МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»

Кафедра теоретических основ компьютерной безопасности и криптографии

Проверка чисел на простоту

ОТЧЁТ

ПО ДИСЦИПЛИНЕ

«ТЕОРЕТИКО-ЧИСЛОВЫЕ МЕТОДЫ В КРИПТОГРАФИИ»

студента 5 курса 531 группы

специальности 10.05.01 Компьютерная безопасность

факультета компьютерных наук и информационных технологий

Мызникова Сергея Анатольевича

Преподаватель, профессор ___________ В.А. Молчанов

подпись, дата

Саратов 2024

СОДЕРЖАНИЕ

1 Постановка задачи

Цель работы — изучение основных методов проверки простоты чисел и их программная реализация.

Порядок выполнения работы:

1. Рассмотреть тест Ферма проверки чисел на простоту и привести его программную реализацию.
2. Рассмотреть тест Соловея-Штрассе на проверки чисел на простоту и привести его программную реализацию.
3. Рассмотреть тест Миллера-Рабина и привести его программную реализацию.

2 Теоретические сведения

2.1 Тест Ферма

Плотность распределения простых чисел:

1 − 10 - 40%, 1 − 100 - 25%, 1 − 1000 - 17%, 1 – 10^5 - <10%.

Вероятностный алгоритм проверки числа n на простоту использует необходимые условия простоты P(a):

1) выбирается случайным образом $1 < a < n$ и проверяется выполнимость теста $P(a)$ – некоторого условного алгоритма,
2) если тест не проходит, то есть $P(a)$ не выполняется, то вывод “число n, составное”
3) если тест проходит, то есть выполняется $P(a)$, то вывод “число n, вероятно, простое”

Если событие A – “число n простое” имеет вероятность $P(A) > \frac{1}{2}$, то вероятность ошибки – получить для составного числа n вывод “число n возможно простое” $P(A) < \frac{1}{2}$ и при t повторах теста вероятность ошибки $P(A^t) < \frac{1}{2^t} \approx 0$.

Малая теорема Ферма. Если p – простое число и a – произвольное целое число, такое что $1 \leq a \leq p-1$, выполняется сравнение: $a^{p-1} \equiv 1 \ (mod\ p)$

Определение. Число n называется псевдопростым по основанию a, если для чисел a и n выполняется сравнение $a^{n-1} \equiv 1 \ (mod\ n)$.

Лемма. Пусть N нечетное число. Тогда выполняются утверждения:

а) множество всех $a \in Z_N$, относительно которых N является псевдопростым, образует подгруппу в Z_N;

б) если N не является псевдопростым хотя бы по одному основанию a, то N не является псевдопростым относительно по крайней мере половины чисел из Z_n.

Вероятность успеха – вероятность получить "Число n составное" для составного числа n равна $P_0 = 1 - \frac{|F_n^+|}{n-1}$.

Возможны 3 случая:

1) Число n простое и тест всегда дает ответ "Число n, вероятно, простое"

2) Число n составное и $F_n^+ \neq Z_n^*$, тогда тест дает ответ "Число n составное" с вероятность успеха $P_0 = 1 - \frac{|F_n^+|}{n-1} \geq 1 - \frac{|F_n^+|}{|Z_n^*|} \geq 1 - \frac{1}{2} = \frac{1}{2}$

3) Число n составное и $F_n^+ = Z_n^*$, тогда тест дает ответ "Число n составное" с вероятностью $P_0 = 1 - \frac{\varphi(n)}{n-1}$.

В случае 2) при k повторах теста вероятность успеха $P_0^{(k)} = 1 - (1 - P_0)^k \geq 1 - \frac{1}{2^k} \approx 1$

Определение. Нечетное составное число n называется числом Кармайкла, если $F_n^+ = Z_n^*$(и, значит, вероятность успеха Алгоритма Ферма будет $P_0 = 1 - \frac{\varphi(n)}{n-1}$)

Лемма. Для любого числа Кармайкла n справедливы утверждения:

1) $n = p_1 p_2 \dots p_k$ для $k \geq 3$ простых различных чисел $p_1 p_2 \dots p_k$

2) $(\forall p - \text{простое})\ p|n \ \rightarrow p - 1|n - 1$

Плотность распределения чисел Кармайкла:

$1 - 10^5$ - 16 чисел: 561, 1105, 1729, …;

$1 - 2{,}5 * 10^{10}$ - 2163 чисел.

2.2 Тест Соловея-Штрассена

Критерий Эйлера. Нечетное число n является простым тогда и только тогда, когда для любого а выполняется свойство:

$$E_n(a) = (a^{\frac{n-1}{2}} \equiv \left(\frac{a}{n}\right) (mod\, n)).$$

n −простое число тогда и только тогда, когда $E_n^+ = Z_n^*$, где $E_n^+ = \{a \in \{1, 2, \dots, n-1\} |\ E_n(a)\}$.

Определение: Число n называется эйлеровым псевдопростым по основанию a, если для чисел a, n выполняется сравнение $a^{\frac{n-1}{2}} \equiv \left(\frac{a}{n}\right) (mod\, n)$.

Лемма. Пусть n нечетное число. Тогда выполняются утверждения:

а) множество всех $a \in Z_n$, относительно которых n является эйлеровым псевдопростым, образует подгруппу в Z_n;

б) если n – составное число, то n не является псевдопростым эйлеровым числом относительно по крайней мере половины чисел из Z_n.

Вероятность успеха теста Соловея-Штрассена для составного числа n равна $P_0 = 1 - \frac{|E_n^+|}{n-1} \geq \frac{1}{2}$

Возможны 2 случая:

1) Число n простое и тест всегда дает ответ "Число n, вероятно, простое"
2) Число n составное и тест дает ответ "Число n составное" с вероятность успеха $P_0 \geq \frac{1}{2}$

   В случае 2) при k повторах теста вероятность успеха $P_0^{(k)} = 1 - (1 - P_0)^k \geq 1 - \frac{1}{2^k} \approx 1$

2.3 Тест Миллера-Рабина

Критерий Миллера. Пусть n нечётное и $n - 1 = 2^s t$ для нечетного t. Тогда n является простым тогда и только тогда, когда для любого a $a \in Z_n^*$ выполняется свойство:

$$M_n(a) = \{a^t \equiv 1\ (mod\, n) \vee (\exists\ 0 \leq k < s)\ (2^{2^k t} \equiv 1\ (mod\, n))).$$

n −простое число тогда и только тогда, когда $M_n^+ = Z_n^*$, где $M_n^+ = \{a \in \{1, 2, \dots, n-1\} |\ M_n(a)\}$.

Определение. Число N псвдопростое по основанию a называется сильно псевдопростым по основанию a, если выполняется одно из условий:

1) Либо $a^t = 1\ (mod\ n)$

2) Либо $a^{2^k t} = -1\ (mod\ n)$ для некоторого $0 \leq k < s$

Для составного числа n выполняется $|M_n^+| \leq \frac{|Z_n^*|}{4}$ и, значит, вероятность успеха теста Миллера-Рабина для составного числа n равна $P_0 = 1 - \frac{|M_n^+|}{n-1} \geq \frac{3}{4}$. При k повторах теста вероятность успеха $P_0^{(k)} = 1 - (1 - P_0)^k \geq 1 - \frac{1}{4^k} \approx 1$

3 Результаты работы

3.1 Алгоритмы: описание и временная сложность

3.1.1 Тест Ферма

Вход: Нечетное число n, количество итераций k

Выход: “Число n, вероятно, простое” или “Число n составное”

Шаг 1: Выбрать случайно $a\ \in \{1, 2, \dots, n-1\}$ и вычислить $d = \text{НОД}(a, n)$

Если $d > 1$, то ответ “Число составное”;

Шаг 2: Если d = 1, то проверить условие: $F_n(a) = (a^{n-1} \equiv 1 (mod\ n))$. Если оно не выполнено, то ответ “Число составное”, иначе “Число n, вероятно, простое”.

Временная сложность алгоритма. O($\log^3$n)

3.1.2 Тест Соловея-Штрассена

Вход: Нечетное число n, количество итераций k

Выход: “Число n, вероятно, простое” или “Число n составное”

Шаг 1: Выбрать случайно $a \in \{1, 2, \dots, n-1\}$ и вычислить $d = \text{НОД}(a, n)$

Если $d > 1$, то ответ “Число составное”;

Шаг 2: Если d = 1, то проверить условие: $E_n(a) = (a^{\frac{n-1}{2}} \equiv \left(\frac{a}{n}\right)(mod\ n))$. Если оно не выполнено, то ответ “Число составное”, иначе “Число n, вероятно, простое”.

Временная сложность алгоритма. O($\log^3$n)

3.1.3 Тест Миллера-Рабина

Вход: Нечетное число n, количество итераций k

Выход: “Число n, вероятно, простое” или “Число n составное”

Шаг 1: Выбрать случайно $a \in \{1, 2, \dots, n-1\}$ и вычислить $d = \text{НОД}(a, n)$

Если $d > 1$, то ответ “Число составное”;

Шаг 2: Если d = 1, то вычислить $r_k = a^{2^k t}$ для значений $k \in \{0, 1, 2, \dots, s-1\}$. Если $r_0 \equiv 1(mod\ n)$ или $r_k \equiv -1(mod\ n)$ для некоторого $0 \le k < s$, то ответ “Число n, вероятно, простое”. В противном случае ответ “Число n составное”.

Временная сложность алгоритма. O($\log^3$n)

3.2 Тестирование алгоритмов

Тест Ферма

```
Выберите тест на простоту числа:
1. Тест Ферма
2. Тест Миллера-Рабина
3. Тест Соловея-Штрассена
Введите номер теста: 1
Введите число, которое нужно проверить на простоту: 991
Введите количество итераций: 10
Результат: число 991, вероятно, простое.
```

Рисунок 1. Тестирование проверки простоты с помощью теста Ферма

Тест Соловея-Штрассена

```
1. Тест Ферма
2. Тест Миллера-Рабина
3. Тест Соловея-Штрассена
Введите номер теста: 3
Введите число, которое нужно проверить на простоту: 891
Введите количество итераций: 100
Результат: число 891 составное.
```

Рисунок 2. Тестирование проверки простоты с помощью теста Соловея-Штрассена

Тест Миллера-Рабина

```
Выберите тест на простоту числа:
1. Тест Ферма
2. Тест Миллера-Рабина
3. Тест Соловея-Штрассена
Введите номер теста: 2
Введите число, которое нужно проверить на простоту: 9771
Введите количество итераций: 100
Результат: число 9771 составное.
```

Рисунок 3. Тестирование проверки простоты с помощью теста Миллера-Рабина

ЗАКЛЮЧЕНИЕ

В ходе работы были рассмотрены алгоритмы проверки чисел на простоту. В качестве тестов были рассмотрены тест Ферма, тест Миллера-Рабина, тест Соловея-Штрассена. Все эти тесты являются вероятностными тестами проверки на простоту, поэтому число, проверенное с помощью этих тестов, можно считать лишь вероятно простым.

Для проверки работоспособности этих алгоритмов была реализована программа на языке Python. Программы были протестированы и была подтверждена правильность их работы. Также была произведена оценка временной сложности алгоритмов.

ПРИЛОЖЕНИЕ А

Программа с реализованными алгоритмами

```
import random

# Тест Ферма

def fermat_primality_test(n, k=5):

    if n <= 1:

        return False

    if n == 2:

        return True

    for _ in range(k):

        a = random.randint(2, n - 2)

        if pow(a, n - 1, n) != 1:

            return False

    return True

# Тест Миллера-Рабина

def miller_rabin(n, k):

    if n == 2 or n == 3:

        return True

    if n < 2 or n % 2 == 0:

        return False

    r, d = 0, n - 1

    while d % 2 == 0:

        r += 1

        d //= 2

    for _ in range(k):

        a = random.randint(2, n - 2)

        x = pow(a, d, n)
```

```
        if x == 1 or x == n - 1:
            continue

        for _ in range(r - 1):
            x = pow(x, 2, n)
            if x == n - 1:
                break
        else:
            return False
    return True


def jacobi_symbol(a, n):
    if n <= 0 or n % 2 == 0:
        return 0
    result = 1
    while a != 0:
        while a % 2 == 0:
            a //= 2
            if n % 8 in [3, 5]:
                result = -result
        a, n = n, a
        if a % 4 == 3 and n % 4 == 3:
            result = -result
        a %= n
    return result if n == 1 else 0
# Тест Соловея-Штрассена
def solovay_strassen_test(n, k=5):
```

```
    if n <= 2:
        return n == 2
    if n % 2 == 0:
        return False

    for _ in range(k):
        a = random.randint(2, n - 1)
        jacobi = jacobi_symbol(a, n)
        mod_exp = pow(a, (n - 1) // 2, n)

        if jacobi == 0 or mod_exp != (jacobi % n):
            return False
    return True

def primality_test_interface():
    print("Выберите тест на простоту числа:")
    print("1. Тест Ферма")
    print("2. Тест Миллера-Рабина")
    print("3. Тест Соловея-Штрассена")

    choice = int(input("Введите номер теста: "))

    n = int(input("Введите число, которое нужно проверить на простоту: "))
    k = int(input("Введите количество итераций: "))

    if choice == 1:
        result = fermat_primality_test(n, k)
```

```
        if result:
            print(f"Результат: число {n}, вероятно, простое.")
        else:
            print(f"Результат: число {n} составное.")

    elif choice == 2:
        result = miller_rabin(n, k)
        if result:
            print(f"Результат: число {n}, вероятно, простое.")
        else:
            print(f"Результат: число {n} составное.")

    elif choice == 3:
        result = solovay_strassen_test(n, k)
        if result:
            print(f"Результат: число {n}, вероятно, простое.")
        else:
            print(f"Результат: число {n} составное.")

    else:
        print("Неверный выбор.")

primality_test_interface()
```