**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-306Б-21

Студент: Н. И. Лохматов

Преподаватель: А. В. Борисов

Оценка:

Дата: 30.03.2024

Москва, 2024

**ОГЛАВЛЕНИЕ**

[1 Тема 3](#_Toc158983147)

[2 Задание 3](#_Toc158983148)

[3 Теория 4](#_Toc158983149)

[4 Ход лабораторной работы 5](#_Toc158983150)

[5 Выводы 6](#_Toc158983151)

# **Тема**

Подбор эллиптической кривой и поиск порядка её точки за заданное время.

# **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 Z\_p.

# **Теория**

Но для нас достаточно того, что эллиптическая кривая — это просто **множество точек, описываемое уравнением:**

, где

Порядок точки p на эллиптической кривой — это наименьшее положительное число n, такое что np=O, где O — "бесконечно удаленная" точка, служащая нейтральным элементом группы.

Для нахождения порядка точки методом полного перебора необходимо последовательно вычислять np для n = 1, 2, 3, …, пока не будет найдено n, для которого np = O. Этот метод чрезвычайно ресурсоемкий для больших значений p.

Ниже в блоке теории рассмотрю теоремы и алгоритмы для ускорения решения задачи полного перебора.

**Теоремы Хассе**

Для эллиптической кривой над полем Zp, порядок кривой (количество точек на кривой) ограничен теоремой Хассе: | N – (p + 1) | <= 2, где N – порядок кривой. Эта теорема позволяет сузить диапазон для поиска порядка точки.

**Алгоритм Шуфа**

Вычисляет порядок эллиптической кривой за полиномиальное время. Он не предназначен для нахождения порядка отдельной точки, но знание порядка кривой может помочь в этом. Алгоритм Шуфа основан на использовании полиномов деления для эллиптической кривой и применении их к вычислению порядка кривой в полиномиальное время относительно размера поля. Основная идея заключается в том, чтобы вычислить, как точки эллиптической кривой умножаются на малые простые числа, и использовать эти данные для сужения возможного порядка кривой.

**Алгоритм Полларда (Полига-Хеллмана)**

Алгоритм Полларда (иногда неверно называемый Поллига-Хеллманом) для ро-метода факторизации, а также его модификация для нахождения порядка элемента в группе, используют идею случайных прогулок для определения циклов и, соответственно, факторов порядка группы. Выбираются случайные точки и выполняются операции группы (например, сложение точек на эллиптической кривой), формируя "случайную прогулку" по элементам группы. Используется идея Флойда для обнаружения циклов в последовательности точек. Когда цикл найден, можно вычислить порядок (или фактор порядка) элемента.

**Алгоритм Бейбиджа-Шэнкса**

Алгоритм Бейбиджа-Шэнкса предназначен для нахождения логарифма в группе (в нашем контексте — порядка точки на эллиптической кривой), используя метод "встречи посередине". Этот алгоритм эффективен, когда размер группы известен и невелик. Проблема нахождения порядка разбивается на две меньшие задачи, которые решаются независимо, обычно через создание двух списков: один для "прямых" операций, другой — для "обратных". Ищется совпадение между значениями в двух списках, что позволяет вычислить искомый порядок (или логарифм) "по середине" изначальной задачи.

# **Ход лабораторной работы**

Характеристики ПК (ноутбука):

Процессор AMD Ryzen 5 5500U (6 ядер, 12 потоков, базовая частота 2.1 ГГц)

16 Гб оперативной памяти типа DDR4 (скорость 3200 МГц)

Работу я выполнял на языке Python. Основная идея алгоритма: выбор параметров кривой, генерация точек на ней и вычисление порядка случайной точки с последующим увеличением параметра p для поиска подходящей кривой. Программа запрашивает у пользователя параметры a, b и p для эллиптической кривой и время, в течение которого должен выполняться поиск. Затем, используя экземпляр класса EllipticCurve, программа в цикле ищет такую кривую, порядок точки которой можно вычислить в указанное время. Для этого программа увеличивает параметр p на фиксированное значение (3000) на каждой итерации, пытаясь найти подходящую кривую.

Класс EllipticCurve:

Инициализация: принимает коэффициенты a, b и p, проверяя условие несингулярности кривой.

Проверка принадлежности точки кривой: метод is\_elliptic\_curve используется для проверки, удовлетворяет ли точка уравнению эллиптической кривой.

Вычисление обратного элемента: метод inverse\_of вычисляет обратный элемент для заданного числа в поле по модулю p, используя расширенный алгоритм Евклида.

Сложение точек на кривой: метод add\_points реализует операцию сложения двух точек на эллиптической кривой.

Вычисление порядка точки: метод order\_point находит порядок заданной точки путём повторного сложения точки с самой собой до тех пор, пока не будет достигнута нейтральная точка.

Шаг итерации: метод step выполняет один шаг итерации, включая генерацию точек на кривой и вычисление порядка случайно выбранной точки.

Проверка на простоту и поиск следующего простого числа: методы is\_prime\_number и get\_next\_prime\_number используются для проверки чисел на простоту и поиска следующего простого числа, начиная с заданного значения.

Но из-за того, что я использовал Python, программа работала сильно дольше заданного времени. Вот ряд тестов:

Входные данные:

a: 34567

b: 22887

p: 661

Далее я вводил время 2, 4, 8, 32, 64, 128 и 256

# 2s => 8.49s | 1 iter

# 4s => 8.40s | 1 iter

# 8s => 8.61s | 1 iter

# 32s => 8.29s | 1 iter | 60.86s | 2 iter

# 64s => 8.22s | 1 iter | 57.68 | 2 iter | 218.48s | 3 iter

# 128s => 8.28s | 1 iter | 59.55s | 2 iter | 216.15s | 3 iter

# 256s => 8.21s | 1 iter | 57.80s | 2 iter | 215.64s | 3iter | 587.70s | 4 iter

При вводе 256, программа работала 587 секунд, что примерно равно 10 минутам.

Результат:

# y^2 = x^3 + 3878\*x + 22887 % 30689

# Curve order: 30605

# Point (6284, 17343) order: 186148

# Time: 587.7021234035492 seconds

Найденная кривая:

# **Выводы**

В ходе выполнения лабораторной работы я подобрал такую эллиптическую кривую, порядок точки которой полным перебором находится за 587 секунд. Также я описал теорему Хассе и ряд алгоритмов для оптимизации решения этой задачи.

# **Список используемой литературы**

Доступно о криптографии на эллиптических кривых - https://habr.com/ru/articles/335906/