1 Задачи работы

Элемент a имеет порядок q по модулю p. Найти дискретный логарифм x - такое целое число $1 < x < q$, что $a^x = b\ (mod\ p)$:

1. Ро-методом Полларда.
2. Методом Гельфонда;
3. Методом базы разложения.

В отчёте привести:

1. В случае результативного завершения работы программы: результат дискретного логарифмирования - число x.
2. Для метода Гельфонда:

2.1. Значение s.

2.2. Базу данных вида $\{k, b * a^{-k\,s}\ (mod\ p)\}$, отсортированную по первой координате. При большом объёме базы - первые 5 и последние 5 её элементов.

2.3. Значение t, для которого $a^t = b * a^{-k\,s} (mod\ p)$ для некоторого k. Уравнение для вычисления логарифма x. Метод решения уравнения.

2.4. При невозможности найти дискретный логарифм более нескольких часов – значение t, до которого построена последовательность $a^t (mod\ p)$, время, затраченное на построение второй последовательности, расчетное время, оставшееся до завершения работы (через оценку сложности алгоритма), выводы (объяснение нерезультативного завершения).

3. Для метода базы разложения:

3.1 При результативном завершении работы:

3.1.1 Базу разложения B (первоначальную, измененную (если потребовалось изменение, обосновать необходимость изменения)), при большом объеме базы – число элементов в базе и ее последний элемент.

3.1.2 B-гладкие значения $a^{u_i}\ (mod\ p)$, соответствующие им показатели u_i, при большом объеме данных – по 5 указанных значений

3.1.3 Векторы показателей для B-гладких значений $a^{u_i}\ (mod\ p)$, при большом объеме данных – 5 векторов, соответствующих значениям $a^{u_i}\ (mod\ p)$ из предыдущего пункта.

3.1.4 Показатель v, для которого значение $b^v \ (mod\ p)$ является B-гладким, соответствующий вектор показателей.

3.1.5 Метод, использованный для исключения переменных (если была использована готовая процедура – указать, из какой библиотеки).

3.1.6 Метод, использованный для решения линейного сравнения относительно неизвестного x.

3.2 При невозможности найти дискретный логарифм более нескольких часов:

3.2.1 Данные по пп. 3.1.1–3.1.3 на момент прерывания программы.

3.2.2 Расчетное необходимое число гладких чисел $a^{u_i} \ (mod\ p)$ и время, необходимое для их поиска.

3.2.3 Показатель v, для которого значение $b^v \ (mod\ p)$ является B-гладким, соответствующий вектор показателей.

4. Для ро-метода Полларда:

4.1 Параметры алгоритма (использованное отображение, начальное значение (несколько, если их пришлось менять)).

4.2 При результативном завершении работы – первые 5 и последние 5 значений c, d, $log_a(c)$, $log_a(d)$, число итераций, время работы программы, выводы (объяснение результативного завершения).

4.3 При работе программы более нескольких часов – первые 5 и последние (на момент прерывания программы) 5 значений c, d, $log_a(c)$, $log_a(d)$, число выполненных итераций, время, затраченное на их выполнение, расчетное время, оставшееся до завершения работы (через оценку сложности алгоритма), выводы (объяснение нерезультативного завершения).

2 Теоретические сведения

Задача дискретного логарифмирования в конечном поле F_p формулируется так: для данных целых чисел a и b, $1 < a,\ b < p$, найти логарифм – такое целое

число x, что $a^x \equiv b(mod\ p)$ (если такое число существует). По аналогии с вещественными числами используется обозначение $x = log_a b$.

Безопасность соответствующих криптосистем основана на том, что, зная числа a, x, p вычислить $a^x(mod\ p)$ легко, а решить задачу дискретного логарифмирования трудно.

2.1 Ро-метод Полларда

Случайное отображение f должно обладать не только сжимающими свойствами, но и вычислимостью логарифма. Для дискретного логарифмирования в качестве случайного отображения f чаще всего используются «ветвящиеся» отображения, например:

$$f(c) = \begin{cases} ac, & \text{если } c < \frac{p}{2} \\ bc, & \text{если } c > \frac{p}{2} \end{cases}$$

При $c < \frac{p}{2}$ имеем $log_a f(c) = log_a b + 1$, при $c > \frac{p}{2}$ – $log_a f(c) = log_a c + x$.

Алгоритм:

Вход. Простое число p, число a порядка r по модулю p, целое число b, $1 < b < p$; отображение f, обладающее сжимающими свойствами и сохраняющее вычислимость логарифма.

Выход. Показатель x, для которого $a^x \equiv b(mod\ p)$ (если такое число существует).

Выбрать произвольные целые числа u, v и положить $c \leftarrow a^u b^v\ (mod\ p)$, $d \leftarrow c$.

Выполнять $c \leftarrow f(c)\ (mod\ p), d \leftarrow f(f(d))\ (mod\ p)$, вычисляя при этом логарифмы для c и d как линейные функции от x по модулю r, до получения равенства $c \equiv d(mod\ p)$.

Приравняв логарифмы для c и d, вычислить логарифм x решением сравнения по модулю r. Результат: x или «Решений нет».

Сложность данного алгоритма в среднем составляет O(4,5$\sqrt{(\pi r)/8}$), что приблизительно равно 2.82$\sqrt{r}$ операций вычисления отображения f.

2.2 Метод Гельфонда

Теорема Гельфонда:

Пусть целые числа a и b таковы, что $1 < a, b < p$, где число p – простое, порядок числа a по модулю p равен r и $a^x \equiv b(mod\ p)$. Тогда число x можно найти, выполнив не более чем $2(\sqrt{r} + log_2 r) - 1$ операций умножения по модулю p.

Обобщение теоремы:

Пусть целые числа a и b таковы, что $1 < a, b < p$, где число p – простое, порядок числа a по модулю p равен r и $a^x \equiv b(mod\ p)$. Кроме того, число $r = r_1 r_2$. Тогда число x можно найти, выполнив не более чем $2(\sqrt{r_1} + \sqrt{r_2}) + 8log_2 r_1 r_2 + 2log_2 r_1 - 1$ операций умножения по модулю p.

Метод встречи посередине представляет собой вероятностный вариант алгоритма Гельфонда и предполагает создание базы данных объема $O(\sqrt{r})$ из пар вида $(a^{xj}(mod\ p), x_j)$ для случайных чисел y_j вычисляются значения b^{yi} и сравниваются с базой данных. Сравнение $a^{xj} \equiv b^{yj}(mod\ p)$ означает, что $xy_i \equiv x_j\ (mod\ r)$.

Сложность метода встречи посередине определяется этапами создания базы данных и ее сортировки. Сложность создания базы данных равна $O(\sqrt{r})$, сложность сортировки равна $O(\sqrt{r}\ log\ r)$. Поэтому итоговая сложность равна $O(\sqrt{r}\ log\ r)$.

2.3 Метод базы разложения

Алгоритм:

Вход. $p, a, b\ (1 < b < p), r$ (порядок a по модулю p).

Выход. Показатель x, для которого $a^x \equiv b(mod\ p)$ (если такое число существует).

Строится база разложения $B = \{-1, p_1, p_2, \dots, p_h\}$, где p_i – простые числа, $h = \sqrt{e^{\sqrt{\ln p * lnlnp}}}$.

Случайно выбирается h показателей u_i таких, что $b_i = a^{u_i} \pmod p$ или $p - b_i$ было B-гладким.

$$b_i \equiv \prod_{j=0}^{h} p_j^{\alpha_u} (\mathrm{mod}\, p)$$

Строится матрица A, состоящая из α_i для каждого из u_i, и вектор U из u_i.

Вычисляется показатель v, такой что b^v (mod p) является B-гладким.

$$b^v \equiv \prod_{j=0}^{h} p_j^{\beta_j} (\mathrm{mod}\, p)$$

Из β_i строится вектор.

Логарифмируются соотношения:

$$u_i \equiv \sum_{j=0}^{h} \alpha_{ij} \log_a p_j (\mathrm{mod} r), \; xv \equiv \sum_{j=0}^{h} \beta_j \log_a p_j (\mathrm{mod} r)$$

Методом Гаусса решается СЛАУ $A * X = U$, где $X - log_a p_i$.

Вычисляется xv:

$$xv \equiv \sum_{j=0}^{h} \beta_j \log_a p_j (\mathrm{mod} r)$$

Затем определяется, будет ли это линейное сравнение иметь решение. Для этого применены теоремы:

$$x * v \equiv z \pmod r \quad (1)$$

Если $(v, r) = d$, т. е. число v и модуль r имеет общий делитель d, причем число d не делит z, то сравнение 1 не имеет решений.

Если $(v, r) = 1$, т. е. числа v и r взаимно просты, то сравнение 1 имеет одно и только одно решение.

Если $(v, z) = 1$, т.е числа v и z взаимно простые, то решением сравнения 1 $x \equiv zv^{y(r)-1} \pmod r$, является класс где $\varphi(r)$ – значение функции Эйлера для модуля r.

Если x вычислен, и он удовлетворяет $a^x = b\ (mod\ p)$, то возвращаем x. В противном случае заново выбираем u_i.

3 Ход работы

3.1 Результаты работы программы

Результаты работы программы для каждого из алгоритмов показаны в Таблица 1-3.

Ро-метод Полларда не завершился спустя 7200.083614349365 с ≈ 2 ч, число итераций на момент завершения программы составило 1686164919. Длительность одной итерации составляла $t_0 = 4.27009454011145e - 06$ с, поэтому расчётное время выполнения программы: $t_o * \left(4.5\sqrt{\frac{\pi q}{8}}\right) = 34812211060 * t_0 = 148651.43237651343\ c = 1.0$ д 17.0 ч 17.0 м 31.432376513432246 с.

Таблица 1 – Результаты работы программы, Ро-метод Полларда

i	c	d	$\log_a c$	$\log_a d$
1	205058476140512971345	205058476140512971345	2904056658377588957 72 + 2585125988236945782 33x	2904056658377588957 72 + 2585125988236945782 33x
2	2386792104753574 33327	2679011901383373 73947	2904056658377588957 72 + 1061196056489283 68975x	2904056658377588957 72 + 1061196056489283 68976x
3	2679011901383373 73947	9517196713801532 4344	2904056658377588957 72 + 1061196056489283 68976x	2904056658377588957 72 + 1061196056489283 68978x
4	2135468310358914 08689	2434403589248223 72785	2904056658377588957 72 + 1061196056489	1380126726629926 86514 + 1061196056489

			8368977x	8368979x
5	95171967138015324344	160731072985916415390	290405665837758895772 + 106119605648928368978x	138012672662992686515 + 106119605648928368980x
1686164915	1776452641118839453 05	2585324994322679302 28	138012672663982418923 + 106119605649624801478x	138012672664972103014 + 106119605650321282301x
1686164916	2261416828093835708 6	1066251228617159340 84	138012672663982418923 + 106119605649624801479x	138012672664972103015 + 106119605650321282302x
1686164917	6784250484281507125 8	4526814670684615119 9	138012672663982418924 + 106119605649624801479x	138012672664972103017 + 106119605650321282302x
1686164918	2035275145284452137 74	1026273340120829422 72	138012672663982418925 + 106119605649624801479x	138012672664972103019 + 106119605650321282302x
1686164919	2095909398460700394 78	9288047060149224891	138012672663982418925 + 106119605649624801480x	138012672664972103021 + 106119605650321282302x

Метод Гельфонда завершился с ошибкой памяти спустя 1542.0011518001556 с ≈ 26 м, число итераций на момент завершения программы составило 82481824. Длительность одной итерации составляла $t_0 = 1.8695041853101547e-05$ с, поэтому расчётное время выполнения программы:

$t_o * \left(2\left(\sqrt{q} + \log_2 q\right) - 1\right) = \ 24689511524 * t_0 = 461571.45127381297\ c\ =$ 5.0 д 8.0 ч 12.0 м 51.45127381297061 с.

Число элементов последовательности составляет 12344755696, по причине нехватки памяти было сгенерировано только 82481824 элементов.

Таблица 2 – Результаты работы программы, метод Гельфонда

k	Элемент последовательности $b * a^{-ks}\ (mod\ p)$
0	19
1	51385731510467500035
2	208896679946962551617
3	184515571648194832025
4	30069534993584071109
82481820	168555945435509962853
82481821	264667576553416001620
82481822	157201758929773417332
82481823	285951225495249773821
82481824	216999595626360141632

Метод базы разложения не завершился спустя 7200.079793214798 с ≈ 2 ч, на момент завершения программы не завершилась первая итерация. Длительность одной итерации составляла $t_0 = 7200.079793214798$ с, поэтому расчётное время выполнения программы: $t_o * (\exp(2c\sqrt{\ln p \ln\ln p})) = \ 513186349969 * t_0 =$ $3694982668565454.5\ c\ =\ 117167131.0$ лет 293.0 д 9.0 ч 30.0 м 54.5 с.

Количество элементов базы разложения составило 147. База выглядит следующим образом: [-1, 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431, 433, 439, 443, 449, 457, 461, 463, 467, 479, 487, 491, 499, 503, 509, 521, 523, 541, 547, 557, 563, 569,

571, 577, 587, 593, 599, 601, 607, 613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683, 691, 701, 709, 719, 727, 733, 739, 743, 751, 757, 761, 769, 773, 787, 797, 809, 811, 821, 823, 827, 829, 839]

Таблица 2 – Результаты работы программы, метод базы разложения

u_i	$a^{u_i} \ (mod\ p)$	Вектор
149441747697880250995	173672367774461869736	[0, 0]
268968803046954469405	111738727424073080686	[-310.0, -2.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

		0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -1.0, -1.0, 0.0, 0.0, -1.0, 0.0, 0.0]
247325289018140554832	159440643895600706693	[0, 0]
195033522991596933390	251560136636854265812	[-290.00000000000006, -0.3333333333333333, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,

		0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, -241.4, -257.00000000000006, -210.84615384615387, -241.0, -289.0, -16.333333333333332, 0.0, 0.0, 0.0]
10086003068576233837	23479125686046976053	[0.09657320872274133, 0.0004615207107418945, -0.0, -0.0, -0.0, -0.0, -0.0, 0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.008307372793354102, -0.0, -0.0, -0.004153686396677051, -0.0, -0.0, -0.0, -0.0, -0.0, -0.004153686396677051, -0.0, -

		0.0, -0.00415368639667705l, -0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.0, -0.00415368639667705l, -0.0, 0.9351332641052267, 0.9968847352024922, 0.8125249620576723, 0.9345794392523364, 1.1228798892350293, 0.06276681666089766, -0.0, -0.0, 1.0]

4 Выводы

В ходе выполнения лабораторной работы были реализованы три различных алгоритма для решения задачи дискретного логарифмирования. Нерезультативное завершение работы каждого из алгоритмов (по причине нехватки памяти или большого значения времени работы программы) показало, что такая задача сложна, и для её решения потребуется либо усовершенствованное оборудование, либо проектирование решений с учетом возможности параллельных вычислений.