Отчёт по лабораторной работе №7. Дискретное логарифмирование в конечном поле

Дисциплина: Математические основы защиты информации и информационной безопасности

Студент: Леонова Алина Дмитриевна, 1032212306
Группа: НФИмд-01-21
Преподаватель: д-р.ф.-м.н., проф. Кулябов Дмитрий Сергеевич

25 декабря, 2021, Москва

Цель и задание работы

Цель работы

Целью данной работы является ознакомление с ρ-методом Полларда для задач дискретного логарифмирования и его реализация на выбранном языке программирования.

Задание

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

Теоретическое введение

Дискретное логарифмирование в конечном поле

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

Алгоритм, реализующий ρ–Метод Полларда для задач дискретного логарифмирования.

Алгоритм, реализующий ρ–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 и положить $c \leftarrow a^u b^v \ (mod\ p), d \leftarrow c$.
2. Выполнять $c \leftarrow f(c)(mod\ p), d \leftarrow f\big(f(d)\big)(mod\ p)$, 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".

Figure 1: Торетическая справка из задания

Пример

Пример. Решим задачу дискретного логарифмирования $10^x \equiv 64 \ (mod\ 107)$, используя р–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

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

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

Выполнение лабораторной работы

Промежуточные функции

```python
# Расширенный алгоритм Евклида
# d = НОД(a,b) = ax + by
def nod3(a, b):
    if a == 0 or b == 0:
        return max(a, b)
    if a == 1 or b == 1:
        return 1
    if abs(a) < abs(b):
        a, b = abs(b), abs(a)

    x, y = [1,0], [0,1]
    a_, b_ = a, b

    while b_ != 0:
        a_, b_, p = b_, a_ % b_, a_ // b_

        x[0], x[1] = x[1], x[0] - p*x[1]
        y[0], y[1] = y[1], y[0] - p*y[1]

    d = a_
    #print(a,'*',x[1],' + ',b,'*',y[1],' = ',d)
    return d, abs(y[0]), abs(x[0])


# Функция
def f(c, u, v):
    if c < r:
        return a*c % p, u+1, v
    else:
        return b*c % p, u, v+1

# Печать промежуточных шагов
def pr(c,uc,vc,d,ud,vd):
    print(' ',c,'   ',uc,' + ',vc,'x    ', d,'   ',ud,'+',vd,'x')
```

Figure 3: Промежуточные функции

p-метод Полларда для задач дискретного логарифмирования

```python
# p-метод Полларда для задач дискретного логарифмирования
def Pollard_log(a, p, r, b, u, v):
    c = a**u * b**v % p
    d = c
    uc, vc = u, v
    ud, vd = u, v

    print('  c       log_c       d       log_d')
    print('--------------------------------------')
    pr(c,uc,vd,d,ud,vd)

    c, uc, vc = f(c, uc, vc)
    c %= p
    d, ud, vd = f(*f(d, ud, vd))
    d %= p
    pr(c,uc,vd,d,ud,vd)

    while c%p != d%p:
        c, uc, vc = f(c, uc, vc)
        c %= p
        d, ud, vd = f(*f(d, ud, vd))
        d %= p
        pr(c,uc,vd,d,ud,vd)

    v = vc - vd
    u = ud - uc

    d, x, y = nod3(v, r)

    while d != 1:
        v /= d
        u /= d
        r /= d
        d, x, y = nod3(v, r)
        pr(c,uc,vd,d,ud,vd)

    return x*u % r
```

Figure 4: p-метод Полларда для задач дискретного

Результат проверка работы алгорима

```
In [17]: runfile('E:/GitHub/1.2-IS/Lab_7/
L7_Leonova.py', wdir='E:/GitHub/1.2-IS/
Lab_7')
р-метод Полларда ля задач дискретного
логарифмирования
  c       log_c       d      log_d
--------------------------------------
  4     2  +  2 x     4     2 + 2 x
  40     3  +  2 x     79     4 + 2 x
  79     4  +  3 x     56     5 + 3 x
  27     4  +  5 x     75     5 + 5 x
  56     5  +  7 x     3     5 + 7 x
  53     5  +  7 x     86     7 + 7 x
  75     5  +  8 x     42     8 + 8 x
  92     5  +  9 x     23     9 + 9 x
  3     5  +  9 x     53     11 + 9 x
  30     6  +  11 x     92     11 + 11 x
  86     7  +  12 x     30     12 + 12 x
  47     7  +  13 x     47     13 + 13 x
20

In [18]:
```

Figure 5: Результат выполнения L7_Leonova.py

Результат выполнения программы, проверка реализации р-метод Полларда для задач дискретного логарифмирования на заданном примере (см. рис. 5).

Выводы

Цель лабораторной работы была достигнута, ρ-метод Полларда для задач дискретного логарифмирования был реализован на языке программирования Python и проверен на заданном примере.