Дискретное логарифмирование в конечном поле

Гаджиев Нурсултан Тофик оглы

2022 Moscow, Russia

RUDN University, Moscow, Russian Federation

Цель работы

Цель работы

Реализация алгоритма, реализующий p-метод Полларда для задач дискретного логарифмирования.

Задачи

Задачи

1. Реализовать алгоритм, реализующий p-метод Полларда для задач дискретного логарифмирования.

Реализация

1. Написал функцию ext_euclid и inverse (рис. 1)

```python
def ext_euclid(a, b):
    """
    Extended Euclidean Algorithm
    :param a:
    :param b:
    :return:
    """
    if b == 0:
        return a, 1, 0
    else:
        d, xx, yy = ext_euclid(b, a % b)
        x = yy
        y = xx - (a // b) * yy
        return d, x, y
def inverse(a, n):
    """
    Inverse of a in mod n
    :param a:
    :param n:
    :return:
    """
    return ext_euclid(a, n)[1]
```

Figure 1: Функция для расширенного алгоритма Евклида и обратного значнения

2. Написал функцию xab (рис. 2)

```
23 ▼ def xab(x, a, b, xxx_todo_changeme):
24       """
25       Pollard Step
26       :param x:
27       :param a:
28       :param b:
29       :return:
30       """
31       (G, H, P, Q) = xxx_todo_changeme
32       sub = x % 3 # Subsets
33
34 ▼     if sub == 0:
35           x = x*xxx_todo_changeme[0] % xxx_todo_changeme[2]
36           a = (a+1) % Q
37
38 ▼     if sub == 1:
39           x = x * xxx_todo_changeme[1] % xxx_todo_changeme[2]
40           b = (b + 1) % xxx_todo_changeme[2]
41
42 ▼     if sub == 2:
43           x = x*x % xxx_todo_changeme[2]
44           a = a*2 % xxx_todo_changeme[3]
45           b = b*2 % xxx_todo_changeme[3]
46
47       return x, a, b
```

Figure 2: Функция xab

3. Написал функцию pollard (рис. 3)

```python
def pollard(G, H, P):

    # P: prime
    # H:
    # G: generator
    Q = int((P - 1) // 2)  # sub group
    x = G*H
    a = 1
    b = 1
    X = x
    A = a
    B = b
    for i in range(1, P):
        x, a, b = xab(x, a, b, (G, H, P, Q))
        X, A, B = xab(X, A, B, (G, H, P, Q))
        X, A, B = xab(X, A, B, (G, H, P, Q))

        if x == X:
            break
    nom = a-A
    denom = B-b
    # Необходимо вычислить обратное значение, чтобы правильно вычислить дробь по модулю q.
    res = (inverse(denom, Q) * nom) % Q


    if verify(G, H, P, res):
        return res

    return res + Q
```

Figure 3: Функция для алгоритма pollard

4. Написал функцию verify и блок работы программы(рис. 4)

```
77 ▼ def verify(g, h, p, x):
78       """
79       Verifies a given set of g, h, p and x
80       :param g: Generator
81       :param h:
82       :param p: Prime
83       :param x: Computed X
84       :return:
85       """
86       return  pow(g, x, p)== h
87
88 ▼ args = [
89       (10, 64, 107),
90   ]
91
92
93 ▼ for arg in args:
94       res = pollard(*arg)
95       print(arg, ': ', res)
96       print("Validates: ", verify(arg[0], arg[1], arg[2], res))
97       print()
```

Figure 4: Функция verify и блок работы программы

Результат

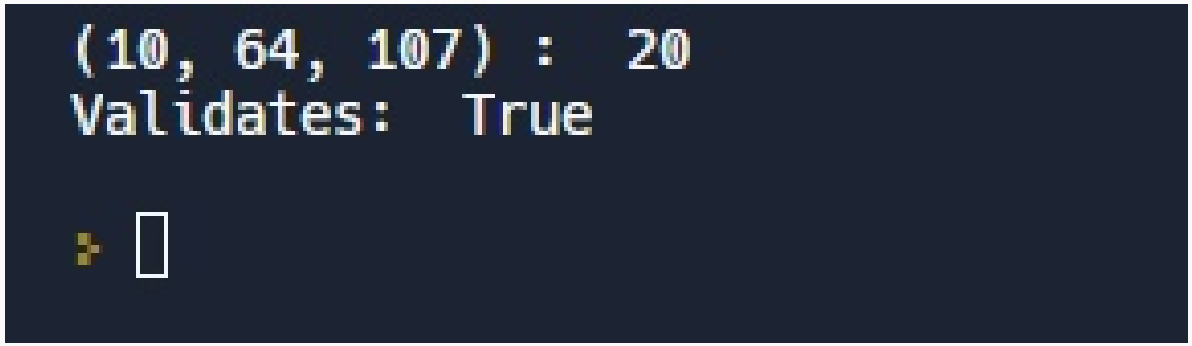

Figure 5: Результат алгоритма

Вывод

Реализовал реализующий p-метод Полларда для задач дискретного логарифмирования.

Спасибо за внимание