### Разложение чисел на множители

Миличевич Александра

15 Февраля, 2025, Москва, Россия

Российский Университет Дружбы Народов

# <u>Цели и задачи</u>

## Цель лабораторной работы

Ознакомление с р-методом Полларда

Выполнение лабораторной

работы

## Задача разложения на простые множители

Разложение на множители — предмет непрерывного исследования в прошлом; и такие же исследования, вероятно, продолжатся в будущем. Разложение на множители играет очень важную роль в безопасности некоторых криптосистем с открытым ключом.

### р-алгоритм Поллрада

• Функция `pollard\_rho\_function(x, n)

#### Как работает:

- 1. \*\*Обновление значений:\*\* Вычисляются новые значения `a\_val` и `b\_val` путем итеративного применения функции `pollard\_rho\_function`. Значение `b\_val` обновляется дважды за итерацию.
- 2. \*\*Вычисление НОД:\*\* Вычисляется наибольший общий делитель (НОД) между разностью `a\_val` и `b\_val` и исходным числом `number` с помощью функции `gcd` из модуля `math`.

### р-алгоритм Поллрада

- 3. \*\*Проверка делителя:\*\*
- \* Если 'divisor' находится между 1 и 'number' (1 < divisor < number), значит, найден нетривиальный делитель. Функция выводит этот делитель и завершает выполнение программы.
- \* Если 'divisor' равен 'number', то это означает неудачу, и функция выводит сообщение об этом.
- \* Если 'divisor' равен 1, это означает, что на текущей итерации делитель не найден, и функция продолжает свою работу рекурсивно.

#### Оценка сложности

Сложность. Заметим, что этот метод требует сделать B-1операций возведения в степень  $a = a^e mod n$ . Есть быстрый алгоритм возведения в степень, который выполняет это за  $2 * 10 g_2 B$  операций. Метод также использует вычисления НОД, который требует  $n^3$ операций. Мы можем сказать, что сложность — так или иначе больше, чем O(B) или  $O(2^n)$ , где  $n_b$  — число битов в В. Другая проблема – этот алгоритм может заканчиваться сигналом об ошибке. Вероятность успеха очень мала, если В имеет значение, не очень близкое к величине

#### Оценка сложности

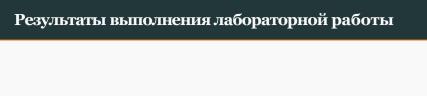
- 4. \*\*Использование глобальной переменной: \*\*
- \* Перед рекурсивным вызовом, текущее значение `b\_val` сохраняется в глобальной переменной `global\_b`, что позволяет отслеживать значение `b\_val` между рекурсивными вызовами.
- \* `global\_b` объявляется как глобальная переменная внутри функции с помощью ключевого слова `global`.

## Пример работы алгоритма

```
from math import gcd
# Глабальные переменные для апслеживания значений
global a - 1
global b = 1
def pollard rho function(x, n):
    Функция f(x) = (x^2 + 5) \% п, используемая в р-методе Полларда.
    Ares:
       х (int): Входное число.
       n (int): Mogynb.
    Returns:
        int: Pasynerar (x^2 + 5) % n.
    return (x*x+5)%n
def pollard rho recursive(number, a val. b val. divisor);
    Рекурсивная функция для реализации р-метода Полларда.
        number (int): Число, для которого ищется нетривиальный делитель.
       a val (int): Текущее значение 'a'.
       b val (int): Текущее значение 'b'.
        divisor (int): Текуций наибольший общий делитель.
    # Вычисляем следующее значение 'a'
    a val = pollard rho function(a val, number) % number
    # Вычисляем следующее эначение 'b' (дважды применяем функцию)
    b val = pollard rho function(pollard rho function(b val. number), number) % number
    # Вычисляем новый наибольший общий делитель
    divisor - gcd(a val - b val, number)
    # Если 1 < divisor < number, ны нашли нетривиальный делитель, выводим его и завершаем работу.
    if 1 < divisor < number:
       print(divisor)
    # Ecnu divisor paвен number, это означает неудачу, и ны ничего не выводит
    if divisor -- number:
        print("Делитель не найден")
    # Если divisor рабен 1, продолжаем рекурсивно с новыми значениями
        global global b # Используем глобольную переменную для сохранения значения b_val.
       global b = b_val # Сохраняем текущее значение b val в глобальную переменную.
        pollard rho recursive(number, a val, b val, divisor)
```

#### **Рис. 1:** Pollard

# Выводы



Изучили задачу разложения на множители и р-алгоритм Поллрада.