЕ ЧАСТНЫХ

Общая схема решения определенной системы p-адическим методом была предложена Дж. Диксоном (J. Dixon, 1982 [3]). Она заключается в следующем.

Выбирается подходящий простой элемент p кольца R. Его удобно выбирать размером в одно машинное слово, так, чтобы арифметические операции по модулю p можно было выполнять не выходя за пределы максимального целого числа, определяемого вычислительной системой. Необходимо, чтобы элемент p не делил определитель матрицы коэффициентов. Случайный выбор числа p может оказаться неудачным. Так как числа, кратные простому числу p, в кольце $\mathbb{Z}$ встречаются с вероятностью $1/p$, то выбор числа p в кольце $\mathbb{Z}$ может оказаться

Algorithm RationalReconstruction
Input: Модуль μ и остаток $u \in \mathbb{Z}_\mu$.

Output: Пара целых чисел (a, b), $b > 0$, такая, что $a \equiv bu (\mathrm{mod} \mu)$, $|a|, |b| < \sqrt{\mu/2}$.

$(a_1, a_2) := (\mu, u)$; $(v_1, v_2) := (0, 1)$;
while $TRUE$ **do**
  if $v_2 \geq \sqrt{\mu/2}$ **then return NIL** ;
  if $a_2 < \sqrt{\mu/2}$ **then return** $(\mathrm{sign}(v_2)a_2, |v_2|)$;
  $q := \lfloor a_1/a_2 \rfloor$; $(a_1, v_1) := (a_1, v_1) - q(a_2, v_2)$;
  ПЕРЕСТАНОВКА$((a_1, v_1) \leftrightarrow (a_2, v_2))$;
endwhile.

Рис. 1. Алгоритм П.С. Ванга "Реконструкция рационального числа".

неудачным с вероятностью $1/p$. Если проверка решения покажет, что решение неверное, то выбирается следующий простой элемент p. Решение системы в кольце вычетов по модулю p можно искать любым быстрым методом решения систем в поле со сложностью матричного умножения. Можно воспользоваться, например, рекурсивным методом [18].

Из неравенства Адамара следует оценка для определителя матрицы A порядка n: $|\det A| \leq (\sqrt{n}\|A\|)^n$. Она позволяет найти верхние оценка для числителей и знаменателей решения системы, по ним находится оценка для величины p^k – границы подъема. Граница подъема должна быть не меньше удвоенного произведения наибольшего из числителей на наибольший из знаменателей в векторе решения, чтобы можно было восстановить любую дробь с учетом знака ($p^k \geq 2(\sqrt{n}\|A\|)^{2n}$). Затем решение поднимается по модулю p до p^k, и реконструируется рациональное решение. Подробное обсуждение p-адических методов можно найти в [16].

Приведенные в настоящей статье алгоритмы опираются на алгоритм Дж. Диксона для решения одной определенной системы с помощью линейного p-адического подъема, алгоритм реконструкции рациональной дроби П.С. Ванга (P.S. Wang, 1981 [19]) и общий метод сведения решения произвольной системы к решению определенных систем, теоремы 1 и 2.

3.1. *Реконструкция дроби*

Алгоритм реконструкции рационального числа a/b по данному модулю μ (b взаимно просто с μ) и остатку u – это алгоритм вычисления таких целых чисел a и b, не превосходящих по модулю $\sqrt{\mu/2}$, что $a \equiv bu \bmod \mu$.

Такой алгоритм был предложен П.С. Вангом [19], а затем уточнен Г.Е. Коллинзом и М.Дж. Энкарнасионом (G.E. Collins, M.J. Encarnacion, 1995 [20]). Для полноты картины на рис. 1 приведен алгоритм П.С. Ванга [19].

Алгоритм П.С. Ванга для чисел μ и u повторяет алгоритм Евклида и вычисляет последовательность остатков (a_1, a_2). Одновременно вычисляется дополнительная последовательность (v_1, v_2), начинающаяся с пары $(1, 0)$ и вычисляющая один из двух коэффициентов в расширенном алгоритме Евклида. Вычисления прекращаются, когда очередной остаток будет меньше по абсолютной величине, чем $\sqrt{\mu/2}$. При этом последний остаток – это число a, последний член дополнительной последовательности – число b. Сложность алгоритма П.С. Ванга такая же, как и сложность алгоритма Евклида.

3.2. *P-адический метод решения системы в поле частных*

Пусть R – евклидова область,

$$Ax = c \tag{9}$$

– система линейных уравнений над R с матрицей коэффициентов $A \in R^{n \times m}$ ранга r. Пусть S и T – матрицы перестановок. Матрицы SAT и Sc запишем в блочной форме:

$$SAT = \begin{pmatrix} A_0 & A_1 \\ A_2 & A_3 \end{pmatrix}, \; Sc = \begin{pmatrix} c_0 \\ c_1 \end{pmatrix}, \; c_0 \in R^r, \tag{10}$$

Algorithm RationalBasisH[A]
Input: $A \in R^{n\times m}$
Output: Решения системы $Ax = 0$.
$p := FirstPrime$;
repeat
  Инициализация алгоритма:
  $\{\tilde{A}_0^{-1}, S, T, r\} := DiagonalFormH[mod_p\ A]$;
  if $m = r$ **then return** $\{\emptyset\}$;
  $\begin{pmatrix} A_0 & A_1 \\ A_2 & A_3 \end{pmatrix} := SAT$; $(A'_0, a_0) := A_0$; $(a_1, \ldots, a_{m-r}) := A_1$;
  $j := 1$; $Test := TRUE$;
  repeat
    p-адический подъем:
    $l := 1$; $x := 0$; $c^* := -a_j$;
    repeat
      $\tilde{x} := mod_p \tilde{A}_0^{-1} c^*$; $c^* := (c^* - A_0\tilde{x})/p$; $x := x + l\tilde{x}$; $l := lp$;
    until $\mathrm{Nr}(l) \geq L_j^H[A]$;
    $\begin{pmatrix} \bar{x}_j \\ \xi_j \end{pmatrix} := RationalReconstruction[x, N_j^H, D]$;
    Проверка решения:
    Комментарий: $(e_1, \ldots, e_{m-r}) = I_{m-r}$
    $x_j := T \begin{pmatrix} \bar{x}_j \\ \xi_j e_j \end{pmatrix}$;
    if $Ax_j \neq 0$ **then** $Test := FALSE$;
    $j := j + 1$;
  until $j > m - r$ **or** $Test = FALSE$;
  $p := NextPrime[p]$;
until $Test = TRUE$;
return $\{x_1, \ldots, x_{m-r}\}$.

Рис. 2. Алгоритм "Вычисление рационального базиса однородной системы линейных уравнений с помощью p-адического подъема".

где A_0 – это подматрица размера $r \times r$, и $\det A_0 \neq$ $\neq 0$.

Отдельно рассмотрим случай *однородной системы* ($c = 0$).

3.2.1. P-адический метод решения однородной системы в поле частных

Теорема 1. *Пусть* (9) *– однородная система линейных уравнений, и* $a_j \in R^r$ *– столбцы матрицы* A_1: $A_1 = (a_1, a_2, \ldots, a_{m-r})$. *Тогда системы* $A_0 x_j = -a_j$, $j = 1, \ldots, m-r$, *определены, и их решения* $x_j \in K^r$ *задают базисное множество решений системы* (9):

$$\left\{ T \begin{pmatrix} x_j \\ e_j \end{pmatrix}, \quad j = 1, \ldots, m-r \right\},$$

где e_j *– это столбцы единичной матрицы* $I_{m-r} = (e_1, e_2, \ldots, e_{m-r})$.

Доказательство. Так как $\det A_0 \neq 0$, то каждая система $A_0 x_j = -a_j$ определена. Обозначим $y = T^{-1}x$. По условию теоремы, $(A_0, A_1)y = 0$. Если искать решение в виде $y_j = \begin{pmatrix} x_j \\ e_j \end{pmatrix}$, то придем к системе $A_0 x_j = -a_j$. Линейная независимость векторов y_j следует из линейной независимости векторов e_j, $j = 1, \ldots, m-r$.

Соответствующий алгоритм приведен на рис. 2.

В этом алгоритме оценки для границы p-адического подъема $L_j^H[A]$ вычисляются отдельно для каждой системы. Для их вычисления нужны оценки для числителя и знаменателя решения системы. Они вычисляются для $\mathbb{Z}$ по неравенству Адамара.

В этом алгоритме используются следующие функции:

$D = \sum_{k=1}^{n} \mathrm{Nr}(a_k)$ – оценка для знаменателя в $F[x]$;

$D = \frac{n}{2}\log_w n + \sum_{k=1}^{n} \mathrm{Nr}(a_k)$ – оценка для знаменателя в $\mathbb{Z}$;

$N_j^H = D + \mathrm{Nr}(a_j) - \min\limits_{k=1,\ldots,n} \mathrm{Nr}(a_k)$ – оценка для числителя в $F[x]$;

$N_j^H = \frac{n}{2}\log_w n + D + \mathrm{Nr}(a_j) - \min\limits_{k=1,\ldots,n} \mathrm{Nr}(a_k)$ – оценка для числителя в $\mathbb{Z}$;

$L_j^H[A] = N_j^H + D$ – оценка для границы p-адического подъема в $F[x]$;

$L_j^H[A] = N_j^H + D + \log_w 2$ – оценка для границы p-адического подъема в $\mathbb{Z}$, последнее слагаемое для учета знака;

$NextPrime[p]$ – генератор простых элементов кольца R, который возвращает следующий простой элемент после p;

$FirstPrime$ обозначает первый простой элемент;

$\tilde{R} = R/pR$ – обозначение фактор-кольца;

$DiagonalFormH[mod_p\ A]$ – алгоритм диагонализации в кольце $\tilde{R}$ для однородной системы, вычисляет $\{\tilde{A}_0^{-1}, S, T, r\}$;

$RationalReconstruction[\tilde{x}, N, D]$ – восстановление вектора решения в поле частных кольца R по решению $\tilde{x}$ в кольце R/p^kR, когда числитель и знаменатель ограничены числами N и D (алгоритм П.С. Ванга или улучшенный алгоритм из [20]).

3.2.2. P-адический метод решения неоднородной системы в поле частных

Теперь перейдем к случаю *неоднородной системы* ($c \neq 0$).

Теорема 2. *Пусть (9) – неоднородная система линейных уравнений, S, T и P – матрицы перестановок, такие, что S и T удовлетворяют (10), а матрица P такая, что, если обозначить*

$$(B, b) = PA_0^{-1}(A_1, c_0),$$

$$B = \begin{pmatrix} \bar{b}_1 & \ldots & \bar{b}_{m-r} \\ \beta_1 & \ldots & \beta_{m-r} \end{pmatrix},$$

$$b = \begin{pmatrix} \bar{b} \\ \beta \end{pmatrix},\ \beta, \beta_j \in R,$$

то β будет ненулевым элементом столбца свободных членов. Пусть J – множество номеров столбцов матрицы B с нулевыми элементами в последней строке: $J = \{j \mid \beta_j = 0, j = 1, \ldots, m-r\}$. Обозначим

$$Q = \begin{pmatrix} P & 0 \\ 0 & I_{m-r} \end{pmatrix},\ U = I_{m-r+1} + \sum_{j \in J} E_{1,j+1},$$

$$W = \begin{pmatrix} I_{r-1} & 0 \\ 0 & U \end{pmatrix},$$

$V = QW$, $(\bar{A}_0, a_0, a_1, \ldots, a_{m-r}) = P(A_0, A_1)V$, $\bar{A}_0 \in R^{r \times (r-1)}$, $a_j \in R^r$.

Тогда системы

$$(\bar{A}_0, a_j)x_j = Pc_0, \quad j = 0, 1, \ldots, m-r, \tag{11}$$

определены, решениями этих систем будут векторы

$$x_j = \begin{pmatrix} \bar{x}_j \\ \xi_j \end{pmatrix} = \begin{cases} \begin{pmatrix} I_{r-1} & -\bar{b}_j \\ 0 & 1 \end{pmatrix} PA_0^{-1}c_0, \\ \text{при } j \in J; \\ \begin{pmatrix} I_{r-1} & -\bar{b}_j\beta_j^{-1} \\ 0 & \beta_j^{-1} \end{pmatrix} PA_0^{-1}c_0, \\ \text{при } j \notin J; \end{cases} \tag{12}$$

а базисное множество решений системы $Ax = c$ образуют векторы

$$TV \begin{pmatrix} \bar{x}_j \\ \xi_j \bar{e}_j \end{pmatrix}, \quad j = 0, 1, \ldots, m-r, \tag{13}$$

где $\bar{e}_j$ – столбцы единичной матрицы $I_{m-r+1} = (\bar{e}_0, \bar{e}_1, \ldots, \bar{e}_{m-r})$.

Доказательство. Обозначим $y = V^{-1}T^{-1}x$. Тогда по условию $P(A_0, A_1)Vy = Pc_0$ и $P(A_0, A_1)V = (\bar{A}_0, a_0, a_1, \ldots, a_{m-r})$. Если искать решение в виде $y = \begin{pmatrix} \bar{x}_j \\ \xi_j \bar{e}_j \end{pmatrix}$, то придем к системам (11).

Покажем, что системы (11) определены. Для этого умножим их слева на $PA_0^{-1}P$. Так как $P = P^{-1}$, то

$$PA_0^{-1}P(\bar{A}_0, a_j)x_j = PA_0^{-1}c_0, \quad j = 0, 1, \ldots, m-r.$$

Так как $b = PA_0^{-1}c_0$ и $PA_0^{-1}P(\bar{A}_0, a_0) = I_r$, то первая из систем (11) принимает вид $I_r x_0 = b$. Обозначим $I_r = (\bar{I}, e)$, $e \in R^r$, тогда $PA_0^{-1}P\bar{A}_0 = \bar{I}$, $PA_0^{-1}Pa_0 = e$.

По условию,

$$(\bar{A}_0, a_0, a_1, a_2, \ldots, a_{m-r}) = P(A_0, A_1)\begin{pmatrix} P & 0 \\ 0 & I_{m-r} \end{pmatrix}\begin{pmatrix} I_{r-1} & 0 \\ 0 & U \end{pmatrix}.$$

Отсюда следует, что $PA_0^{-1}P(a_0, \ldots, a_{m-r}) = PA_0^{-1}P(a_0, PA_1)U = (e, B)U$. Обозначим через d_j, $j = 1, \ldots, m-r$, столбцы матрицы $(e, B)U$. Так как $U = I_{m-r+1} + \sum\limits_{j\in J} E_{1,j+1}$ и J – множество номеров столбцов матрицы B с нулевыми элементами в последней строке, то в результате умножения на матрицу U к каждому столбцу с номером $j \in J$ матрицы (e, B) прибавляется столбец e. Поэтому последняя строка матрицы $(e, B)U$ не содержит нулевых элементов:

$$d_j = \begin{cases} b_j & \text{при} \quad j \notin J, \\ b_j + e & \text{при} \quad j \in J. \end{cases}$$

Следовательно, после умножения на $PA_0^{-1}P$ системы (11) принимают вид

$$(\bar{I}, d_j)x_j = b, \quad j = 0, 1, \ldots, m-r, \qquad (14)$$

при этом $\det(\bar{I}, d_j) \neq 0$. Так как у вектора b последний элемент не равен нулю, то решения систем $x_j = \begin{pmatrix} \bar{x}_j \\ \xi_j \end{pmatrix}$ таковы, что $\xi_j \neq 0$. Поэтому векторы $\xi_j \bar{e}_j$, $j = 1, \ldots, m-r$, линейно независимы, следовательно, и векторы (13) линейно независимы. Умножая на обратную матрицу к $(\bar{I}, d_j)$, получим решения систем в виде (12).

Так как $V = QW = \begin{pmatrix} P & 0 \\ 0 & I_{m-r} \end{pmatrix}\begin{pmatrix} I_{r-1} & 0 \\ 0 & U \end{pmatrix}$, то, вычисляя произведение Vx_j, получим базисное множество решений в виде

$$\left\{ T\begin{bmatrix} P\begin{bmatrix} \bar{x}_0 \\ \xi_0 \end{bmatrix} \\ 0 \end{bmatrix}, \quad T\begin{bmatrix} P\begin{bmatrix} \bar{x}_j \\ \xi_j \end{bmatrix} \\ \xi_j e_j \end{bmatrix} \ \forall j \in J, \quad T\begin{bmatrix} P\begin{bmatrix} \bar{x}_j \\ 0 \end{bmatrix} \\ \xi_j e_j \end{bmatrix} \forall j \notin J \right\},$$

где e_j – столбцы единичной матрицы $I_{m-r} = (e_1, \ldots, e_{m-r})$.

Соответствующий алгоритм приведен на рис. 3.

В этом алгоритме оценки для границы p-адического подъема $L_j[A, c]$ вычисляются отдельно для каждой системы. Для кольца $\mathbb{Z}$ они получаются по неравенству Адамара, при этом учитывается, что знаменатель – это минор основной матрицы коэффициентов, а числитель – это минор расширенной матрицы коэффициентов. Это позволяет сделать более тонкую оценку, что может быть существенным, если свободные члены малы. Кроме того, учитывается, что при переходе от системы j к системе $j+1$ меняется только последний столбец матрицы A, обозначим его $\hat{a}_j$.

В алгоритме на рис. 3 дополнительно используются следующие функции:

$D_j = \mathrm{Nr}(\hat{a}_j) + \sum\limits_{k=1}^{n-1} \mathrm{Nr}(a_k)$ – оценка для знаменателя в $F[x]$;

$D_j = \frac{n}{2}\log_w n + \mathrm{Nr}(\hat{a}_j) + \sum\limits_{k=1}^{n-1} \mathrm{Nr}(a_k)$ – оценка для знаменателя в $\mathbb{Z}$;

$N_j = D_j + \mathrm{Nr}(c) - \min\limits_{k=1,\ldots,n-1,j} \mathrm{Nr}(a_k)$ – оценка для числителя в $F[x]$;

$N_j = \frac{n}{2}\log_w n + D_j + \mathrm{Nr}(c) - \min\limits_{k=1,\ldots,n-1,j} \mathrm{Nr}(a_k)$ – оценка для числителя в $\mathbb{Z}$;

$L_j[A, c] = N_j + D_j$ – оценка для границы p-адического подъема в $F[x]$;

$L_j[A, c] = N_j + D_j + \log_w 2$ – оценка для границы p-адического подъема в $\mathbb{Z}$, последнее слагаемое для учета знака;

$MinPrime$ – обозначает наименьший по норме простой элемент;

$NumberOfPrimes = \max(D_j)$ – количество простых элементов в $F[x]$;

Algorithm RationalBasis[A, c]
Input: $A \in R^{n\times m}, c \in R^n$
Output: Решения системы $Ax = c$ или NIL
$p := FirstPrime; i := 1;$
repeat
Инициализация алгоритма:
$\{\tilde{A}_0^{-1}, \tilde{B}, \tilde{b}, S, T, P, r, r_c\} := DiagonalForm[mod_p(A, c)];$
if $r < r_c$ **then return** $NIL;$
Комментарий: $\tilde{B} = (\tilde{b}_1, \tilde{b}_2, \ldots, \tilde{b}_{m-r}),\ I_{m-r+1} = (\bar{e}_0, \bar{e}_1, \ldots, \bar{e}_{m-r}),$
Комментарий: $\tilde{b}_j = \begin{pmatrix} \bar{b}_j \\ \tilde{\beta}_j \end{pmatrix},\ Q = \begin{pmatrix} P & 0 \\ 0 & I_{m-r} \end{pmatrix}.$
$\begin{pmatrix} A_0 & A_1 \\ A_2 & A_3 \end{pmatrix} := SAT;\ \begin{pmatrix} c_0 \\ c_1 \end{pmatrix} := Sc;$
$\hat{c} := Pc_0;\ (\bar{A}_0, a_0) := PA_0P;\ (a_1, \ldots, a_{m-r}) := PA_1;$
$J := \emptyset;$
for $j := 1$ **to** $m - r$ **do if** $\beta_j = 0$ **then** $J := J \cup \{j\};$
$j := 0;\ Test = TRUE;$
repeat
if $j = 0$ **then** $A_0 := (\bar{A}_0, a_0); \tilde{A}_0^{-1} := P\tilde{A}_0^{-1}$
else if $j \in J$
then $A_j := (\bar{A}_0, a_j + a_0);\ \tilde{A}_j^{-1} := \begin{pmatrix} I_{r-1} & -\bar{b}_j \\ 0 & 1 \end{pmatrix} \tilde{A}_0^{-1}$
else $A_j := (\bar{A}_0, a_j);\ \tilde{A}_j^{-1} := \begin{pmatrix} I_{r-1} & -\bar{b}_j\tilde{\beta}_j^{-1} \\ 0 & \tilde{\beta}_j^{-1} \end{pmatrix} \tilde{A}_0^{-1}$
p-адический подъем:
$l := 1; x := 0; c^* := \hat{c};$
repeat
$\tilde{x} := mod_p \tilde{A}_j^{-1} c^*;$
$c^* := (c^* - A_j\tilde{x})/p;$
$x := x + l\tilde{x};\ l := lp;$
until $\mathrm{Nr}(l) \geq L_j[A, c]$
$\begin{pmatrix} \bar{x}_j \\ \xi_j \end{pmatrix} := RationalReconstruction[x, N_j, D_j];$
Проверка решения:
if $j = 0$ **then** $x_0 := T \begin{pmatrix} P \begin{pmatrix} \bar{x}_0 \\ \xi_0 \end{pmatrix} \\ 0 \end{pmatrix}$
else if $j \in J$
then $x_j := T \begin{pmatrix} P \begin{pmatrix} \bar{x}_j \\ \xi_j \end{pmatrix} \\ \xi_j e_j \end{pmatrix}$ **else** $x_j := T \begin{pmatrix} P \begin{pmatrix} \bar{x}_j \\ 0 \end{pmatrix} \\ \xi_j e_j \end{pmatrix};$
if $Ax_j \neq c$ **then** $Test := FALSE;$
$j := j + 1;$
until $j > m - r$ **or** $Test = FALSE$
$i := i + 1; p := NextPrime[p];$
until $Test = TRUE$ **or** $i = NumberOfPrimes$
if $Test = FALSE$ **then return** NIL **else return** $\{x_0, x_1, \ldots, x_{m-r}\}.$

Рис. 3. Алгоритм "Вычисление рационального базиса неоднородной системы линейных уравнений с помощью p-адического подъема".

$NumberOfPrimes = \lfloor w^{\max(D_j)} \,/\, |MinPrime| \rfloor$ – количество простых элементов в $\mathbb{Z}$;
$DiagonalForm[mod_p(A,c)]$ – алгоритм диагонализации в кольце $\tilde{R}$ для неоднородной системы. Результатом вычислений будет $\{\tilde{A}_0^{-1}, \tilde{B}, \tilde{b}, S, T, P, r, r_c\}$. Знак тильды обозначает матрицы над фактор-кольцом $\tilde{R}$. r и r_c – это ранги матриц $mod_p A$ и $mod_p(A,c)$.

3.3. *Оценка сложности p-адического метода решения системы в поле частных*

Будем рассматривать решение системы $Bx = c$ с матрицей коэффициентов $A = (B,c)$ размера $n \times m$ и ранга n. Модуль p в кольце $\mathbb{Z}$ будем выбирать так, чтобы он был наибольшим из чисел, которые помещаются в одно машинное слово, а в кольце $F[x]$ в качестве модуля p будем брать многочлен первой степени.

Будем обозначать $\gamma n^\beta + o(n^\beta)$ оценку сложности умножения матриц порядка n. Здесь γ и β – некоторые постоянные, $3 \geq \beta > 2.376$. Можно рассматривать γ как функцию от β. Например, $\gamma(3) = 1$, $\gamma(\log_2 7)$=1, когда n – некоторая степень 2.

Число мультипликативных операций над элементами, которое достаточно для приведения матрицы размера $n \times m$ рекурсивным методом к диагональному виду, обозначим $\gamma_\beta m n^{\beta-1} + o(mn^{\beta-1})$. Оценки функции γ_β можно найти в [18]. Отметим некоторые значения для γ_β: $\gamma_3 = 1/3$, $\gamma_{\log_2 7} = 7/15$.

Предложение 1. Решение системы в поле частных. Сложность решения системы линейных уравнений с матрицей коэффициентов A размера $n \times m$ в поле частных $\mathbb{Z}$ и в поле частных $F[x]$ p-адическим методом имеет следующие оценки:

$$C_{Cp} \leq \gamma_\beta m n^{\beta-1} + 4n^3(m-n)\rho_A^2(\log_2 w + 1) + O(n^2(m-n)\rho_A) + o(mn^{\beta-1}), \quad (15)$$

$$C_{Bp} = O(mn^{\beta-1} + \rho_A^2 n^3(m-n) + \rho_A n^2 (m-n)\log^2 \rho_A n \log\log \rho_A n). \quad (16)$$

Доказательство. Рассмотрим общую вычислительную модель.

1-й этап. Матрица (A,c) приводится к диагональному виду по простому модулю p с помощью рекурсивного метода. Сложность такой операции составляет

$$C_1 = \gamma_\beta m n^{\beta-1} + o(mn^{\beta-1}).$$

2-й этап. Вычисление $m-n$ матриц A_j^{-1} по модулю p:

$$C_2 = n^2(m-n).$$

3-й этап. Подъем до p^k. Состоит из $k = 2\rho_A n$ шагов подъема. Вычисление произведения матрицы на вектор $\tilde{x} = mod_p \tilde{A}_j^{-1} c^*$ осуществляется за n^2 операций, а произведение матрицы A_j на вектор $\tilde{x}$ – за $n^2\|A\|$ операций. Здесь сложность алгоритма умножения (6) берется, для упрощения, при $\delta = 2$ и $\eta = 1$. Итого,

$$C_3 = 2\rho_A n^3(\|A\| + 1)(m-n).$$

4-й этап. Сложность реконструкции $n(m-n)$ рациональных дробей по их остаткам по модулю $p^{2\rho_A n}$, с учетом оценки (7), составляет

$$C_4 \leq n(m-n)(2\rho_A n)^2 \log_2 w + O(n^2(m-n)\rho_A).$$

Сложность алгоритма складывается из сложности отдельных этапов $C_1 + C_2 + C_3 + C_4$. Отсюда получим:

$$C_{Cp} \leq \gamma_\beta m n^{\beta-1} + 2n^3(m-n)\rho_A(2\rho_A \log_2 w + \|A\| + 1) + o(mn^{\beta-1}) + O(n^2(m-n)\rho_A).$$

Отсюда следует доказываемая оценка в C-модели. Аналогично, с использованием (4) и (5), получается оценка в B-модели.

Это оценки и для кольца целых чисел, и для многочленов над полем, отличие состоит в значениях ρ_A и w. Нужно также отметить, что решение требует проверки подстановкой. С вероятностью $1/p$ полученное решение в кольце целых чисел может оказаться неверным. То есть с вероятностью $1/p^k$ сложность окажется в k раз больше, чем C_p.

4. ВЕРОЯТНОСТНЫЙ МЕТОД РЕШЕНИЯ СИСТЕМЫ В КОММУТАТИВНОЙ ОБЛАСТИ

4.1. *Коммутативная область с единицей*

Пусть R – коммутативная область с единицей, в которой имеется алгоритм вычисления базиса конечнопорожденного идеала, например,

Algorithm IntegerBasis[A, c]
Input: $A \in R^{n\times m}, c \in R^n$
Output: Базис из целых решений или NIL
$\{h, \boldsymbol{x}_1\chi_1^{-1}, \boldsymbol{x}_2\chi_2^{-1}, \ldots, \boldsymbol{x}_h\chi_h^{-1}\} := RationalBasis[A, c]$;
if $h = 0$ **then return** NIL;
$(gcd, u_1, u_2, \ldots, u_h) := GI[(\chi_1, \chi_2, \ldots, \chi_h)]$;
if $gcd \neq 1$ **then return** NIL
else
$k := 1$; **while** $u_k = 0$ **do** $k := k + 1$;
$z_k := \sum_{i=1}^{h} x_i u_i$;
for $i = 1, 2, \ldots, k-1, k+1 \ldots, h$ **do**
$z_i := x_i - x_k(\chi_i - 1)$;
return $\{z_1, z_2, \ldots, z_h\}$.

Рис. 4. Вероятностный алгоритм вычисления базиса, составленного из целых решений системы.

алгоритм вычисления базиса Гребнера. Для $x = (x_1, x_2, \ldots, x_s)$ и $y = (y_1, y_2, \ldots, y_s)$ будем обозначать $\langle x, y\rangle = \sum_{i=1}^{s} x_i y_i$.

Пусть $Ax = c$ – система линейных уравнений над R, и пусть уже найден рациональный базис множества $\mathcal{M}$ ее решений:

$$x_i = \chi_i^{-1}\boldsymbol{x}_i,\ \chi_i \in R\backslash 0,\ \boldsymbol{x}_i \in R^m,\quad i = 1, 2, \ldots, h. \tag{17}$$

Нетрудно показать, что множество $\mathcal{M}$ может быть записано в виде

$$\mathcal{M} = \left\{ \frac{\langle \hat{\boldsymbol{x}}, q\rangle}{\langle \chi, q\rangle} \,\middle|\, q \in R^h, \langle \chi, q\rangle \neq 0 \right\}. \tag{18}$$

Рассмотрим вероятностный алгоритм нахождения базиса, составленного из целых решений этой системы, т.е. решений, лежащих в R^m.

Пусть идеал, порожденный всеми знаменателями $\chi_1, \chi_2, \ldots, \chi_h$ рационального базиса (17), является единичным, $(q_1, q_2, \ldots, q_h)$ – такие числа, что $\sum_{i=1}^{h} q_i\chi_i = 1$. Тогда

$$z = \sum_{i=1}^{h} q_i \boldsymbol{x}_i$$

является целым решением системы $Ax = c$. Если при этом $q_k \neq 0$, то

$$\{z, \boldsymbol{x}_i - z(\chi_i - 1),\ \ i = 1, 2, 3, \ldots, k-1, k+1, \ldots, h\}$$

– линейно независимые целые решения, составляющие базисное множество решений системы $Ax = c$.

В алгоритме на рис. 4 используются следующие функции.

Функция $RationalBasis[A, c]$ вычисляет множество решений системы $Ax = c$, составляющее базисное множество рациональных решений $\{\boldsymbol{x}_1\chi_1^{-1}, \boldsymbol{x}_2\chi_2^{-1}, \ldots, \boldsymbol{x}_h\chi_h^{-1}\}$ и количество h векторов в нем.

Функция $GI[(\chi_1, \chi_2, \ldots, \chi_h)]$ по h числам $\chi_1, \chi_2, \ldots, \chi_h$ вычисляет генератор идеала, порожденного этими числами. Если этот идеал является главным идеалом (gcd), порожденным обратимым элементом, то вычисляются также и множители $q_1, q_2, \ldots, q_h$, такие, что $1 = \sum_{i=1}^{h} q_i\chi_i$, и возвращается значение $gcd = 1$. Если идеал не является главным, то возвращается значение $gcd = 0$.

Если результатом вычислений по этому алгоритму является NIL, то необходимо выполнять следующую итерацию.

Для следующей итерации нужно выбрать случайным образом новые матрицы перестановок S и T и вычислить новый рациональный базис. Алгоритм GI нужно применять к знаменателям всех рациональных решений, полученным ранее.

4.2. Решение системы в евклидовой области p-адическим вероятностным методом

Если область является евклидовой, то алгоритм GI – это расширенный алгоритм Евклида.

Идея применения вероятностного алгоритма на основе равенства (18) для нахождения реше-

ния диофантовой системы линейных уравнений была сформулирована в работе [21] и применена для решения разреженных систем, затем применялась в работе [9] для нахождения одного решения для плотных систем диофантовых уравнений. При этом предполагалось для каждого нового решения строить новую обратную матрицу.

Здесь мы применяем вероятностный метод для вычисления всего базисного множества решений, при этом уменьшаем число обращений матриц, вычисляя одну обратную матрицу для нахождения $m-n$ базисных решений.

Оценка ожидаемого числа случайных рациональных решений, которое необходимо для получения одного целого решения в кольце целых чисел, составляет $O(\rho_A)$ [9]. За одну итерацию будет получено $m-n$ рациональных решений, поэтому ожидаемое количество итераций в кольце целых чисел будет $O(\rho_A/(m-n)))$. Это оценки для матриц коэффициентов общего вида. В некоторых случаях матрицы коэффициентов оказываются очень неудобными для вероятностного метода. Для таких систем вероятностным методом не удается найти решение. Такими являются, например, системы с почти диагональными матрицами, у них знаменатели рациональных решений имеют общие сомножители – диагональные элементы.

Сложность вычисления базисного множества решений в кольце определяется количеством операций в алгоритме решения системы в поле частных и в алгоритме вычисления базиса конечнопорожденного идеала.

Асимптотическая сложность первого из них составляет $O(mn^{\beta-1})$, а второго – $O(m(m-n)+$ $+C_G)$ операций в кольце R. Здесь C_G – количество операций, необходимых для разложения единицы по $m-n$ образующим единичного идеала в кольце R. Для такого разложения единицы можно воспользоваться алгоритмом вычисления базиса Гребнера (см. [16, 22–24]).

Для случая *евклидовой области* R сложность вычисления рационального базиса решений определена в (15), (16). Что касается сложности вычисления НОД для $m-n$ знаменателей, которые по норме не превосходят $n\rho_A$, то эта величина по крайней мере в n раз меньше, чем сложность вычисления рационального базиса. Отсюда следует, что сложность одной итерации определяется формулами (15) и (16).

5. РЕШЕНИЕ СИСТЕМЫ В ЕВКЛИДОВОЙ ОБЛАСТИ p-АДИЧЕСКИМ ДЕТЕРМИНИСТСКИМ МЕТОДОМ

Пусть система $Bx=c$ такова, что $A=(B,c)\in$ $\in\mathbf{R}^{n\times m}$, $c\in\mathbf{R}^n$, $n<m$, $\operatorname{rank}A=n$, $k=m-n$ – число линейно-независимых решений. Пусть $A=(A_0,A_1)$, блок A_0 – квадратный, $\det A_0\neq 0$, и последняя строка матрицы $A_0^{-1}(A_1,c)$ не содержит нулей. Общий случай сводится к данному так же, как в теореме 2.

Обозначим $\boldsymbol{e}_j$ – столбцы единичной матрицы $I_k=(\boldsymbol{e}_1,\ldots,\boldsymbol{e}_k)$ порядка k. Если искать решение системы $Ax=c$ в виде $\begin{pmatrix} x_j \\ \xi_j\mathbf{e}_j \end{pmatrix}$, $x_j\in\mathbf{K}^{n-1}$, $\xi_j\in\mathbf{K}\backslash\{0\}$, $j=1,\ldots,k$, то придем к определенной системе порядка n. Будем решать каждую определенную систему p-адическим методом. В результате получим базисное множество решений системы в виде

$$\left\{\begin{pmatrix} \mathbf{v}_1 \\ \alpha\boldsymbol{e}_1 \end{pmatrix}\delta_1^{-1},\begin{pmatrix} \mathbf{v}_2 \\ \alpha\boldsymbol{e}_2 \end{pmatrix}\delta_2^{-1},\ldots,\begin{pmatrix} \mathbf{v}_k \\ \alpha\boldsymbol{e}_k \end{pmatrix}\delta_k^{-1}\right\}.$$

Обозначим через $\mathbf{u}=(\delta_1,\ldots,\delta_k)$ вектор и через $V=(\mathbf{v}_1,\ldots,\mathbf{v}_k)$ – матрицу, $\mathbf{u}\in\mathbf{R}^k$, $V\in$ $\in\mathbf{R}^{(n-1)\times k}$.

Воспользуемся результатом, полученным в работе [11].

Теорема 3. *Для того чтобы была совместна система $Ax=c$, необходимо и достаточно, чтобы была совместна система*

$$\begin{cases} Vy &= 0 \mod \alpha, \\ \mathbf{u}y &= \alpha. \end{cases} \tag{19}$$

Если Y – множество решений системы (19), то $X=\left\{\begin{pmatrix} y \\ Vy\alpha^{-1} \end{pmatrix}\;\middle|\; y\in Y\right\}$ – множество решений системы $Ax=c$.

Таким образом, задача сводится к решению в факторкольце однородной системы размера $(n-1)\times(m-n)$. Соответствующий алгоритм приведен на рис. 5.

Элементы факторкольца по модулю α, где α – минор порядка n, представляются элементами с нормой $O(n\rho_A)$. Рассмотрим два алгоритма

Algorithm IntegerBasisDeterministic[A, c]
Input: $A \in R^{n\times m}, c \in R^n$
Output: Базис из целых решений или $\emptyset$
$\left\{ \begin{pmatrix} \mathbf{v}_1 \\ \alpha \boldsymbol{e}_1 \end{pmatrix} \delta_1^{-1}, \begin{pmatrix} \mathbf{v}_2 \\ \alpha \boldsymbol{e}_2 \end{pmatrix} \delta_2^{-1}, \ldots, \begin{pmatrix} \mathbf{v}_k \\ \alpha \boldsymbol{e}_k \end{pmatrix} \delta_k^{-1} \right\} := RationalBasis[A, c];$
Комментарий: $\mathbf{u} = (\delta_1, \ldots, \delta_k)$, $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k)$.
$Y := SolutionOfSystem(Vy = 0 \mod \alpha,\ \mathbf{u}y = \alpha);$
if $Y = \emptyset$ **then return** $\emptyset$
else return $\left\{ \begin{pmatrix} y \\ Vy\alpha^{-1} \end{pmatrix} \mid y \in Y \right\}.$

Рис. 5. Детерминистский алгоритм вычисления базиса, составленного из целых решений системы.

умножения матриц порядка n в таком факторкольце.

Первый алгоритм использует быстрое матричное умножение и быстрый алгоритм умножения чисел, тогда, в зависимости от сложности этих алгоритмов, сложность умножения матриц будет оцениваться в пределах от $O(n^5\rho_A^2)$ до $O^{\sim}(n^{\beta+1}\rho_A)$.

Второй алгоритм использует модулярную арифметику. Для него потребуется $O(n\rho_A)$ простых модулей. Для произведения матриц по каждому простому модулю потребуется n^3 операций. Наконец, для восстановления результата нужно восстановить n^2 чисел, и для каждого из них достаточно $O(n^2\rho_A^2)$ операций. Следовательно, сложность такого алгоритма оценивается $O(n^4\rho_A^2)$.

Каждый из алгоритмов завершается вычислением остатка по модулю α для каждого из n^2 элементов матрицы. При этом при использовании стандартного алгоритма умножения и деления чисел потребуется $O(n^2\rho_A^2)$ операций, а при использовании алгоритмов быстрого умножения и деления чисел – $O^{\sim}(n\rho_A)$ операций на один элемент.

Оценки сложности алгоритмов для решения системы линейных уравнений в факторкольце евклидова кольца приведены в [11]. Основываясь на них, приведем без доказательства оценку сложности решения системы линейных уравнений в евклидовых областях для двух случаев. В первом случае будем применять стандартную арифметику с быстрым алгоритмом умножения. Во втором случае матричное умножение будем выполнять с помощью модулярного метода.

Предложение 2. Сложность решения системы линейных уравнений с матрицей коэффициентов A размера $n \times m$ в кольцах $\mathbb{Z}$ и $F[x]$ p-адическим методом с быстрым алгоритмом умножения имеет следующие оценки:

$$C_{C,s}^{m-n\ll n} \le \gamma_\beta n^{\beta-1}m + 7\rho_A^2 n^3(m-n)(\log_2 w + 1) +$$
$$+ 2\gamma_\beta\eta(\lambda_\beta + 4)\rho_A^\delta n^{\delta+1}(m-n)^{\beta-1} +$$
$$+ o(n^{\beta-1}m + \rho_A n^\beta + \rho_A^\delta n^{\delta+1}(m-n)^{\beta-1});$$
$$C_{C,s}^{n\times 2n} \le 7n^4\rho_A^2(\log_2 w + 1) +$$
$$\gamma_\beta\eta\psi_\beta\rho_A^\delta n^{\delta+\beta} + o(\rho_A^\delta n^{\delta+\beta});$$
$$C_{C,s}^{m\ll n} \le 7\rho_A^2 mn^3(\log_2 w + 1) +$$
$$+ 2\gamma_\beta\eta(\lambda_\beta + 6)\rho_A^\delta n^{\beta+\delta-1}m + o(\rho_A^\delta n^{\beta+\delta-1}m);$$
$$C_{B,s} = O(\rho_A^2 n^3 m +$$
$$+ \rho_A(n^\beta + n^2\log\rho_A n)m\log\rho_A n\log\log\rho_A n).$$

Предложение 3. Сложность решения системы линейных уравнений с матрицей коэффициентов A размера $n \times m$ в кольцах $\mathbb{Z}$ и $F[x]$ p-адическим методом с применением модулярного алгоритма умножения матриц имеет следующие оценки:

$$C_C^{m-n\ll n} \le \gamma_\beta n^{\beta-1}m + 4\rho_A^2 n^3(m-n)(\log_2 w + 1) +$$
$$+ 2\gamma_\beta(\lambda_\beta + 6)\rho_A n(m-n)^\beta +$$
$$+ \rho_A^2 n(m-n)^3(3\log_2 w + 49\log_2(m-n) + 28) +$$
$$+ o(\rho_A^2 n^3(m-n) + n^{\beta-1}m);$$
$$C_C^{n\times 2n} \le 2\gamma_\beta\psi_\beta\rho_A n^{\beta+1} + \rho_A^2 n^4(7\log_2 w +$$
$$+ 49(\log_2^2 n + 2\log_2 n) + 4) + o(\rho_A^2 n^4);$$
$$C_C^{m\gg n} \le 2\gamma_\beta(\lambda_\beta + 6)\rho_A n^\beta m +$$
$$+ \rho_A^2 n^3 m(7\log_2 w + 49\log_2 n + 46) + o(\rho_A^2 n^3 m);$$
$$C_B = O(mn^{\beta-1} + \rho_A^2 n^3(m-n) +$$
$$+ \rho_A n^2(m-n)\log^2 n\log^2\rho_A n\log\log\rho_A n).$$

Здесь оценка для B-модели (4), (5), которая обозначается нижним индексом B, получена для общего случая произвольных n и m. Оценка для случая общей вычислительной модели (6) и (7) обозначается нижним индексом C, она получена

отдельно для трех случаев. Верхний индекс содержит обозначение для трех типов матрицы коэффициентов: квазиквадратная матрица коэффициентов $m - n \ll n$, матрица размера $n \times 2n$ и матрица коэффициентов с числом столбцов гораздо большим, чем число строк (когда можно пренебречь n по сравнению с m) $m \gg n$. Обозначаем также

$$\lambda_\beta = \frac{7}{(2^{\beta-2}-1)} \quad \text{и} \quad \psi_\beta = \frac{7 \cdot 2^{\beta-1}}{(2^\beta - 2)(2^{\beta-1}-2)}.$$

6. ЗАКЛЮЧЕНИЕ

Традиционный метод решения систем линейных уравнений в евклидовой области предполагает выполнение унимодулярного преобразования расширенной матрицы коэффициентов системы к форме Смита, либо преобразования этой матрицы, дополненной единичным вектором, к форме Эрмита. В обоих случаях главной проблемой остается неконтролируемый рост величины коэффициентов, появляющихся в промежуточных вычислениях при вычислении унимодулярного сомножителя.

Основной отличительной стороной предлагаемого метода решения является контролируемость коэффициентов в промежуточных вычислениях. Каждый появляющийся в процессе вычислений коэффициент является либо минором исходной матрицы коэффициентов, либо числом в кольце вычетов по модулю такого минора. В обоих случаях оценка его величины легко описывается неравенством Адамара.

P-адические методы очень хорошо себя зарекомендовали во многих задачах компьютерной алгебры, и лучшим методом для решения определенной системы в евклидовой области остается p-адический метод.

На этом фоне выглядит естественной попытка применить p-адический метод для получения базисного множества решений системы в поле частных. Каждый вектор решения образован отношениями чисел, которые не превосходят соответствующих миноров исходной матрицы.

Вычисление базисного множества решений в области можно дальше вести либо вероятностным методом, получая одно целое решение и достраивая рациональный базис до целого базиса, либо детерминистским методом, переходя к эквивалентной системе в кольце вычетов по модулю ведущего минора. Такой "естественный" алгебраический путь оказывается оправданным.

Отметим также, что в этой статье рассматриваются только плотные системы, то есть системы, в которых специально не предполагается большого количества нулевых элементов или специального строения матрицы коэффициентов. С методами решения разреженных систем можно познакомиться по работам [25, 26], с параллельными алгоритмами – по работам [26, 27].

Автор выражает признательность рецензенту за критические замечения, высказанные к исходному варианту рукописи, которые позволили добиться большей прозрачности в окончательном варианте статьи.

СПИСОК ЛИТЕРАТУРЫ

1. *Labahn G., Storjohann A.* Asimptotically fast computation of Hermite normal forms of integer matrices. Proc. ISSAC'96. ACM Press, 1996. P. 259–266.
2. *Aho A.V., Hopcroff J.E., Ullman J.D.* The design and Analysis of Computer Algorithms. Addison-Wesley, 1974. Имеется перевод: *Ахо А., Хопкрофт Дж., Ульман Дж.* Построение и анализ вычислительных алгоритмов. М.: Мир, 1979.
3. *Dixon J.* Exact solution of linear equations using p-adic expansions // Numer. Math. 1982. V. 40. P. 137–141.
4. *Smith H.J.S.* On system of linear indeterminate equations and congruences // Phil. Trans. Royal Soc. London. 1861. V. A151. P. 293–326.
5. *Smith H.J.S.* On the arithmetical invariants of a rectangular matrix of which the constituents are integral numbers. Proc. London. Math. Soc. 1873. V. 4. P. 236–349.
6. *Gregory R.T., Krishnamurthy E.V.* Methods and Applications of Error-Free Computation. Berlin, 1984.
7. *Hurt M.F., Waid C.A.* A generalized inverse wich give all the integral solutions to a system of linear equations // SIAM J. Appl. Math. 1970. V. 10. P. 547–550.
8. *Hafner J.L., McCuley K.S.* Asymptotically fast triangularization of martix over rings // SIAM Journal of Computing. 1991. V. 20. № 6. P. 1068–1083.

9. *Mulders T., Storjohann A.* Diophantine Linear System Solving. Proceedings of ISSAC'99. Vancouver: ACM Press, 1999. P. 181–188.

10. *Von zur Gathen J., Gerhard J.* Modern Computer Algebra. Cambridge University Press, 1999.

11. *Malaschonok G.I.* Solution of Systems of Linear Diophantine Equations // Computer Algebra in Scientific Computing, CASC 2001. Springer, 2001. P. 401–415.

12. *Schönhage A., Strassen V.* Schnelle Multiplikation grosser Zahlen // Computing. 1971. V. 7. P. 281–292.

13. *Schönhage A.* Schnelle Berechnung von Kettenbruchentwicklungen // Acta Informatica. 1971. V. 1. P. 139–144.

14. *Абрамов С.А.* Некоторые оценки, связанные с алгоритмом Евклида // Журнал вычислительной математики и математической физики. 1979. Т. 19. С. 756–760.

15. *Abramov S.A.* A Note on the Number of Division Steps in the Euclidean Algorithm // SIGSAM Bulletin. 2000. V. 34. № 4. Issue 134. P. 1–2.

16. Buchberger B., Collins G.E., Loos R. (eds.) Computer Algebra. Symbolic and Algebraic Computation. New York: Springer, 1982. Имеется перевод: Бухбергер Б., Коллинз Дж., Лоос Р. (ред.) Компьютерная алгебра. Символьные и алгебраические вычисления. М.: Мир, 1986.

17. *Кнут Д.Е.* Искусство программирования на ЭВМ. Т. 2. Получисленные алгоритмы. М.: Мир, 1977.

18. *Malaschonok G.I.* Recursive Method for the Solution of systems of Linear Equations // Sydow A. (ed.) Computational Mathematics. Proceedings of the 15th IMACS World Congress. V. I. Berlin, August 1997. Berlin: Wissenschaft & Technik Verlag, 1997. P. 475–480.

19. *Wang P.S.* A p-adic algorithm for univariate partial factors // Proc. SYMSAC'81. ACM Press, 1981. P. 212–217.

20. *Collins G.E., Encarnacion M.J.* Efficient rational number reconstruction // Journal of Symbolic Computation. 1995. V. 20. P. 287–297.

21. *Giesbrecht M.* Efficient parallel solution of sparse system of linear diophantine equations // Hitz M., Kaltofen E. (eds.) Second Int. Symp. on Parallel Symbolic Computation: PASCO'97. ACM Press, 1997. P. 1–10.

22. *Латышев В.Н.* Комбинаторная теория колец. Стандартные базисы. М.: Издательство МГУ, 1988.

23. *Михалев А.В., Панкратьев Е.В.* Компьютерная алгебра. Вычисления в дифференциальной и разностной алгебре. М.: Издательство МГУ, 1989.

24. *Панкратьев Е.В.* Компьютерная алгебра. Факторизация многочленов. М.: Издательство МГУ, 1988.

25. *Wiedemann D.* Solving sparse linear equations over finite fields // IEEE Trans. Inf. Theory. 1986. IT. 32. P. 54–62.

26. *Kaltofen E., Pan V.* Processor efficient parallel solution of linear systems over an abstract field // Proc. SPAA'91 3rd Ann. ACM Symp. Parallel Algor. Architecture. New York: ACM Press, 1991. P. 180–191.

27. *Kaltofen E.* Blocked iterative sparse linear system solvers for finite fields // Proc. Symp. Parallel Comput. Solving Large Scale Irregular Applic. Stratagem'96. Sophia Antipolis, France, 1996. P. 91–95.

28. *Huang X., Pan V.Y.* Fast rectangula matrix multiplications and improving parallel matrix computations // Proc. PASCO'97. ACM Press, 1997. P. 11–23.

29. *Malaschonok G.I.* Effective Matrix Methods in Commutative Domains // Formal Power Series and Algebraic Combinatorics. Springer, 2000. P. 506–517.

30. *Storjohann A.* Algorithms for Matrix Canonical Forms. Ph. D. Thesis. Swiss Federal Inst. of Technology. Zurich, 2000.