Miller-Rabin 素性检测算法

班级：网安一班

姓名：密语

学号：202200460104

2024 年 5 月 5 日

目录

1 Miller-Rabin 素性检测算法的实现

1.1 算法原理

Miller-Rabin 素性检测算法是一种基于概率的素性检测算法，其基本原理是：对于一个奇数 n，若 n 是素数，则对于任意整数 a，有：

$$a^{n-1} \equiv 1 \pmod{n} \tag{1}$$

若 n 是合数，则对于大多数整数 a，有：

$$a^{n-1} \not\equiv 1 \pmod{n} \tag{2}$$

一些合数也满足 $a^{n-1} \equiv 1 \pmod{n}$，且 a 即使遍历 $[2, n-1]$，也无法筛去，这些合数被称为 Carmichael 数。
因此，我们考虑引入新的定理来提高检测的准确性。
二次探测定理：对于素数 p，若 $x^2 \equiv 1 \pmod{p}$，则小于 p 的解只有两个，$x_1 = 1, x_2 = p-1$
如果用费马定理检测得到 $a^{p-1} \equiv 1 \pmod{p}$，我们可以将其转化为二次探测定理的形式，即：

$$(a^{\frac{p-1}{2}})^2 \equiv 1 \pmod{p} \tag{3}$$

如果 $a^{\frac{p-1}{2}} \pmod{p}$ 不等于 1 或者 $p-1$，则 p 一定不是素数。
如果 $a^{\frac{p-1}{2}} \pmod{p}$ 等于 1 或者 $p-1$，则 p 可能是素数。可以再进行一次检测，判断 $a^{\frac{p-1}{4}} \pmod{p}$，直到指数变成奇数。
如果都等于 1 或者 $p-1$，则认为 p 是素数。

1.2 算法流程

1. 先特判筛掉 3 以内的素数。
2. 将待检验的数 n，分解为 $n-1 = 2^t \cdot u$，其中 u 是奇数。
3. 随机选取多个底数 a，满足 $1 < a < n-1$，分别对 $a^u, a^{u\times 2}, a^{u\times 2^2}$…… 进行检验，判断其解是否都是 1，或者在非最后一个数的情况下出现 $n-1$。
4. 如果都满足，则认为这个数就是素数。

1.3 代码实现

我们设置一个参数 k，表示随机选取底数的次数，一般取 $k = 5$ 即可。
代码如下：

```
1 import random
2 def miller_rabin(n, k=5):
3     if n <= 1:
4         return False
5     if n <= 3:
6         return True
```

```
    d = n - 1
    s = 0
    while d % 2 == 0:
        d //= 2
        s += 1

    for _ in range(k):
        a = random.randint(2, n - 1)
        x = pow(a, d, n)
        if x == 1 or x == n - 1:
            continue
        for _ in range(s - 1):
            x = pow(x, 2, n)
            if x == n - 1:
                break
        else:
            return False
    return True
```

2 Miller-Rabin 素性检测算法的运行效率分析

我们随机生成 100 个 2048bit 的素数，并使用 Miller-Rabin 算法进行检测，统计检测时间，并计算正确率。

2.1 测试代码

```
import random
import time
from Crypto.Util.number import *


def miller_rabin(n, k=5):
    if n <= 1:
        return False
    if n <= 3:
        return True

    d = n - 1
    s = 0
    while d % 2 == 0:
        d //= 2
```

```
        s += 1

    for _ in range(k):
        a = random.randint(2, n - 1)
        x = pow(a, d, n)
        if x == 1 or x == n - 1:
            continue
        for _ in range(s - 1):
            x = pow(x, 2, n)
            if x == n - 1:
                break
        else:
            return False
    return True
sum_time=0
sum=0
for i in range(100):
    p=getPrime(2048)
    start_time= time.time()
    result= miller_rabin(p)
    end_time= time.time()
    print(result)
    print("time:",end_time-start_time)
    sum_time+=end_time-start_time
    if result:
        sum+=1
print(f"average time:{sum_time/100}s")
print(f"Accuracy:{sum}%")
```

2.2 测试结果

平均检测时间 (s)	正确率
0.09710812568664551	100%

测试结果表明，Miller-Rabin 素性检测算法在检测 2048bit 的素数时，平均检测时间为 0.097s，正确率为 100%。