

Из этого вектора, в свою очередь, последовательно переходим в $x_2 = (0, 1, 0, 0, 0, 0, 1)$ с $\mu(x_2) = 3$, $x_3 = (0, 1, 0, 0, 0, 1, 1)$ с $\mu(x_3) = 2$, $x_4 = (0, 1, 0, 1, 0, 0, 1)$ с $\mu(x_4) = 2$ и попадаем в тупиковую точку. Эта ветвь алгоритма потребовала проведения 25 вычислений невязок и так и не привела к успеху.

Предложенный модифицированный алгоритм уже на первом шаге устанавливает запрет на проставление единицы в первой и седьмой координатах вектора и присваивает шестой координате значение 1: $x_1 = 0$, $x_7 = 0$ и $x_6 = 1$. Отметим, что срываются первый и второй случаи стратегии запрета перехода в следующую вершину и блокируются возможности расстановки единиц соответственно по переменным x_7 и x_1 , при этом $\mu(0, 0, 0, 0, 0, 0, 1) = 8$ и является минимальной из всех посчитанных на первом шаге. В результате $x_1 = (0, 0, 0, 0, 0, 1, 0)$. На втором шаге запрещается присваивание единицы переменным x_3 и x_5 , переменной x_2 даётся значение 1 и получается вектор $x_2 = (0, 1, 0, 0, 0, 1, 0)$. Присваивание единицы четвёртой координате приводит к нахождению решения $x_3 = (0, 1, 0, 1, 0, 1, 0)$. При этом модифицированная версия алгоритма потребовала вычисления невязки всего 11 раз.

Экспериментальные исследования, которые проводились на случайных системах неравенств, показали, что эффективность применения как базового алгоритма Балаша, так и его модификации зависит от структуры системы неравенств. Основным достоинством предложенной модификации алгоритма Балаша является исключение попадания в целые классы тупиковых точек.

ЛИТЕРАТУРА

1. Балакин Г. В., Никонов В. Г. Методы сведения булевых уравнений к системам пороговых соотношений // Обзорение прикладной и промышленной математики. 1994. Т. 1. Вып. 3. С. 389–401.
2. Рыбников К. К., Никонов Н. В. Прикладные задачи, сводящиеся к анализу и решению систем линейных неравенств. Метод разделяющих плоскостей // Вестник Московского государственного университета леса — Лесной вестник. 2002. № 2(22). С. 191–195.
3. Анашкина Н. В. Использование алгоритма Балаша для нахождения решения системы линейных ограничений специального вида // Вестник Московского государственного университета леса — Лесной вестник. 2004. № 4(35). С. 176–179.
4. Кофман А., Анри-Лабордер А. Методы и модели исследования операций. М.: Мир, 1977. 432 с.
5. Анашкина Н. В. Обзор методов решения систем линейных неравенств // Вестник Московского государственного университета леса — Лесной вестник. 2004. № 1(32). С. 144–148.
6. Гришукин В. П. Среднее число итераций в алгоритме Балаша // Сб. статей. Численные методы в линейном программировании. М.: Наука, 1973. С. 31–38.

УДК 519.688

СРАВНИТЕЛЬНЫЙ АНАЛИЗ НЕКОТОРЫХ АЛГОРИТМОВ РАСПОЗНАВАНИЯ ГЛАДКИХ ЧИСЕЛ

Д. С. Арбузов, Л. И. Туктарова

Приводятся результаты экспериментальных исследований трёх алгоритмов нахождения чисел, разложимых в заданной факторной базе: просеивания (с делением и логарифмического) и Бернштейна.

Ключевые слова: гладкие числа, просеивание, алгоритм Бернштейна.

Пусть задана факторная база S — множество, состоящее из некоторых простых чисел.

Определение 1. Целое число будем называть S -гладким, если все его простые делители входят в S .

Задачу нахождения большого количества гладких чисел приходится решать в некоторых методах факторизации чисел (например, квадратичное решето, решето числового поля) и дискретного логарифмирования. Рассмотрим два метода её решения: просеивания (в двух вариантах) и Бернштейна.

1. Метод просеивания [1]

Задан многочлен $F(x) \in \mathbb{Z}[x]$, факторная база S и целые числа $A, C, A < C$. Требуется найти все такие $x, A \leq x \leq C$, что $F(x)$ — S -гладкое. Строится таблица, ячейки которой занумерованы целыми числами от A до C . Метод заключается в заполнении таблицы, её изменении и последующем просмотре.

Алгоритм 1. Метод просеивания

Вход: S — факторная база; A, C — целые числа, $A < C$; многочлен $F(x) \in \mathbb{Z}[x]$;
 $M = \max_{A \leq x \leq C} |F(x)|$; первоначальное заполнение таблицы T :

x	A	$A+1$	$\dots$	C
$F(x)$	$F(A)$	$F(A+1)$	$\dots$	$F(C)$

Выход: множество чисел $x, A \leq x \leq C$, таких, что $F(x)$ — S -гладкое.

- 1: Для всех $q \in S, l \in \{1, 2, \dots, \ln M / \ln q\}$
 - 2: найти все решения x_0 сравнения $F(x) = 0 \pmod{q^l}$.
 - 3: Для всех решений x_0
 - 4: содержимое ячеек $T(x)$ с номерами $x = x_0 + hq^l, (A - x_0)/q^l \leq h \leq (C - x_0)/q^l$, поделить на q .
 - 5: Просмотреть содержимое всех ячеек. Значение $F(x)$ является S -гладким тогда и только тогда, когда $T(x) = \pm 1$.
-

В отличие от метода пробных делений, в данном алгоритме все деления выполняются точно, то есть нет «неудачных» делений.

Улучшение алгоритма 1 (логарифмический вариант) заключается в том, что вычисляется не $F(x)$, а $\log |F(x)|$; на шаге 4 вместо деления на q вычитается $\log q$; на шаге 5 значения $T(x)$ сравниваются с 0. В результате получаем выигрыш как по времени выполнения алгоритма, так и по требуемой памяти. Однако из-за погрешностей округления в вычислении логарифма даже для S -гладких значений $F(x)$ можем получить $T(x) \neq 0$, и шаг 5 надо модифицировать следующим образом: значение $F(x)$ является S -гладким тогда и только тогда, когда $|T(x)| \leq E$ для некоторого «малого» E . При этом возможны ошибки первого и второго рода: необнаружение гладкого числа (влияет на эффективность алгоритма) и принятие негладкого числа за гладкое (влияет на корректность алгоритма).

Для оценки величины E проведён следующий эксперимент: $F(x) = (x + m)^2 - N$, где $m = \sqrt{N}$, N — модуль RSA длиной $|N|$ бит; логарифм берётся по основанию 2 и округляется до ближайшего целого снизу; факторная база S состоит из $|S|$ первых простых чисел q , таких, что $\left(\frac{N}{q}\right) = 1$ (параметры соответствуют методу квадратичного решета для факторизации числа N). Результаты представлены в табл. 1. Здесь

Min — минимальное значение $T[x]$, полученное после просеивания и соответствующее негладкому числу; Max — максимальное значение $T[x]$, которое соответствует гладкому числу. Видно, что во всех случаях можно выбрать такое E , что $\text{Max} < E < \text{Min}$. Это позволит избежать ошибок обоих родов.

Т а б л и ц а 1

Оценка параметра E

	$ S = 20$		$ S = 30$		$ S = 50$		$ S = 100$	
$ N $	Min	Max	Min	Max	Min	Max	Min	Max
20	7,6	3,2	8	2,8	9	3,2	10,6	3,2
40	8,8	4,2	8,6	4,6	9,6	4,6	11	7
50	8,8	6	9	5,2	9,8	6,6	10,8	5,8
60	10,2	5,4	10	6,3	10	6,4	11,2	7,4

Эксперименты показали, что логарифмический вариант просеивания работает быстрее, чем вариант с делением; результаты приведены в табл. 2 (CPU Intel Pentium P6100 2,00 ГГц, память 1,7 Гбайт, swap 1,9 Гбайт).

Т а б л и ц а 2

Выигрыш логарифмического просеивания (по времени)

$ S $	$ N $	$C - A$	Выигрыш, %
30	50	2^{17}	41,16
50	50	2^{15}	55,2
50	60	2^{18}	48,72
100	50	2^{13}	45
100	60	2^{17}	45,24
150	50	2^{13}	56
150	60	2^{16}	43
150	70	2^{17}	49

Кроме того, хранение логарифмов чисел вместо самих чисел позволяет экономить используемую память. Так, при размере числа $|N| = 64$ бита, длине отрезка $C - A = 2^{18}$ и факторной базе из 50 элементов во время работы программы, реализующей просеивание с делением, используется память файла подкачки на 50 % (972 Мбайт), в то время как программа с вычислением логарифмов использует лишь 27 % (525 Мбайт). Это сильно влияет на время выполнения программы; в первом случае оно составляет 129 с, во втором — 42 с.

2. Метод Бернштейна [2]

Метод Бернштейна позволяет сократить объём вычислений при помощи использования деревьев. Задача формулируется так: заданы факторная база S и множество целых чисел P . Требуется найти все S -гладкие $p \in P$.

Отличие в постановке задачи от метода просеивания состоит в том, что множество чисел произвольно.

Обозначим $Q = \prod_{s \in S} s$ и найдём остатки от деления Q на числа $p \in P$. Пусть $Q \bmod p = r$. Тогда $r = ab$, где $a = (Q, p)$ — произведение тех простых, которые присутствуют в базе S и в исследуемом числе. Пусть наибольший показатель в каноническом разложении числа p на простые множители не превосходит z . Тогда в каноническом

разложении r^z все простые присутствуют в степени, не меньшей, чем в p . Таким образом, $(r^z \bmod p, p)$ есть часть числа p , разложимая в базе S ; и число p является S -гладким, если $(r^z \bmod p, p) = p$, т. е. $r^z \bmod p = 0$.

Определение 2. Деревом произведений для множества чисел называется двоичное дерево, листья которого соответствуют элементам этого множества и каждому узлу сопоставлено число, равное произведению чисел, соответствующих его потомкам.

Алгоритм 2. Метод Бернштейна

Вход: S — факторная база; множество чисел P ; $|P| \geq |S|$.

Выход: все S -гладкие числа $p \in P$.

- 1: Построить дерево произведений для факторной базы S . В корне этого дерева получим Q .
 - 2: Построить дерево произведений T для множества P (при этом вычисляются только значения, не большие Q , остальные помечаются *).
 - 3: Вычислить дерево остатков $Q \bmod T$ следующим образом: сначала число R , приписанное корню дерева T , заменяется на $Q \bmod R$, а затем происходит спуск по ветвям, во время которого каждое число заменяется на его остаток от деления на новое значение, приписанное его родителю.
 - 4: Найти наименьшее натуральное число k , для которого $\max P \leq 2^{2^k}$.
 - 5: **Для всех** $p \in P$:
 найти $r = Q \bmod p$ в дереве остатков $Q \bmod T$;
 вычислить $s_p := r^{2^k} \bmod p$.
 - 6: Ответ: все числа p , для которых $s_p = 0$.
-

Замечание. Значение z выбирается в виде 2^k для того, чтобы возведение в степень z выполнить за k возведений в квадрат.

В табл. 3 приведены результаты сравнения метода просеивания (с логарифмированием) и метода Бернштейна.

Т а б л и ц а 3

Сравнение методов просеивания и Бернштейна

Метод	$ S = 50$				$ S = 100$			
	$ N = 50$		$ N = 60$		$ N = 50$		$ N = 60$	
	Время, с	Память, Мбайт	Время, с	Память, Мбайт	Время, с	Память, Мбайт	Время, с	Память, Мбайт
Просеивание	4	200	19	880	3	155	23	990
Метод Бернштейна	3	5	13	5	3	29	25	30

Таким образом, метод Бернштейна существенно выигрывает по памяти и времени при небольшой мощности факторной базы, однако при увеличении размеров факторной базы и просеиваемых чисел метод просеивания с логарифмированием догоняет и обгоняет метод Бернштейна по времени, хотя и существенно проигрывает по памяти.

ЛИТЕРАТУРА

1. Глухов М. М., Круглов И. А., Пижур А. Б., Черемушкин А. В. Введение в теоретико-числовые методы криптографии: учебник для вузов. М.: Лань, 2011.
2. Крендалл Р., Померанс К. Простые числа: криптографические и вычислительные аспекты. М.: УРСС, Либроком, 2011.