

Дискретное логарифмирование в конечном поле

Гебриал Ибрам Есам Зекри¹

2022 Moscow, Russia

¹RUDN University, Moscow, Russian Federation

Цель работы

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

Задачи

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

Реализация

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

```
1 def ext_euclid(a, b):
2     """
3     Extended Euclidean Algorithm
4     :param a:
5     :param b:
6     :return:
7     """
8     if b == 0:
9         return a, 1, 0
10    else:
11        d, xx, yy = ext_euclid(b, a % b)
12        x = yy
13        y = xx - (a // b) * yy
14        return d, x, y
15 def inverse(a, n):
16     """
17     Inverse of a in mod n
18     :param a:
19     :param n:
20     :return:
21     """
22     return ext_euclid(a, n)[1]
```

Figure 1: Функция для расширенного алгоритма Евклида и обратного значения

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)

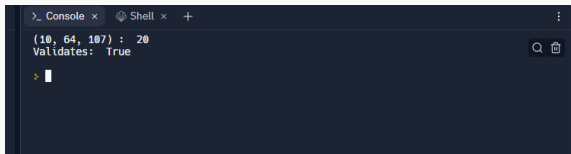
```
49 def pollard(G, H, P):
50     # P: prime
51     # G: generator
52     Q = int((P - 1) // 2) # sub group
53     x = G*H
54     a = 1
55     b = 1
56     X = x
57     A = a
58     B = b
59     for i in range(1, P):
60         x, a, b = xab(x, a, b, (G, H, P, Q))
61         X, A, B = xab(X, A, B, (G, H, P, Q))
62         X, A, B = xab(X, A, B, (G, H, P, Q))
63
64         if x == X:
65             break
66     nom = a-A
67     denom = B-b
68     # Необходимо вычислить обратное значение, чтобы правильно вычислить
69     # дробь по модулю Q.
70     res = (inverse(denom, Q) * nom) % Q
71     if verify(G, H, P, res):
72         return res
73     return res + Q
74
```

Figure 3: Функция для алгоритма pollard

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

```
75▼ def verify(g, h, p, x):  
76     """  
77     Verifies a given set of g, h, p and x  
78     :param g: Generator  
79     :param h:  
80     :param p: Prime  
81     :param x: Computed X  
82     :return:  
83     """  
84     return pow(g, x, p) == h  
85  
86▼ args = [  
87     (10, 64, 107),  
88 ]  
89  
90  
91▼ for arg in args:  
92     res = pollard(*arg)  
93     print(arg, ': ', res)  
94     print("Validates: ", verify(arg[0], arg[1], arg[2], res))  
95     print()  
96
```

Figure 4: Функция verify и блок работы программы



A screenshot of a terminal window with a dark background. The window has two tabs: 'Console' and 'Shell'. The 'Console' tab is active and displays the output of an algorithm. The output consists of two lines: '(10, 64, 107) : 20' and 'Validates: True'. Below the output, there is a prompt character '➤' followed by a white cursor. In the top right corner of the terminal window, there are icons for search and trash.

```
>_ Console x Shell x +  
(10, 64, 107) : 20  
Validates: True  
➤
```

Figure 5: Результат алгоритма

Реализовал реализующий p -метод Полларда для задач дискретного логарифмирования.

Спасибо за внимание