MA/CSSE 473 Day 9

Primality Testing

Encryption Intro

The algorithm (modified)

- To test N for primality
  - Pick positive integers $a_1, a_2, \ldots, a_k < N$ at random
  - For each a_i, check for $a_i^{N-1} \equiv 1 \pmod{N}$
    - Use the Miller-Rabin approach, (next slides) so that Carmichael numbers are unlikely to thwart us.
    - If a_i^{N-1} is not congruent to 1 (mod N), or Miller-Rabin test produces a non-trivial square root of 1 (mod N)
      - return false
  - return true

Note that this algorithm may produce a “false prime”, but the probability is very low if k is large enough.

Miller-Rabin test

- A **Carmichael number** N is a composite number that passes the Fermat test for all **a** with $1 \le$ **a** <N and gcd(a, N)=1.
- **A way around the problem (Rabin and Miller):** Note that for some t and u (u is odd), $N-1 = 2^t u$.
- As before, compute a^{N-1}(mod N), but do it this way:
  - Calculate a^u (mod N), then repeatedly square, to get the sequence
    a^u (mod N), a^{2u} (mod N), …, $a^{2^t u}$ (mod N) $\equiv a^{N-1}$ (mod N)
- Suppose that at some point, $a^{2^i u} \equiv 1$ (mod N), but $a^{2^{i-1} u}$ is not congruent to 1 or to N-1 (mod N)
  - then we have found a nontrivial square root of 1 (mod N).
  - We will show that if 1 has a nontrivial square root (mod N), then N cannot be prime.

Example (first Carmichael number)

- N = 561. We might randomly select a = 101.
  - Then $560 = 2^4 \cdot 35$, so u=35, t=4
  - $a^u \equiv 101^{35} \equiv 560$ (mod 561) which is -1 (mod 561) (we can stop here)
  - $a^{2u} \equiv 101^{70} \equiv 1$ (mod 561)
  - …
  - $a^{16u} \equiv 101^{560} \equiv 1$ (mod 561)
  - So 101 is not a witness that 561 is composite (we say that 101 is a ***Miller-Rabin liar*** *for 561,* if indeed 561 is composite)
- Try a = 83
  - $a^u \equiv 83^{35} \equiv 230$ (mod 561)
  - $a^{2u} \equiv 83^{70} \equiv 166$ (mod 561)
  - $a^{4u} \equiv 83^{140} \equiv 67$ (mod 561)
  - $a^{8u} \equiv 83^{280} \equiv 1$ (mod 561)
  - So 83 is a witness that 561 is composite, because 67 is a non-trivial square root of 1 (mod 561).

Lemma: Modular Square Roots of 1

- If there is an s which is neither 1 or -1 (mod N), but $s^2 \equiv 1$ (mod N), then N is not prime
- **Proof** (by contrapositive)**:**
  - Suppose that N is prime and $s^2 \equiv 1$ (mod N)
  - $s^2 - 1 \equiv 0$ (mod N) [subtract 1 from both sides]
  - $(s - 1)(s + 1) \equiv 0$ (mod N) [factor]
  - So N divides (s - 1) (s + 1) [def of congruence]
  - Since N is prime, N divides (s - 1) or N divides (s + 1) [def of prime]
  - S is congruent to either 1 or -1 (mod N) [def of congruence]
- This proves the lemma, which validates the Miller-Rabin test

Accuracy of the Miller-Rabin Test

- Rabin* showed that if N is composite, this test will demonstrate its non-primality for at least ¾ of the numbers **a** that are in the range 1…N-1, even if **a** is a Carmichael number.
- Note that 3/4 is the worst case; randomly-chosen composite numbers have a much higher percentage of witnesses to their non-primeness.
- If we test several values of **a**, we have a very low chance of incorrectly flagging a composite number as prime.

*Journal of Number Theory 12 (1980) no. 1, pp 128-138

Efficiency of the Test

- Testing a k-bit number is $\Theta(k^3)$
- If we use the fastest-known integer multiplication techniques (based on Fast Fourier Transforms), this can be pushed to $\Theta(k^2 * \log k * \log \log k)$

Testing "small" numbers

- **From Wikipedia article on the Miller-Rabin primality test:**
- When the number N we want to test is small, smaller fixed sets of potential witnesses are known to suffice. For example, Jaeschke* has verified that
  - if N < 9,080,191, it is sufficient to test a = 31 and 73
  - if N < 4,759,123,141, it is sufficient to test a = 2, 7, and 61
  - if N < 2,152,302,898,747, it is sufficient to test a = 2, 3, 5, 7, 11
  - if N < 3,474,749,660,383, it is sufficient to test a = 2, 3, 5, 7, 11, 13
  - if N < 341,550,071,728,321, it is sufficient to test a = 2, 3, 5, 7, 11, 13, 17

* Gerhard Jaeschke, "On strong pseudoprimes to several bases", Mathematics of Computation 61 (1993)

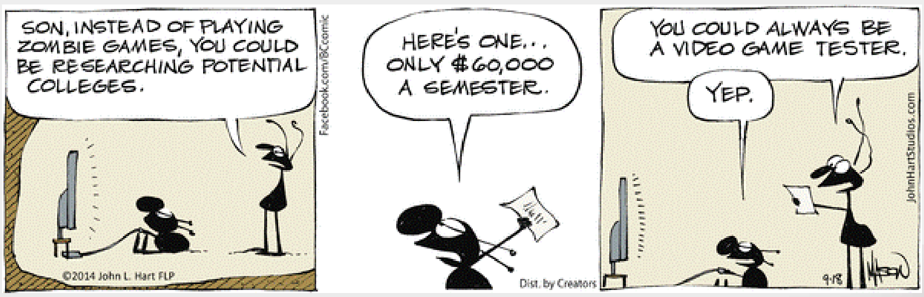
Generating Random Primes

- For cryptography, we want to be able to quickly generate random prime numbers with a large number of bits
- Are prime numbers abundant among all integers? Fortunately, yes
- Lagrange's prime number theorem
  - Let $\pi(N)$ be the number of primes that are $\leq N$, then $\pi(N) \approx N / \ln N$.
  - Thus the probability that an k-bit number is prime is approximately $(2^k / \ln(2^k)) / 2^k \approx 1.44 / k$

Random Prime Algorithm

- To generate a random k-bit prime:
  - Pick a random k-bit number N
  - Run a primality test on N
  - If it passes, output N
  - Else repeat the process
  - Expected number of iterations is $\Theta(k)$

Interlude



We'll only scratch the surface, but there is MA/CSSE 479

CRYPTOGRAPHY INTRODUCTION

Cryptography Scenario

- I want to transmit a message **m** to you
  - in a form **e**(**m**) that you can readily decode by running **d**(**e**(**m**)),
  - And that an eavesdropper has little chance of decoding
- Private-key protocols
  - You and I meet beforehand and agree on e and d.
- Public-key protocols
  - You publish an e for which you know the d, but it is very difficult for someone else to guess the d.
  - Then I can use e to encode messages that only you* can decode

* and anyone else who can figure out what d is if they know e.

Messages can be integers

- Since a message is a sequence of bits …
- We can consider the message to be a sequence of b-bit integers (where b is fairly large), and encode each of those integers.
- Here we focus on encoding and decoding a single integer.

RSA Public-key Cryptography

- Rivest-Shamir-Adleman (1977)
  - A reference : Mark Weiss, Data Structures and Problem Solving Using Java, Section 7.4
- Consider a message to be a number modulo N, an k-bit number (longer messages can be broken up into k-bit pieces)
- The encryption function will be a bijection on {0, 1, …, N-1}, and the decryption function will be its inverse
- How to pick the N and the bijection?

bijection: a function f from a set X to a set Y with the property that for every y in Y, there is exactly one x in X such that f(x) = y. In other words, f is both one-to-one and onto.

N = p q

- Pick two large primes, p and q, and let N = pq.
- **Property**: If e is any number that is relatively prime to N' = (p-1)(q-1), then
  - the mapping $x \rightarrow x^e$ mod N is a bijection on {0, 1, …, N-1}, and
  - If d is the inverse of e mod (p-1)(q-1), then for all x in {0, 1, …, N-1}, $(x^e)^d \equiv x \pmod{N}$.
- We'll first apply this property, then prove it.

Public and Private Keys

- The first (bijection) property tells us that $x \rightarrow x^e$ mod N is a reasonable way to encode messages, since no information is lost
  - If you publish (N, e) as your *public key*, anyone can encrypt and send messages to you
- The second tells how to decrypt a message
  - When you receive a message m', you can decode it by calculating $(m')^d$ mod N.

Example (from Wikipedia)

- p=61, q=53. Compute N = pq = 3233
- (p-1)(q-1) = 60·52 = 3120
- Choose e=17 (relatively prime to 3120)
- Compute multiplicative inverse of 17 (mod 3120)
  - d = 2753 (evidence: 17·2753 = 46801 = 1 + 15·3120)
- To encrypt m=123, take 123^{17} (mod 3233) = 855
- To decrypt 855, take 855^{2753} (mod 3233) = 123
- In practice, we would use much larger numbers for p and q.
- On exams, smaller numbers ☺

Recap: RSA Public-key Cryptography

- Consider a message to be a number modulo N, n k-bit number (longer messages can be broken up into n-bit pieces)
- Pick any two large primes, p and q, and let N = pq.
- **Property**: If e is any number that is relatively prime to (p-1)(q-1), then
  - the mapping $x \rightarrow x^e$ mod N is a bijection on {0, 1, …, N-1}
  - If d is the inverse of e mod (p-1)(q-1), then for all x in {0, 1, …, N-1}, $(x^e)^d \equiv x \pmod{N}$
- We have applied the property; we should prove it