Primality Testing

With Algorithm Implementations in Python

Jackson Howe - Supervisor: Kumar

Western University

October 13, 2023

Outline

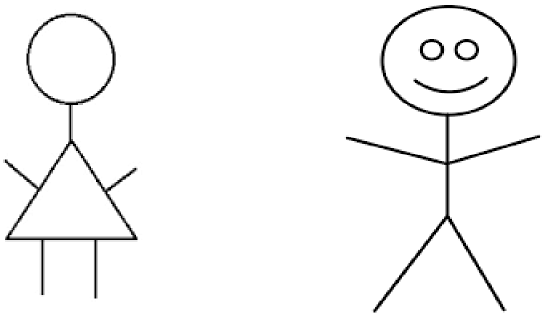
A History of Primality Testing Before Computers

- **3rd century BCE** Sieve of Eratosthenes: returns all numbers less than n that are prime
- **1228** Fibonacci gives algorithm to determine if n is prime by dividing it by numbers up to $\sqrt{n}$
- **1548 - 1626** Perfect Numbers: if $2^n - 1$ is prime, then n is prime and $(2^n - 1)(2^n - 1)$ is perfect.

Why Primes are Useful

- Primes are used in everyday life, but the main use of primes is in cryptography
- Cryptograpic protocols are used in web server requests, ecommerce, secure communication and data exchange, and generally enforcing privacy

Why Primes are Useful



- Alice wants to send a message, m to Bob
- Bob will generate two keys, a public key and a private key

RSA

- **Key Generation**
  - N = pq, where p, q are "large" primes
  - $\phi(N) = \{a \in \mathbb{N} : 1 \leq a \leq N,\ (a, N) = 1\}$
  - Choose e such that (e, $\phi(N)$) = 1
  - public key: [N, e], private key: d $\equiv e^{-1} \pmod{\phi(N)}$
- m := unencoded message, c := cipher text
- **Encryption**
  - c = $m^e \pmod{N}$
- **Decryption**
  - m $\equiv c^d \pmod{N}$
  - $c^d \equiv m^{ed} \equiv m \pmod{N}$
- **Finding Primes**
  - *Prime Number Theorem*: For numbers n $\in \mathbb{N}$ with the "same" number of digits, it will take approximately **log(n)** tries to find a prime

Naive Tests

- We search for a deterministic polynomial time with respect to the input length, so a test with complexity of the form $log^t n$, where $t \in \mathbb{N}$

Primality Testing Strategy

For some n $\in \mathbb{N}$, $\forall\ 2 \leq$ m $\leq \sqrt{n}$, test if m $|$ n

Complexity

Complexity is $O(2^{log_2(\sqrt{n})})$ + memory issues

Fermat's Test

Theorem (Fermat's Little Theorem)

Let p be a prime number and let $a \in \mathbb{Z}$ be coprime. Then

$$a^{p-1} \equiv 1 \pmod{p}$$

Fermat's Test

for some n $\in \mathbb{N}$,

$$\text{if } n \nmid a \text{ and } a^{n-1} \equiv 1 \pmod{n}$$

then n is prime

Fermat Witnesses and Liars

Definition

If $a \in \mathbb{N}$ is such that (a, n) $= 1$ and $a^{n-1} \not\equiv 1 \pmod{n}$, a is called a *Fermat witness* for n and n is definitely composite

Definition

if $a \in \mathbb{N}$ is such that (a, n) $= 1$ and $a^{n-1} \equiv 1 \pmod{n}$, but n is composite, a is called a *Fermat liar* for n

Carmichael Numbers

- Consider the number 561
- $2^{560} \equiv 1 \pmod{561}$, $5^{560} \equiv 1 \pmod{561}$, ..., $379^{560} \equiv 1 \pmod{561}$
- But $561 = 3 \cdot 11 \cdot 17$

Definition

if some number $c \in \mathbb{N}$ satisfies $a^{c-1} \equiv 1$ (mod c) for 2 $\leq$ a $\leq$ c - 1 such that (a, c) = 1, but c is composite, then c is called a *Carmichael Number*

Validity

There are infinitely many Carmichael Numbers!

Fermat's Algorithm and Probability

Data: n $\in \mathbb{N}$
Result: n in PRIMES
Choose a random $2 \leq a \leq n-1$
if *(n, a) == 1* **then**
  if $a^{n-1} \not\equiv 1 \pmod{n}$ **then** *return false*;
  else *return true;*
end

Primality Testing Strategy

For n $\in \mathbb{N}$, such that n is not a Carmichael number one Fermat test has a probability of being correct of at least $1/2$.

Non-trivial Square Roots

Definition

a **non-trivial square root modulo n** is some number a not equal to 1 or n - 1 such that $a^2 \equiv 1 \pmod{n}$

Example

For example,

$$4^2 \equiv 1 \pmod{15},$$

so 4 is a non-trivial square root modulo 15

Non-trivial Square Roots

Theorem

For $n \in \mathbb{N}$, if n is prime, then the $x^2 \equiv 1 \pmod{n}$ has no nontrivial solutions (only 1 and -1)

Theorem

For $n \in \mathbb{N}$ such that $n = pq$, where p and q are distinct odd primes, then $x^2 \equiv 1 \pmod{n}$ has non-trivial solutions

Remark

This congruence is "stronger" than Fermat's test, as there is somewhat of a converse

Miller-Rabin Test

Recall

Fermat's Theorem:

$$a^{p-1} \equiv 1 \pmod{p} \text{ if } p \nmid a$$

or

$$a^{p-1} - 1 \equiv 0 \pmod{p} \text{ if } p \nmid a$$

Difference of Squares

As long as $p-1$ is even, we can continue to factor this equation as a difference of squares, we get:

$$(a^{(p-1)/2^k} - 1)(...)(a^{(p-1)/2} + 1) \equiv 0 \pmod{p}$$

Miller-Rabin Test

Data: n $\in \mathbb{N}$
Result: n in PRIMES
if *n > 2 and n is even* **then**
  return false
end
s $\leftarrow$ 0
t $\leftarrow$ n - 1
while *t is even* **do**
  s $\leftarrow$ s + 1
  t $\leftarrow$ t // 2
end

Miller-Rabin Test Continued and Probability

Randomly select some x $\in$ 1,2,...,n - 1
if $x^{n-1} \not\equiv 1$ **then**
  return false
end
for *i in 1,2,...,s* **do**
  if $x^{2^i t} \equiv 1 \pmod{n}$ *and* $x^{2^{i-1} t} \not\equiv \pm 1 \pmod{n}$ **then**
    return false
  end
end
return true

Primality Testing Strategy

The probability this test is correct is $3/4$ **always**

Time Complexity

The time complexity is $log^3 n$

AKS Test

- In 2002 Agrawal, Kayal, and Saxena developed a deterministic polynomial-time primality test, meaning the probability of correctness is 1
- The basis of the test is the fact that for X $\in$ P[x], a $\in \mathbb{Z}$, n $\in \mathbb{N}$, n is prime if and only if

$$(X+a)^n \equiv X^n + a \pmod{n}$$

- The time complexity for this algorithm is approx. O($log^{15/2} n$), meaning that for the time being, the Miller-Rabin test is still superior for most practical applications

References

- *An Introduction to Mathematical Cryptography* - Hoffstein, Pipher, Silverman
- *An Introduction to the Theory of Numbers* - Niven, Zuckerman, Montgomery
- *Elementary Number Theory: Primes Congruences, and Secrets* - Stein
- *PRIMES is in P* - Agrawal, Kayal, Saxena
- *The Miller-Rabin Randomized Primality Test* - Kleinberg, Cornell University