Отчёт по лабораторной работе №5

Дисциплина: Математические основы защиты информации и информационной безопасности

Дэнэилэ Александр Дмитриевич, НПМмд-02-23

Содержание

Список таблиц

Список иллюстраций

1 Цель работы

Изучить вероятностные алгоритмы проверки чисел на простоту.

2 Задание

Реализовать четыре теста на определение простоты чисел:

1. Тест Ферма
2. Символ Якоби
3. Тест Соловэя-Штрассена
4. Тест Миллера-Рабина

3 Теоретическое введение

Пусть a - целое число. Числа $\pm 1, \pm a$ называются тривиальными делителями числа a.

Целое число $p \in \mathbb{Z}/\{0\}$ называется простым, если оно не является делителем единицы и не имеет других делителей, кроме тривиальных. В противном случае число $p \in \mathbb{Z}/\{-1, 0, 1\}$ называется составным. Например, числа $\pm 2, \pm 3, \pm 5, \pm 7, \pm 11, \pm 13, \pm 17, \pm 19, \pm 23, \pm 29$ являются простыми.

Пусть $n \in \mathbb{N}, m > 1$. Целые числа a и внызываются сравнимыми по модулю m (обозначается $a \equiv b \ (mod \, m)$) если разность $a - b$ делится на m. Также эта процедура называется нахождением остатка от целочисленного деления a на b.

Проверка чисел на простоту является составной частью алгоритмов генерации простых чисел, применяемых в криптографии с открытым ключом. Алгоритмы проверки на простоту можно разделить на вероятностные и детерминированные.

Детерминированный алгоритм всегда действует по одной и той же схеме и гарантированно решает поставленную задачу (или не дает никакого ответа). *Вероятностный* алгоритм использует генератор случайных чисел и гарантированно точный ответ. Вероятностные алгоритмы в общем случае не менее эффективны, чем детерминированные (если используемый генератор случайных чисел всегда дает набор одних и тех же чисел, зависящих от входных данных, то вероятностный алгоритм становится детерминированным).

Для проверки на простоту числа n вероятностным алгоритмом выбирают случайной число a $(1 < a < n)$ и проверяют условия алгоритма. Если число n не проходит тест по основанию a, то алгоритм выдает результат «Число n составное», и число n действительно является составным.

Если же n проходит тест по основанию a, ничего нельзя сказать о том, действительно ли число n является простым. Последовательно проведя ряд проверок таким тестом для разных a и получив для каждого из них ответ «Число n, вероятно, простое», можно утверждать, что число n является простым с вероятностью, близкой к 1. После с независимых выполнений теста вероятность того, что составное число и будет t раз объявлено простым (вероятность ошибки), не превосходит $\frac{1}{2^t}$.

Тест Ферма основан на малой теореме Ферма: для простого числа p и произвольного числа a, $1 \leq a \leq p-1$, выполняется сравнение

$$a^{p-1} \equiv 1 \ (mod\, p)$$

Следовательно, если для нечетного n существует такое целое a, что $1 \leq a < n$, НОД(a, n) = 1 и $a^{n-1} \neq 1 \ (mod\, n)$, то число n составное.

4 Выполнение лабораторной работы

1. Реализуем тест Ферма (рис. 4.1).

Рис. 4.1: Тест Ферма

2. Реализуем алгоритм нахождения символа Якоби (рис. 4.2).

Рис. 4.2: Символ Якоби

3. Реализуем тест Соловэя-Штрассена (рис. 4.3).

```python
n = int(input()) #n >= 5
a = np.random.randint(2, n-3)
r = a ** ((n - 1)/2) % n
if r != 1 and r != (n-1):
    print("Составное")
else:
    s = jac(n, a)
    if r % n == s:
        print("Составное")
    else:
        print("Вероятно, простое")
```

Рис. 4.3: Тест Соловэя-Штрассена

4. Реализуем тест Миллера-Рабина (рис. 4.4).

```python
n = int(input()) #n >= 5 нечет
s, r = odd(n - 1)
a = np.random.randint(2, n-2)
y = a ** r % n
flag = False
if y != 1 and y != (n - 1):
    j = 1
    while j <= (s - 1) and y != (n - 1):
        y = y ** 2 % n
        if y == 1:
            flag = True
        j += 1
    if y != (n - 1):
        flag = True
if flag:
    print("Составное")
else:
    print("Вероятно, простое")
```

Рис. 4.4: Тест Миллера-Рабина

5 Выводы

Изучил вероятностные алгоритмы проверки чисел на простоту.

Список литературы