

# Лабораторная работа 7

Дискретное логарифмирование в конечном поле

---

Баранов Иван

2022 Москва

RUDN University, Moscow, Russian Federation

## Цель работы

---

- Ознакомиться с задачей дискретного логарифмирования в конечном поле
- Рассмотреть теоретические основы представленного алгоритма
- Реализовать  $p$ -Метод Полларда для задач дискретного логарифмирования

## Описание задачи

---

Задача дискретного логарифмирования, как и задача разложения на множители, применяется во многих алгоритмах криптографии с открытым ключом.

Предложена в 1976 году У. Диффи и М. Хеллманом для установления сеансового ключа.

Эта задача послужила основой для создания протоколов шифрования и цифровой подписи, доказательств с нулевым разглашением и других криптографических протоколов.

Пусть над некоторым множеством  $\Omega$  произвольной природы определены операции сложения «+» и умножения « $\cdot$ ». Множество  $\Omega$  называется *кольцом*, если выполняются следующие условия:

1. Сложение коммутативно:  $a + b = b + a$  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:

$$a \cdot (b + c) = a \cdot b + a \cdot c, (a + b) \cdot c = a \cdot c + b \cdot c,$$

для любых  $a, b, c \in \Omega$ .

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

Если в кольце  $\Omega$  умножение ассоциативно:  $(a \cdot b) \cdot c = a \cdot (b \cdot c)$  для любых  $a, b, c \in \Omega$ , то кольцо называется *ассоциативным*.

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

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

## Свойства отношения сравнимости

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

1. *Рефлексивность*:  $a \equiv a \pmod{m}$ .
2. *Симметричность*: если  $a \equiv b \pmod{m}$ , то  $b \equiv a \pmod{m}$ .
3. *Транзитивность*: если  $a \equiv b \pmod{m}$  и  $b \equiv c \pmod{m}$ , то  $a \equiv c \pmod{m}$ .

Отношение, обладающее свойством рефлексивности, симметричности и транзитивности, называется *отношением эквивалентности*. Отношение сравнимости является отношением эквивалентности на множестве  $\mathbb{Z}$  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  $a \pmod m$  или  $\bar{a}$  и представляет собой множество чисел вида  $a + km$ , где  $k \in \mathbb{Z}$ ; число  $a$  называется представителем этого класса вычетов.

Множество классов вычетов по модулю  $m$  обозначается  $\mathbb{Z}/m\mathbb{Z}$ , состоит ровно из  $m$  элементов и относительно операций сложения и умножения является *кольцом классов вычетов по модулю  $m$* .

## Постановка задачи

Пример. Если  $m = 2$ , то  $\mathbb{Z}/2\mathbb{Z} = \{0 \pmod{2}, 1 \pmod{2}\}$ , где  $0 \pmod{2} = 2\mathbb{Z}$  – множество всех четных чисел,  $1 \pmod{2} = 2\mathbb{Z} + 1$  – множество всех нечетных чисел.

Обозначим  $F_p = \mathbb{Z}/p\mathbb{Z}$ ,  $p$  – простое целое число и назовем конечным полем из  $p$  элементов. Задача дискретного логарифмирования в конечном поле  $F_p$  формулируется так: для данных целых чисел  $a$  и  $b$ ,  $a > 1, b > p$ , найти логарифм – такое целое число  $x$ , что  $a^x \equiv b \pmod{p}$  (если такое число существует). По аналогии с вещественными числами используется обозначение  $x = \log_a b$ .

## **p-Метод Полларда**

---

Безопасность соответствующих криптосистем основана на том, что, зная числа  $a, x, p$  вычислить  $a^x \pmod p$  легко, а решить задачу дискретного логарифмирования трудно. Рассмотрим р-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  $c > \frac{p}{2}$  –  $\log_a f(c) = \log_a c + x$ .

**Алгоритм, реализующий р-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  $a^x \equiv b \pmod{p}$ , если такой показатель существует.

1. Выбрать произвольные целые числа  $u, v$  и положить  $c \leftarrow a^u b^v \pmod{p}$ ,  $d \leftarrow c$ .
2. Выполнять  $c \leftarrow f(c) \pmod{p}$ ,  $d \leftarrow f(f(d)) \pmod{p}$ , вычисляя при этом логарифмы для  $c$  и  $d$  как линейные функции от  $x$  по модулю  $r$ , до получения равенства  $c \equiv d \pmod{p}$ .
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  $10^x \equiv 64 \pmod{107}$ , используя р-Метод Полларда. Порядок числа 10 по модулю 107 равен 53.

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

Приравниваем логарифмы, полученные на 11-м шаге:  $7+8x \equiv 13+13x \pmod{53}$ . Решая сравнение первой степени, получаем:  $x \equiv 20 \pmod{53}$ .

Проверка:  $10^{20} \equiv 64 \pmod{107}$ .

# Реализация алгоритма

---



# Реализация алгоритма

```
1 def euclid(a, b):
2     if b == 0:
3         return a, 1, 0
4     else:
5         d, x, y = euclid(b, a % b)
6         return d, y, x - (a // b) * y
7
8
9 def func(x, a, b, G, H, P, Q):
10     sub = x % 3
11
12     if sub == 0:
13         x = x * G % P
14         a = (a + 1) % Q
15
16     if sub == 1:
17         x = x * H % P
18         b = (b + 1) % P
19
20     if sub == 2:
21         x *= x % P
22         a *= 2 % Q
23         b *= 2 % Q
24
25     return x, a, b
26
27
28 def pollard(G, H, P):
29     Q = int((P - 1) // 2)
30
31     x = G*H
32     a = 1
33     b = 1
34
35     X = x
36     A = a
37     B = b
38
39     for i in range(1, P):
40         x, a, b = func(x, a, b, G, H, P, Q)
41         X, A, B = func(X, A, B, G, H, P, Q)
42         X, A, B = func(X, A, B, G, H, P, Q)
43
44         if x == X: break
45
46     nom = a-A
47     denom = B-B
48
49     return (euclid(denom, Q)[1] * nom) % Q
50
51
52 def main():
53     a = 10
54     b = 64
55     p = 107
56
57     x = pollard(a, b, p)
58     print(f"{a}^{x} = {b} (mod {p})")
59
60
61 if __name__ == '__main__':
62     main()
63
```

Figure 1: Код на языке Python

Результат работы данного алгоритма:

$$10^{20} = 64(mod 107)$$

## **Выводы**

---

- Ознакомились с задачей дискретного логарифмирования в конечном поле
- Рассмотрели теоретические основы представленного алгоритма
- Реализовали р-Метод Полларда для задач дискретного логарифмирования

