**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**
**«Московский Авиационный Институт»**
**(Национальный Исследовательский Университет)**

**Институт: №8 «Информационные технологии и прикладная математика»**
**Кафедра: 806 «Вычислительная математика и программирование»**

# Лабораторная работа № 5
по курсу «Криптография»

Группа: М8О-307Б-21

Студент: Ф. А. Меркулов

Преподаватель: А. В. Борисов

Оценка:

Дата: 15.05.2024

Москва, 2024

# ОГЛАВЛЕНИЕ


**1 Тема** ..... 3

**2 Задание** ..... 3

**3 Теория** ..... 4

**4 Ход лабораторной работы** ..... 6

**5 Выводы** ..... 10

**6 Список используемой литературы** ..... 11

# 1 Тема

Эллиптические кривые

# 2 Задание

Подобрать такую эллиптическую кривую, порядок точки которой полным перебором находится за 10 минут на ПК. Упомянуть в отчёте результаты замеров работы программы, характеристики вычислителя. Также указать какие алгоритмы и/или теоремы существуют для облегчения и ускорения решения задачи полного перебора. Рассмотреть для случая конечного простого поля $\mathbb{Z}_p$.

# 3 Теория

Эллиптические кривые играют ключевую роль в современной криптографии, обеспечивая высокий уровень безопасности при меньших ключах по сравнению с другими криптографическими системами. Одной из задач, связанных с эллиптическими кривыми, является определение порядка точки на кривой.

**Эллиптическая кривая** над полем $\mathbb{Z}_p$ задается уравнением вида:

$$y^2 = x^3 + ax + b \, mod \, p,$$

где $a, b \in \mathbb{Z}_p$ и дискриминант $4a^3 + 27b^2 \neq 0 \, mod \; p$.

**Точка на эллиптической кривой** $P$ – это пара чисел $(x, y) \in \mathbb{Z}_p \times \mathbb{Z}_p$, удовлетворяющая уравнению кривой, вместе с точкой на бесконечности $O$.

**Порядок точки** $P$ на эллиптической кривой – это наименьшее положительное число $n$ такое, что $nP = O$, где $O$ – точка на бесконечности.

## Методы и алгоритмы:

**Полный перебор** (brute force) – это метод исчерпывающего поиска, при котором проверяются все возможные варианты до нахождения решения. В контексте данной задачи полный перебор включает вычисление порядков точек на эллиптической кривой для всех возможных значений.

**Алгоритмы для ускорения вычислений:**

- **Алгоритм Бабая-Шэлвита**: Данный алгоритм предназначен для ускорения процесса нахождения порядка точки. Он основан на использовании кривых низкого порядка и модульных инвариантов.

- **Метод Шёнхаге-Штрассена**: Это быстрый алгоритм умножения больших чисел, который может использоваться для ускорения арифметических операций над точками эллиптической кривой.

- **Алгоритм Лагранжа**: Этот метод позволяет существенно сократить количество необходимых операций за счет использования свойств циклических групп и теории чисел.

- **Теорема Хассе**: Теорема Хассе ограничивает количество точек на эллиптической кривой в пределах $p + 1 - 2p \leq n \leq p + 1 + 2p$, что позволяет ограничить область поиска.

- **Использование методов Ленстры для факторизации**: В определенных случаях методы факторизации на эллиптических кривых могут использоваться для нахождения порядка точки путем разложения на простые множители.

# 4 Ход лабораторной работы

Сначала я взял a = 3, b = 7 (просто нравятся эти числа) и решил для себя что должен найти такой порядок поля p, чтобы выполнить задание и найти порядок точки полным перебором за примерно 10 минут.

Для нахождения порядка эллиптической кривой я нахожу количество целочисленных точек из множества $\mathbb{Z}_p$, принадлежащих заданной кривой. Эта операция выполняется за $O(p^2)$.

Затем я выбираю случайную точку и начинаю искать её порядок. Для этого я складываю точку саму с собой до тех пор, пока не получится точка (0, 0). Количество итераций, потребовавшихся на это, и есть порядок точки. Все вычисления проводятся по модулю p.

С помощью бинарного поиска и экспериментального способа был найден порядок поля p = 18757. В итоге получил такие выходные параметры программы:

```
100%|██████████| 18757/18757 [10:14<00:00, 30.27it/s]
Порядок кривой: 18987
Точка (765, 17351) порядок: 20435
Время: 614.9788991204622 s
```

Прогресс бар использовался как ориентир, если выполнилось меньше 25% работы и время переваливало за 4 минуты, то я искал новый параметр p, также если больше 50% и время было меньше 4 минут я тоже начинал искать новый параметр p.

Код программы:

```
import time
import random
from tqdm import tqdm  # Для того чтобы понимать сколько времени будет
работать программа (С помощью progress bar)


def point_belongs_to_an_elliptical_curve(x, y, p):
```

```
    """
    Проверяем, что точка (x, y) принадлежит эллиптической кривой y^2 = x^3
+ ax + b в поле Z_p
    """
    return (y ** 2) % p == (x ** 3 + (a % p) * x + (b % p)) % p


def extended_euclidean_algorithm(a, b):
    """
    Расширенный алгоритм Евклида

    Возвращает (gcd, x, y): ax + by == gcd(a, b)
    """
    s, old_s = 0, 1
    t, old_t = 1, 0
    r, old_r = b, a

    while r != 0:
        quotient = old_r // r
        old_r, r = r, old_r - quotient * r
        old_s, s = s, old_s - quotient * s
        old_t, t = t, old_t - quotient * t

    return old_r, old_s, old_t


def inverse_value(n, p):
    """
    Возвращает m: (n * m) % p == 1
    """
    gcd, x, y = extended_euclidean_algorithm(n, p)
    assert (n * x + p * y) % p == gcd

    if gcd != 1:
        raise ValueError(
            '{} has no multiplicative inverse '
            'modulo {}'.format(n, p))
    else:
        return x % p


def sum_of_points(A, B, p):
    """
    Вычисляем алгебраическую сумму точек A, B в поле Z_p
    Возвращает R = (x_r, y_r) = A + B
```

```
    """
    if A == (0, 0):
        return B
    if B == (0, 0):
        return A
    if A[0] == B[0] and A[1] != B[1]:
       return 0, 0

    if A != B:
        m = ((A[1] - B[1]) * inverse_value(A[0] - B[0], p)) % p
    else:
        m = ((3 * A[0] ** 2 + a) * inverse_value(2 * A[1], p)) % p

    x_r = (m ** 2 - A[0] - A[1]) % p
    y_r = (A[1] + m * (x_r - A[0])) % p
    return x_r, -y_r % p


def order_of_the_point(point, p):
    """
    Вычисляем порядок точки point в поле Z_p
    """
    ans = 0
    found_point_order = False
    prev_point = point
    while not found_point_order:
        ans += 1
        point_sum = sum_of_points(point, prev_point, p)
        if point_sum == (0, 0):
            found_point_order = True
        else:
            prev_point = point
            point = point_sum
    return ans


if __name__ == '__main__':
    a = 3
    b = 7
    p = 18757

    start = time.time()
    points = []
    for x in tqdm(range(p)):
        for y in range(p):
```

```
            if point_belongs_to_an_elliptical_curve(x, y, p):
                points.append((x, y))

    curve_order = len(points)
    print('Порядок кривой:', curve_order)

    point = random.choice(points)
    point_order = order_of_the_point(point, p)
    print('Точка', point, 'порядок:', point_order)

    end = time.time()
    print('Время:', end - start, 'с')
```

В данной работе использовалась вычислительная среда Google Colab с использованием CPU. Характеристики вычислителя, предоставляемого этой средой, включают:

- **Процессор (CPU)**: Intel Xeon с тактовой частотой 2.30 GHz
- **Количество ядер**: 2
- **Оперативная память (RAM)**: 12.6 GB
- **Операционная система**: Среда выполнения Google Colab, основанная на Ubuntu Linux
- **Дополнительное ПО**: Python 3.8, библиотеки NumPy, SciPy и другие необходимые для выполнения задач

# 5 Выводы

В результате работы были достигнуты все поставленные цели: найдена подходящая эллиптическая кривая и порядок точки на ней за примерно 10 минут полным перебором. Были предоставлены результаты замеров времени и характеристики вычислителя, а также рассмотрены методы и алгоритмы, которые могут помочь в ускорении решения подобных задач. Было интересно выполнять это лабораторную работу!

# 6 Список используемой литературы


1. https://habr.com/ru/articles/335906/
2. https://ru.wikipedia.org/wiki/Эллиптическая_кривая