Teste de Primalidade

Rafael de Lucena Valle, Universidade Federal de Santa Catarina 10/10/2014

Teste de Primalidade de Miller Rabin

Números primos são muito utilizados em sistemas de segurança computacional, entretanto são somente úteis se tiverem centenas de dígitos. Não se conhece um método rápido para gerar diretamente números primos grandes, mas existem métodos para se verificar, alguns probabilísticos, se um dado número grande é primo. Dentre os probabilísticos, um dos mais conhecidos e usados é o teste de primalidade de Miller-Rabin, que utiliza as propriedades dos pseudoprimos de Fermat.

Algoritmo

1. Dado um numero inteiro impar n, represento-o de forma fatorada em $(2^r) * s + 1$ com s sendo primo.
2. Escolhe-se um numero randomico a numa faixa de 2 ate n-1.
3. Testo as condições $a^s = 1 \mod n$ ou $a^2js = -1 \mod n$ para cada $j|0 <= j <= r - 1$
4. Um número primo vai passar nos testes para todo valor a, se não encontrar nenhum valor nesta faixa é provavel que seja um número composto.

Exemplos

Para ilustrar a execução do algoritmo, vou mostrar um número composto 33 e o número primo 7.

Número composto

Verificar se 33 é primo. decompondo 33 na forma de $(2^r) * s + 1$ com r ímpar:
Fazendo a decomposição de n para $(2^r) * s$:
32 é divido por 2, 5 vezes
$33 = (2^5) * 1 + 1$, $s = 1$ e $r = 5$
a inteiro na faixa [2, n-1]
$a^s = 1 \mod n$, $a^1 = 1 \mod n$, $a^1 = 1 \mod 33$, $a^1 = 1$
$a^1 = 1$ não existe número inteiro maior que 1 que satisfaz
$a^(s * 2 * j) = n - 1 \mod n$, $a^(2 * 2 * j) = n - 1 \mod n$, $a^4 j = n - 1$
para $0 <= j < r$
$j = 0$
$1 = n - 1$, $1 = 32$, $1 \neq 32$, $j = 1$, $a^4 = 32$
$a =??$ não existe número inteiro que satisfaz

$j = 2$

$a^4 = 32$

$a =??$ não existe número inteiro que satisfaz

$j = 2$

$a^8 = 32$

$a =??$ não existe número inteiro que satisfaz

$j = 3$

$a^12 = 32$

$a =??$ não existe número inteiro que satisfaz

$j = 4$

$a^(2 * 2 * 4) = 32$, $a^16 = 32$

$a =??$ não existe número inteiro que satisfaz, não foi encontrado nenhum valor que satisfaçam as condições, é o número 33 não é primo. Prova: $3 * 11 = 33$

Número Primo

Testando um número primo $7 = 2^t * s + 1$

$6/s = 2^t$, $6/6 = 2^0$, $s = 6$ e $t = 0$

Fazendo os testes: $a^s = 1 \mod n$, $a^6 = 1 \mod 7$, $1^6 = 1$

OK o número não é composto

Código

Listing 1: Miller Rabin prime test em Python.

```python
import random
import sys

"""
Decompoe um numero par na forma (2^r) * s
"""
def decomposeBaseTwo(n):
    exponentOfTwo = 0
    while n % 2 == 0:
      n = n/2
      exponentOfTwo += 1

    return exponentOfTwo, n

"""
Verifica as condicoes
    Se (a^s === 1 (mod n) ou a^2js === -1 (mod n)
    para um j | 0 <= j <= r-1
"""
def fillPrimeConditions(candidateNumber, p, exponent, remainder):
   candidateNumber = pow(candidateNumber, remainder, p)

   if candidateNumber == 1 or candidateNumber == p - 1:
      return False
```

```
25
26      for _ in range(exponent):
27          candidateNumber = pow(candidateNumber, 2, p)
28
29          if candidateNumber == p - 1:
30              return False
31
32      return True
33
34  """
35    O numero randomico a na faixa que inicia em 2 pois, o teste 1^s = 1(mod n)
36    Seria uma tentavia inutil
37  """
38  def probablyPrime(p, accuracy=100):
39      if p == 2 or p == 3: return True
40      if p < 2: return False
41
42      numTries = 0
43      exponent, remainder = decomposeBaseTwo(p - 1)
44
45      for _ in range(accuracy):
46          candidateNumber = random.randint(2, p - 2)
47          if fillPrimeConditions(candidateNumber, p, exponent, remainder):
48              return False
49
50      return True
51
52
53  def checkIsPrime():
54      number = int(raw_input("Give some number to check if is prime: "))
55      if (number == 1):
56          print("\n\tOne is prime!\n")
57          sys.exit()
58      precision = raw_input("Which precision?: ")
59      if (probablyPrime(number, int(precision))):
60          print "\n\tThe number is probably prime!\n"
61      else:
62          print "\n\tThis is a compose number!\n"
63
64  def randomWithNDigits(n):
65      range_start = 10**(n-1)
66      range_end = (10**n)-1
67      return random.randint(range_start, range_end)
68
69  def generateRandomPrime():
70      digit = int(raw_input("Give the number of digits to create a random number: "))
71      if digit < 1:
72          print("\tMust be 1 digits or more!\n")
73          return
74      precision = int(raw_input("Which precision to test primality? "))
75      random_number = randomWithNDigits(digit)
76      while (probablyPrime(random_number, precision) == False):
77          random_number = randomWithNDigits(digit);
78      print "\tThe random number probably prime is: ", random_number, "\n\n"
79
80  def main():
81      print("---- Miller Rabin Primality Test ----")
82      functions = {
83              'prime':checkIsPrime,
```

```
84              'random':generateRandomPrime,
85              'exit':sys.exit
86              }
87
88      while (1):
89          print ("Type random will try generates a random or prime for check if is prime\↩
                n")
90          type = raw_input("Choose random, prime or exit: ")
91          if type not in functions.keys():
92              print ("Invalid choice!")
93              continue
94          functions[type]()
95
96  if __name__ == '__main__':
97      main()
```

Execução

Para executar o script basta abrir um terminal e utilizar um shell tipo o bash, o script é interativo.

Listing 2: Executando o script.

```
1  python miller_rabin.py
2
3  ---- Miller Rabin Primality Test ----
4  Type random will try generates a random or prime for check if is prime
5
6  Choose random, prime or exit: random
7  Give the number of digits to create a random number: 200
8  Which precision to test primality? 100
9      The random number probably prime is:  ↩
           974279308506310072099323731115382400782388944436785931584710025922217829221317262074854667015503
```