

week8_lec 1

Main Ideas

- Number Theoretic Algorithms
 - Greatest Common Divisor
 - Extended Euclid's Algorithm
 - Modular Division
- Public Key Cryptography
- RSA Encryption

Number Theoretic Algorithms

We will look into various Number Theoretic algorithms like Greatest Common Divisor, Extended Euclid's Algorithm, Modular Division.

Greatest Common Divisor

For two numbers a and b , the greatest common divisor is the largest number which divides both a and b and leaves no remainders.

Property : $\gcd(a, b) = \gcd(amodb, b)$

Proof : To prove the above, we first have to prove $\gcd(a, b) = \gcd(a-b, b)$

- Let $c = a - b$

The $\gcd(a, b)$, by definition, evenly divides a . As a result, a must be some multiple of $\gcd(a, b)$. i.e. $X \cdot \gcd(a, b) = a$ where X is some integer

The $\gcd(a, b)$, by definition, evenly divides b . As a result, b must be some multiple of $\gcd(a, b)$. i.e. $Y \cdot \gcd(a, b) = b$ where Y is some integer

$a - b = c$ gives us:

$$X \cdot \gcd(a, b) - Y \cdot \gcd(a, b) = c$$

$$(X - Y) \cdot \gcd(a, b) = c$$

So we can see that $\gcd(a, b)$ evenly divides c .

- The $\text{GCD}(b,c)$, by definition, evenly divides b . As a result, b must be some multiple of $\text{GCD}(b,c)$. i.e. $M \cdot \text{GCD}(b,c) = b$ where M is some integer

The $\text{GCD}(b,c)$, by definition, evenly divides c . As a result, c must be some multiple of $\text{GCD}(b,c)$. i.e. $N \cdot \text{GCD}(b,c) = c$ where N is some integer

$a - b = c$ gives us:

$$b + c = a$$

$$M \cdot \text{GCD}(b,c) + N \cdot \text{GCD}(b,c) = a$$

$$(M + N) \cdot \text{GCD}(b,c) = a$$

So we can see that $\text{GCD}(b,c)$ evenly divides a .

Hence any common factor of $a - b$ and b is also a common factor of a and b .

We have proved $\text{GCD}(a,b) = \text{GCD}(b, a-b)$

Then the the initial property is trivial to prove since mod operation is basically $a - q \cdot b$, where q is a positive integer.

From the property, we arrive at the below pseudo code

```
gcd(a, b)
if(b=0): return a
else
    return gcd(b, a mod b)
```

Time Complexity : $O(\log(\max(m, n)))$

since if $a \geq b$, then $a \bmod b < a/2$

Extended Euclid's Algorithm

First, we need to learn Bezout's Identity.

Bezout's Identity : if $d = \text{gcd}(a, b)$ then there exist integers x and y for which $d = ax + by$

also

If d divides both a and b and $d = ax + by$ for some integers a and b , then necessarily $d = \gcd(a, b)$

Proof : Since it is given that d divides a and b therefore d is a common divisor of a and b . Therefore $d \leq \gcd(a, b)$. Also since $\gcd(a, b)$ divides both a and b , therefore it divides $ax + by$ as well, meaning that it divides d . Therefore $\gcd(a, b) \leq d$. Combining both the equations we get $\gcd(a, b) = d$

Euclid's algorithm helps to find d and also Bezout's coefficients x and y

Therefore, the pseudocode is as follows :

Input : a and b

Output : d, x, y

```
EEA(a, b)
if b=0:
    return (1, 0, a)
else
    (x', y', d)=EEA(b, a mod b)
    return (y', x' - [a/b]y', d)
```

Modular Division

We say that the number x is the multiplicative inverse of $(a \bmod N)$ if $ax = 1 \pmod{N}$

We use the extended euclid algorithm, to help find out the multiplicative inverse for any $a \bmod N$. We notice that a can only have a multiplicative inverse if and only if it is co-prime to N . Hence $\gcd(a, N) = 1$. Notice that this makes it very easy to find the multiplicative inverse. We know from bezout's identity that there exist integers x and y such that

$$ax + Ny = 1$$

Taking modulo N on both sides, we get that the multiplicative inverse of a is nothing but the bezout coefficient corresponding to a .

Public Key Cryptography

Different Operations on numbers

Addition

Multiplication

Comparison

Different Representations of numbers




Roman numeral system (I , II , III , IV, and so on)

Decimal system (1,2,3,4 and so on)

Prime factorization/ prime product system (1, 2, 3, 2^2 , 5, 2.3 and so on)

Residue system involving the mod operation , and others

Different operations have different speeds, depending on the type of representation we are performing the operation in.

<u>Aa</u> Name	 Addition	 Multiplication	 Comparision
<u>Roman</u>	Slow	Slow	Slow
<u>Decimal</u>	Fast	Medium	Fast
<u>Prime Product</u>	Slow	Fast	Medium
<u>Residue System</u>	Fast	Fast	Medium

Slow systems are used in cryptography.

Public key Cryptography involves a key K. This key is written in two different representations, one of which is made public, and the other is kept private.

In the public representation of the key, the encryption process has to be extremely fast, whereas the decryption process has to be extremely slow, to the point where it is no longer feasible to decrypt.

However, in the private key representation, the decryption process is very fast.

We also need to make sure that the process to convert from one representation to the other, is also very slow, otherwise there is no point in having the private and public representation separate.

RSA Encryption

Working

- Take any two prime numbers p and q and $N = pq$

- Take e to be a number relatively prime to $(p - 1)(q - 1)$
- Hence (N, e) is our public key, which we can use to encrypt messages.
- The encryption algorithm is $\text{encrypted}(x) = x^e \bmod N$
- The numbers p, q however are kept private, and we find d to be the multiplicative inverse of e modulo $(p - 1)(q - 1)$.
- Decryption algorithm, if we know d : $(x^e)^d \bmod N = x \bmod N$

Hence we can get back our original message from the encrypted message easily.

In order to understand the correctness of the algorithm we first need to understand the Fermat's little theorem, which states that if there is a prime number p then for every $1 \leq a \leq p$

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

Now we shall use this to prove correctness, we need to prove that

$$(x^e)^d \bmod N = x \bmod N$$

We can do this by proving that

$$(x^e)^d - x = x^{1+k(p-1)(q-1)} - x = x(x^{k(p-1)(q-1)} - 1)$$

From Fermat's theorem

$$p \mid x^{k(p-1)(q-1)} - 1$$

$$q \mid x^{k(p-1)(q-1)} - 1$$

since p and q are primes, therefore $pq \mid x^{k(p-1)(q-1)} - 1$ therefore $N \mid x^{k(p-1)(q-1)} - 1$.

Hence $(x^e)^d - x$ must be divisible by N .