Modular Arithmetic

Mod is the remainder of two numbers.

$$a \bmod b = \text{the remainder of } a \div b$$

Mod can be easily computed with the following formula:

$$a \bmod b = a - b\left\lfloor \frac{a}{b} \right\rfloor$$

Inverse modulo

For modulus n, b is the inverse of a when:

$$a \times b \mod n = 1 \mid 0 < a, b < n$$

The inverse of a exists if and only if a and n are coprime. That is, $\gcd(a, n) = 1$.

Primes

Fermat's Little Theorem. Let p be a prime number, and a an integer that does not divide p.

Then:

$$a^{p-1} \bmod p = 1$$

Euler-Fermat Generalization. Fermat's Little Theorem can be generalized as:

$$a^{\phi(n)} \bmod n = 1$$

Where $\phi(n)$ is Euler's totient function, which gives us the number of integers less than or equal to n that are coprime to n.

Problem 1: Try and prove the Euler-Fermat generalization using Fermat's Little Theorem:

© Caroline Liu, Vincent Macri, and Samantha Unger, 2018

Factoring

If b divides a with no remainder, we will write:

$$b \mid a$$

Read as "b divides a"
Unique Factorization Theorem. Every positive integer has a unique representation as a product of prime numbers.

That is, for all numbers $n \in \mathbb{Z}^+$:

$$n = p_1^{a_1} \times p_2^{a_2} \times \cdots \times p_k^{a_k}$$

Where p_i is prime, and a_i is a positive integer.

Greatest common divisor

$$x = \gcd(a, b) \mid a, b \in \mathbb{Z}$$

Where x is the largest number such that:

$$x \mid a \wedge x \mid b$$

Two numbers are coprime (only share 1 as a factor) when their gcd is 1.

Euclidean algorithm

To find $\gcd(a, b)$, do the following:

1. Let $r_0 = a$, $r_1 = b$, and $i = 1$.
2. If $r_i = 0$ then $\gcd(a, b) = r_i$.
3. Write $r_{i-1} = q_i r_i + r_{i+1}$ and increment i by 1.

   Here, $r_{i+1} = r_i \bmod r_{i-1}$.
4. Go back to step 2.

The extended form of the algorithm lets us find two integers, x and y, such that:

$$\gcd(a, b) = ax + by$$

We can use this to find the inverse of modular multiplication.

© Caroline Liu, Vincent Macri, and Samantha Unger, 2018

RSA

Key generation

The first step of RSA encryption is to generate a public-private keypair.

1. Start by picking two random prime numbers, p and q, that are similar in size. The bigger the better.
2. Calculate:
   (a) $n = p \times q$
   (b) $\phi(n) = (p-1) \times (q-1)$
3. Next, choose a random integer e such that $0 < e < \phi(n)$ and e has an inverse in $\bmod \phi(n)$ (e and $\phi(n)$ are coprime).
4. For the final step, compute d to be the inverse of e. $e \times d \mod \phi(n) = 1 \mid 0 < d < \phi(n)$

Your public keypair is (n, e). Your private key is d.

Encrypting

To encrypt a message m such that $m < n$, calculate the *ciphertext*, c, as so:

$$c = m^e \bmod n$$

Decrypting

To decrypt c, apply the inverse of raising m to the e: raise m to the d.

$$m = r = c^d \bmod n$$

Last Problem

Encrypt whatever you want using RSA.

© Caroline Liu, Vincent Macri, and Samantha Unger, 2018