Отчет по лабораторной работе №5

Дисциплина: Математические основы защиты информации и информационной безопасности

Живцова Анна

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Изучить вероятностные алгоритмы проверки чисел на простоту.

2 Задание

Реализовать алгоритмы тестов Ферма, Соловэя-Штрассена, Миллера-Рабина.

3 Теоретическое введение

Простые числа широко применяются в криптографии с открытым ключем. Подробнее в источниках [1,2]. Алгоритмы проверки на простоту можно разделить на детерминированные и вероятностные. Вероятностные, возможно, вычислительно менее сложные, однако они позволяют дать ответ лишь с некоторой вероятностью. Иногда этого бывает достаточно. Рассмотрим несколько вероятностных тстов на простоту и приведем ключевые факты, на которых основаны тесты. Дале p – простое число, и $a \in [2, p-1]$ – произвольное число.

Тест Ферма. основан на следующем факте

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

Тест Соловэя-Штрассена. основан на критерии Эйлера

>

$$a^{\frac{p-1}{2}} \equiv \left(\frac{a}{p}\right) (mod\ p) \forall a.$$

Тест Миллера-Рабина. основан на тесте Ферма.

4 Выполнение лабораторной работы

В этом разделе n – число, которое требуется проверить на простоту. n нечетно и больше 5.

4.1 Тест Ферма

Для реализации теста Ферма на языке Python была написанна следующая функция.

```
def ferma(n):
    a = np.random.randint(2, n-2)
    if (a**(n-1))%n == 1:
        return 'Число {}, вероятно, простое'.format(n)
    else:
        return 'Число {} составное'.format(n)
```

4.2 Тест Соловэя-Штрассена

Для реализации теста отдельно была реализована функция рассчета символя Якоби

```
def yakoby(n, a, g = 1):
    if a == 0:
        return 0
```

```
    if a == 1:
        return g
    k = 0
    a1 = a
    while a1%2 == 0:
        k += 1
        a1 /=2
    if k%2 == 0:
        s = 1
    elif n%8 == 1 or n%8 == 7:
        s = 1
    elif n%8 == 3 or n%8 == 5:
        s = -1
    if a == 1:
        return g*s
    if n%4 == 3 and a1%4 == 3:
        s *= -1
    a = n%a1
    n = a1
    g *= s
    return yakoby(n, a, g)
```

Сам тест реализован с помощью функции

```
def s_sh(n):
    a = np.random.randint(2, n-2)
    r = a**((n-1)/2)
    if r != 1 and r!= n-1:
        return 'Число {} составное'.format(n)
    else:
        s = yakoby(n, a)
```

```
        if r%n == s:
            return 'Число {} составное'.format(n)
        return 'Число {}, вероятно, простое'.format(n)
```

4.3 Тест Миллера-Рабина

Данный тест реализован с помощью функции

```
def m_r(n):
    s = 0
    r = n-1
    while r%2 == 0:
        s += 1
        r /=2
    a = np.random.randint(2, n-2)
    y = (a**r)%n
    if y != 1 and y != n - 1:
        for j in range(1, s):
            if y != n - 1 and (y*y)%n == 1:
                return 'Число {} составное'.format(n)
        if y != n - 1:
            return 'Число {} составное'.format(n)
    return 'Число {}, вероятно, простое'.format(n)
```

4.4 Тестирование реализованных алгоритмов

Проведено тестирование реализованных алгоритмов. Видно, что в большинстве случаев ответы различных тестов совпадают, однако иногда делается ошибочное предположение о простоте (см. рис. 4.1).

```python
for i in range(10):
    n = np.random.randint(4, 100)*2 + 1
    print('Ферма ', ferma(n))
    print('Соловэй-Штрассен ', s_sh(n))
    print('Миллер-Рабин ', m_r(n))
    print()
```

```
Ферма  Число 77 составное
Соловэй-Штрассен  Число 77 составное
Миллер-Рабин  Число 77 составное

Ферма  Число 105 составное
Соловэй-Штрассен  Число 105 составное
Миллер-Рабин  Число 105 составное

Ферма  Число 15 составное
Соловэй-Штрассен  Число 15 составное
Миллер-Рабин  Число 15 составное

Ферма  Число 147 составное
Соловэй-Штрассен  Число 147 составное
Миллер-Рабин  Число 147 составное

Ферма  Число 141 составное
Соловэй-Штрассен  Число 141 составное
Миллер-Рабин  Число 141 составное

Ферма  Число 89, вероятно, простое
Соловэй-Штрассен  Число 89 составное
Миллер-Рабин  Число 89 составное

Ферма  Число 139, вероятно, простое
Соловэй-Штрассен  Число 139 составное
Миллер-Рабин  Число 139 составное

Ферма  Число 183 составное
Соловэй-Штрассен  Число 183 составное
Миллер-Рабин  Число 183 составное

Ферма  Число 27 составное
Соловэй-Штрассен  Число 27 составное
Миллер-Рабин  Число 27 составное

Ферма  Число 199, вероятно, простое
Соловэй-Штрассен  Число 199 составное
Миллер-Рабин  Число 199 составное
```

Рис. 4.1: Тестирование вероятностных алгоритмов проверки на простоту

5 Выводы

В данной работе я изучила вероятностные алгоритмы проверки чисел на простоту. Реализовала тесты Ферма, Соловэя-Штрассена, Миллера-Рабина. Протестировала реализованные функции.

Таким образом, задачи работы были выполнены, а цели достигнуты.

Список литературы

1. Kulyabov D., Korolkova A., Gevorkyan M. Информационная безопасность компьютерных сетей: лабораторные работы. 2015.
2. Самуйлов К.Е. и др. Сети и телекоммуникации : Учебник и практикум. Издательство Юрайт, 2019. С. 1–363.