Математические основы защиты информации и информационной безопасности

Лабораторная работа №7 - Дискретное логарифмирование в конечном поле

Кейела Патачона, группа НПМмд-02-21

Содержание

List of Figures

1 Цель и задание работы

Цель

Научиться дискретному логарифмированию в конечном поле

Задания к лабораторной работе

1. Реализовать алгоритм программно.
2. Получить у преподавателя задание, содержащее числа p, a, b и вычислить логарифм.

2 Выполнение работы

2.1 Теоретическая часть

Задача дискретного логарифмирования, как и задача разложения на множители, применяется во многих алгоритмах криптографии с открытым ключом. Предложенная в 1976 году У. Диффи и М.Хеллманом для установления сеансового ключа, та задача послужила основой для создания протоколов шифрования и цифровой подписи, доказательств с нулсным разглашением и других криптографических протоколов.

Пусть над некоторым множеством Ω произвольной природы определены операции сложения « $+$ » и умножения «$.$». Множество Ω называется кольцом если выполняются следующие условия: 1. Сложение коммутативно: $a + b = b + c$ для любых $a, b \in \Omega$; 2. Сложение ассоциативно: $(a + b) + c = a + (b + c)$ для любых $a, b, c \in \Omega$; 3. Существует нулевой элемент $0 \in \Omega$ такой, что $a+0 = a$ для любого $a \in \Omega$; 4. Для каждого элемента $a \in \Omega$ существует противоположный элемент $-a \in \Omega$ такой, что $-a + a = 0$; 5. Умножение дистрибутивно относительно сложения:

$$a.(b + c) = a.b + a.c, \quad (a + b).c = a.c + b.c,$$

для любых $a, b, c \in \Omega$.

Если в кольце Ω умножение коммутативно: $a.b = b.a$ для любых $a, b \in \Omega$, то кольцо называется *коммунтативным*.

Если в колые Ω умножение ассоциативно: $(a.b).c = a.(b.c)$ для любых $a, b, c \in$

Ω, то кольцо называется *ассоциативным*.

Если в кольце Ω существует единичный элемент e такой, что $a.e = e.a = a$ для любого $a \in \Omega$, то кольцо называется кольцом с единицей.

Если в ассоциативном, коммутативном кольце с единицей для каждого ненулевого элемента a существует обратный элемент $a^{-1} \in \Omega$ такой, что $a^{-1}.a = a.a^{-1} = e$, то кольцо называется *полем*.

Пусть $m \in N, m > 1$. Целые числа a и b называются *сравнимыми по модулю* m (обозначается $a \equiv b$(*mod m*)), если разность $a - b$ делится на m Некоторые свойства отношения сравнимости:

1. *Рефлексивность:* $a \equiv a$(*mod m*).
2. *Симметричность:* если $a \equiv b$(*mod m*), то $b \equiv a$(*mod m*).
3. *Транзитивность:* если $a \equiv b$(*mod m*) и $b \equiv c$(*mod m*), $a \equiv c$(*mod m*).

Отношение, обладающее свойством рефлексивности, симметричности и транитимности, называется *отношением эквивалентности*. Отношение сравнимости является отношением эквивалентности на множестве Z целых чисел.

Отношение эквивалентности разбивает множество, на котором оно определено, на *классы эквивалентности*. Любые два класса эквивалентности либо не пересекаются, либо совпадают.

Классы эквивалентности, определяемые отношением сравнимости, называются *классами вычетов по модулю m*. Класс вычетов, содержащий число a, обозначается a*(mod m)* и представляет собой множество чисел вида $a + km$, где $k \in Z$: число a называется представителем этого класса вычетов.

Множество классов вычетов по модулю m обозначается Z/mZ, состоит ровно из m элементов и относительно операций сложения и умножения является *кольцом классов вычетов по модулю m*.

Пример. Если $m = 2$, то $Z/2Z$ = *{0 (mod 2), 1 (mod 2))*, где *0 (mod 2)* = $2Z$ множество всех четных чисел, *1 (mod 2)* = $2Z + 1$ множество всех нечетных чисел.

Обозначим $F_p = Z/pZ\, p-$ простое целое число и назовем конечным полем и p элементов. Задача дискретного логарифмирования в конечном поле формулируется так: для данных целых чисел a и b, $a > 1, b > p$, найти логарифм - такое целое число x, что $a^x \equiv b$(*mod m*) (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 решит задачу дискретного логарифмирования трудно. Рассмотрим p-Метод Полларда, который можно применить и для задач дискретного логарифмирования. При этом случайное отображение f должно обладать не только сжимающими свойствами, но и вычислимостью логарифма (логарифм числа $f(c)$ можно выразить через неизвестный логарифм x и $log_a f(c)$). Для дискретного логарифмирования в качестве случайного отображения f чаще всего используются ветвящиеся отображения, например:

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

При $c < \frac{p}{2}$ имеем $log_a f(c) = log_a c + 1$, при $c \geq \frac{p}{2} - log_a f(c) = log_a c + 1$

2.2 Алгоритм, реализующий p—Метод Полларда для задач дискретного логарифмирования.

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 такой показатель существует. 1. Выбрать произвольные целые числа u, v положить $c \leftarrow a^u b^v$*(mod p)*, $d \leftarrow c$ 2. Выполнять $c \leftarrow f(c)$*(mod p)*, $d \leftarrow f(f(c))$*(mod p)*, вычисляя при этом логарифмы для c и d как линейные функции от x по модулю r, до получения

равенства $c \equiv d(mod\ p)$. 3. Приравняв логарифмы для c и d, вычислить логарифм x решением сравнения по модулю r. Результат: x или “Решений нет”.

Пример. Решим задачу дискретного логарифмирования $10^x \equiv 64(mod\ 107)$, используя p—Метод Полларда. Порядок числа 10 по модулю 107 равен 53.

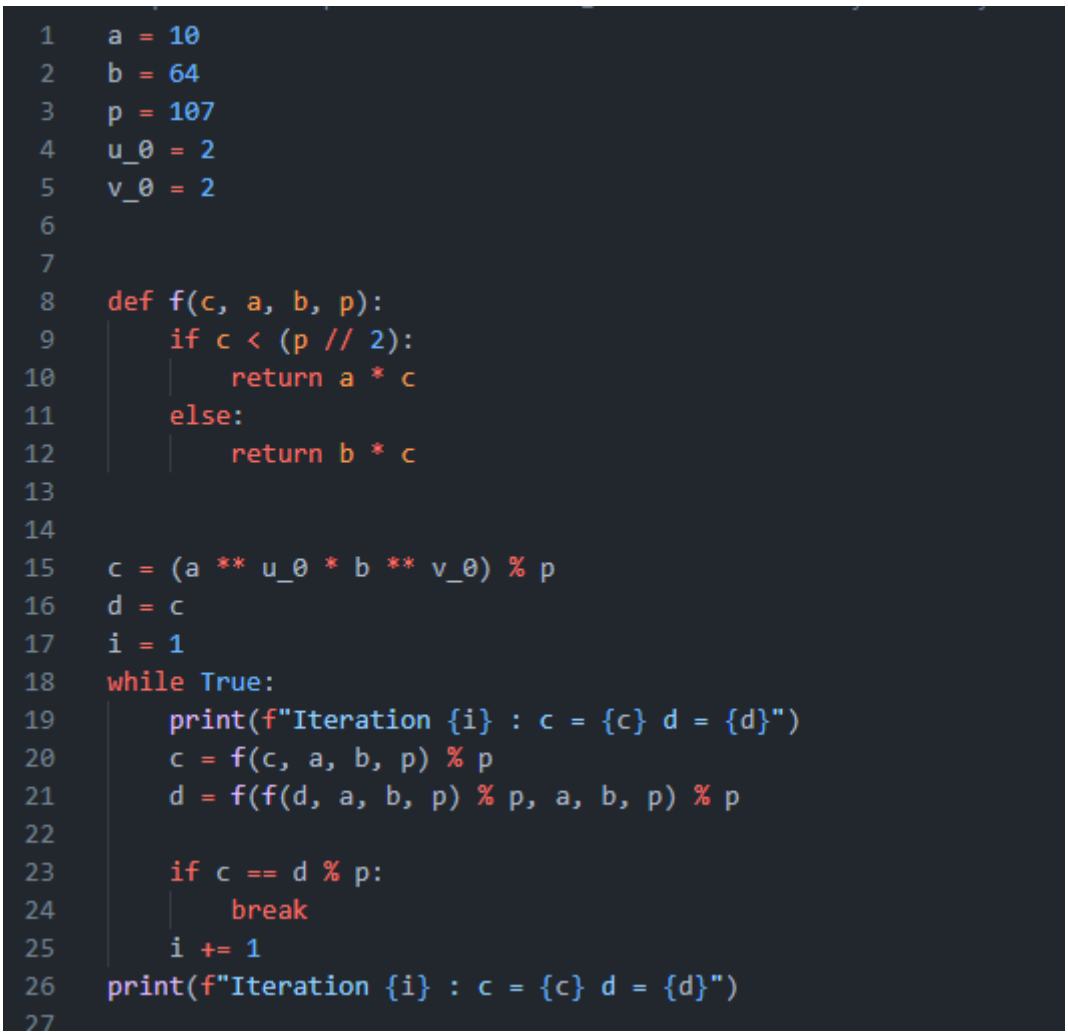
Выберем отображение $f(c) \equiv 10c(mod\ 107)$ при $c < 53$, $f(c) \equiv 64c(mod\ 107)$ при $c \geq 53$. Пусть $u = 2, v = 2$. Результаты вычислений запишем в таблицу:

Номер шага	c	$\log_a c$	d	$\log_a d$
0	4	2+2 x	4	2+2 x
1	40	3+2 x	76	4+2 x
2	79	4+2 x	56	5+3 x
3	27	4+3 x	75	5+5 x
4	56	5+3 x	3	5+7 x
5	53	5+4 x	86	7+7 x
6	75	5+5 x	42	8+8 x
7	92	5+6 x	23	9+9 x
8	3	5+7 x	53	11+9 x
9	30	6+7 x	92	11+11 x
10	86	7+7 x	30	12+12 x
11	47	7+8 x	47	13+13 x

Figure 2.1: Пример дискретного логарифмирования

Приравниваем логарифмы, полученные на 11—м шаге: $7 + 8x \equiv 13 + 13x(mod\ 107)$. Решая сравнение первой степени, получаем: $x = 20(mod\ 53)$.

Проверка: $10^{20} \equiv 64(mod\ 107)$.



```python
a = 10
b = 64
p = 107
u_0 = 2
v_0 = 2


def f(c, a, b, p):
    if c < (p // 2):
        return a * c
    else:
        return b * c


c = (a ** u_0 * b ** v_0) % p
d = c
i = 1
while True:
    print(f"Iteration {i} : c = {c} d = {d}")
    c = f(c, a, b, p) % p
    d = f(f(d, a, b, p) % p, a, b, p) % p

    if c == d % p:
        break
    i += 1
print(f"Iteration {i} : c = {c} d = {d}")
```

Figure 2.2: Алгоритм p—Полларда

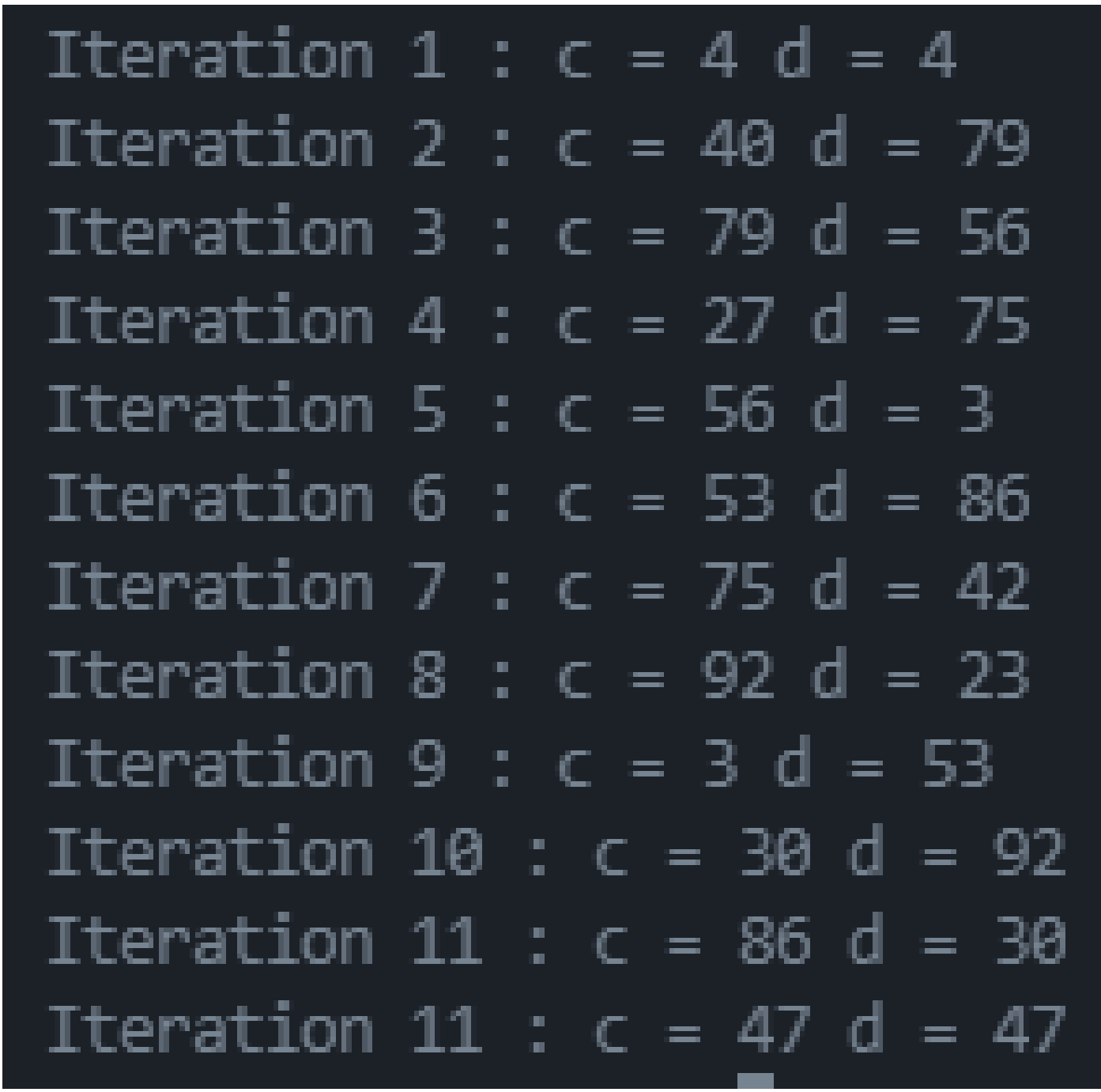


Figure 2.3: Результат алгоритма p—Полларда

3 Выводы

Мной была узчена тема дискретного логарифмирования в конечном поле.

Список литературы

1. Инструкция к лабораторной работе №7