Лабораторная работа №5

Математические основы защиты информации и кибербезопасности

Тема: Вероятностные алгоритмы проверки чисел на простоту

Автор: Лобов Михаил Сергеевич

Цель работы

Изучить вероятностные алгоритмы проверки чисел на простоту.

Задание

Реализовать алгоритмы проверки чисел на простоту на языке Julia.

Теоретическое введение

- Целое число (p \in Z / { 0 }) называется **простым**, если оно не имеет делителей, кроме тривиальных.
- Числа (\pm2, \pm3, \pm5, \pm7, \pm11), и др. являются простыми.

Простые числа важны для криптографии и многих других областей.

Сравнение по модулю

Пусть (m \in N, m > 1).

Целые числа (a) и (b) называются **сравнимыми по модулю** (m):

$$a \equiv b \pmod{m}$$

Если разность (a - b) делится на (m).

Типы алгоритмов проверки простоты

- **Детерминированные**: всегда дают точный ответ.
- **Вероятностные**: используют случайные числа и дают вероятностный ответ, часто быстрее.

Тест Ферма

Тест Ферма основан на малой теореме Ферма:

$$a^{p-1} \equiv 1 \pmod{p}$$

для простого (p).

Если для нечетного (n) существует (a) такое, что:

$$a^{n-1} \not\equiv 1 \pmod{n}$$

то (n) составное.

Код: Тест Ферма (Julia)

```
using Random
n = 20
function is_prime(n::Int, k::Int=5)
    if n < 5 || n % 2 == 0
        return false
    end
    for _ in 1:k
        a = rand(2:n-2)
        r = powermod(a, n-1, n)
        if r != 1
            return false
        end
    end
    return true
end
```

Вычисление символа Якоби

Необходим для теста Соловея-Штрассена.

Символ Якоби (\left(\frac{a}{n} \right)) определяет взаимную простоту чисел.

Код: Вычисление символа Якоби (Julia)

```julia
function jacobi_symbol(a::Int, n::Int)
    g = 1
    while a != 0
        k = 0
        while a % 2 == 0
            a ÷= 2
            k += 1
        end
        a1 = a
        s = if k % 2 == 0 1 else (n % 8 == 1 || n % 8 == 7 ? 1 : -1) end
        g *= s
        a, n = n % a1, a1
    end
    return g
end
```

Тест Соловея-Штрассена

- Использует критерий Эйлера.
- Число (n) является простым, если:

$$a^{\frac{n-1}{2}} \equiv \left(\frac{a}{n}\right) \pmod{n}$$

Код: Тест Соловея-Штрассена (Julia)

```julia
function test_solovei_strassen(n::Int)
    a = rand(2:n-2)
    n_1 = (n-1)/2
    r = powermod(a, n_1, n)
    s = jacobi_symbol(a, n)
    return r == s % n
end
```

Тест Миллера-Рабина

Популярный вероятностный алгоритм, используется для проверки больших чисел на простоту.

Алгоритм Миллера-Рабина

1. Представить (n - 1) как (2^s r), где (r) нечетное.
2. Выбрать случайное (a).
3. Выполнить проверки, чтобы определить, является ли (n) вероятно простым.

Код: Тест Миллера-Рабина (Julia)

```julia
function test_miller_rabin(n::Int, k::Int=5)
    s, r = 0, n - 1
    while r % 2 == 0
        r ÷= 2
        s += 1
    end
    for _ in 1:k
        a = rand(2:n-2)
        y = powermod(a, r, n)
        for j in 1:s-1
            y = powermod(y, 2, n)
            if y == n - 1 break end
        end
        if y != n - 1 return false end
    end
    return true
end
```

Выводы

- Изучены вероятностные алгоритмы проверки простоты: тест Ферма, тест Соловея-Штрассена и тест Миллера-Рабина.
- Эти алгоритмы важны в криптографии для быстрого определения простоты чисел.
- Алгоритм Миллера-Рабина является наиболее популярным для больших чисел.

Спасибо за внимание!