Отчёт по лабораторной работе №7

Дискретное логарифмирование

Шевляков Илья Николаевич НФИмд-01-21

Содержание

# Цель работы

Изучение задачи дискретного логарифмирования.

# Теоретические сведения

Пусть в некоторой конечной мультипликативной абелевой группе задано уравнение

Решение задачи дискретного логарифмирования состоит в нахождении некоторого целого неотрицательного числа , удовлетворяющего уравнению. Если оно разрешимо, у него должно быть хотя бы одно натуральное решение, не превышающее порядок группы. Это сразу даёт грубую оценку сложности алгоритма поиска решений сверху — алгоритм полного перебора нашёл бы решение за число шагов не выше порядка данной группы.

Чаще всего рассматривается случай, когда группа является циклической, порождённой элементом . В этом случае уравнение всегда имеет решение. В случае же произвольной группы вопрос о разрешимости задачи дискретного логарифмирования, то есть вопрос о существовании решений уравнения , требует отдельного рассмотрения.

## p-алгоритм Поллрада

* Вход. Простое число , число порядка по модулю , целое число б ; отображение , обладающее сжимающими свойствами и сохраняющее вычислимость логарифма.
* Выход. показатель , для которого , если такой показатель существует.

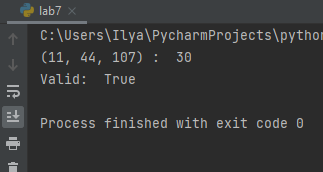
1. Выбрать произвольные целые числа и положить
2. Выполнять $c=f(c)(mod p), d=f(f(d))(mod p), вычисляя при этом логарифмы для и как линейные функции от по модулю , до получения равенства
3. Приняв логарифмы для и , вычислить логарифм решением сравнения по модулю . Результат или РЕШЕНИЯ НЕТ.

# Выполнение работы

## Реализация алгоритма на языке Python

def euclid(a, b):  
 if b == 0:  
 return a, 1, 0  
 else:  
 d, xx, yy = euclid(b, a % b)  
 x = yy  
 y = xx - (a // b) \* yy  
 return d, x, y  
  
  
def inver(a, n):  
 return euclid(a, n)[1]  
  
  
def pol\_ab(x, a, b, todochan):  
 (G, H, P, Q) = todochan  
 sub = x % 3  
 if sub == 0:  
 x = x \* todochan[0] % todochan[2]  
 a = (a + 1) % Q  
 if sub == 1:  
 x = x \* todochan[1] % todochan[2]  
 b = (b + 1) % todochan[2]  
 if sub == 2:  
 x = x \* x % todochan[2]  
 a = a \* 2 % todochan[3]  
 b = b \* 2 % todochan[3]  
 return x, a, b  
  
  
def pollrad(G, H, P):  
 Q = int((P - 1) // 2)  
 x = G \* H  
 a = 1  
 b = 1  
 X = x  
 A = a  
 B = b  
 for i in range(1, P):  
 x, a, b = pol\_ab(x, a, b, (G, H, P, Q))  
 X, A, B = pol\_ab(X, A, B, (G, H, P, Q))  
 X, A, B = pol\_ab(X, A, B, (G, H, P, Q))  
 if x == X:  
 break  
 nom = a - A  
 denom = B - b  
 res = (inver(denom, Q) \* nom) % Q  
 if ver(G, H, P, res):  
 return res  
  
  
def ver(g, h, p, x):  
 return pow(g, x, p) == h  
  
def lab7():  
 args = [  
 (11, 44, 107),  
 ]  
 for arg in args:  
 res = pollrad(\*arg)  
 print(arg, ': ', res)  
 print('Valid: ', ver(arg[0], arg[1], arg[2], res), end='\n')  
  
lab7()

## Контрольный пример



Работа алгоритма

# Выводы

Изучили задачу дискретного логарифмирования.

# Список литературы

1. [Дискретное логарифмирование](https://math.fandom.com/ru/wiki/%D0%94%D0%B8%D1%81%D0%BA%D1%80%D0%B5%D1%82%D0%BD%D0%BE%D0%B5_%D0%BB%D0%BE%D0%B3%D0%B0%D1%80%D0%B8%D1%84%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)
2. [Доступно о криптографии на эллиптических кривых](https://blog.qrator.net/ru/ec-tls-13_60/)