Cryptocurrency

Cryptography

2018.02.
JY

Number Theory

Essential Number Theory for Public-Key Algorithms

Euclidean Algorithm

Solving for greatest common divisor is easy!

eg. Let *a* and *b* both positive integers. If $a = 84$ and $b = 30$. Find the greatest common factor.

$a = 84 = 2 * 2 * 3 * 7$
$b = 30 = 2 * 3 * 5$

$gcd(a,b) = 6$

Essential Number Theory for Public-Key Algorithms

Euclidean Algorithm

We can observe $gcd(a,b) = gcd(a-b,b)$, where we assume that a>b, and that both numbers are positive integers.

eg. Let a and b both positive integers. If $a = 84$ and $b = 30$.

$gcd(54,30) = 6$

We can apply the process iteratively :

$gcd(a,b) = gcd(a-b,b) = gcd(a-2b,b) = \ldots = gcd(a-mb,b)$

as long as $(a-bm) > 0$.

Essential Number Theory for Public-Key Algorithms

Euclidean Algorithm

The algorithm uses the fewest number of steps if we choose the maximum value for *m*.

$gcd(a,b) = gcd(a \bmod b, b)$

Since the first term *(a mod b)* is smaller than the second term b, we usually swap them :

$gcd(a,b) = gcd(b, a \bmod b)$

eg.

$gcd(27,21) = gcd(1*21 + 6, 21) = gcd(21,6)$

$gcd(21,6) = gcd(3*6 + 3, 6) = gcd(6,3)$

$gcd(6,3) = gcd(2*3 + 0, 3) = gcd(3,0)$

$\boldsymbol{gcd(27,21) = gcd(21,6) = gcd(6,3) = gcd(3,0) = 3}$

Extended Euclidean Algorithm

An extension of the algorithm allows us to compute modular inverses, which is of major importance in public-key cryptogra phy. In addition to computing the gcd, the extended Euclidean algorithm computes a linear combination of the form :

$$gcd(a,b) = s*a + t*b$$

eg. Let a and b both positive integers. If $a = 84$ and $b = 30$.

$84 = 2*30 + 24$

$30 = 1*24 + 6$

$24 = 6$

$6 = 30 – 1*24$
$\quad = 30 – 1*(84 – 2*30)$
$\quad = –1 *84 + 3*30$

$gcd(84,30) = –1 *84 + 3*30$

where $s = –1$ and $t = 3$.

Essential Number Theory for Public-Key Algorithms

Extended Euclidean Algorithm

An extension of the algorithm allows us to compute modular inverses, which is of major importance in public-key cryptogra phy. In addition to computing the gcd, the extended Euclidean algorithm computes a linear combination of the form :

$$gcd(a,b) = s*a + t*b$$

eg. Let *a* and *b* both positive integers. If *a = 84* and *b = 30*.

i	*r_(i-2) = q_(i-1)*r_(i-1) + r_i*	*r_i = [s_i]r_0 + [t_i]r_1*
2	84 = 2*30 + 24	24 = [1]r0 – [2]r1
3	30 = 1*24 + 6	6 = 30 – 1*24 = r1 – 1(1*r0 – 2*r1) = [-1]r0 + [3]r1
	24 = 6	

Extended Euclidean Algorithm

Input : positive integers r_0 and r_1 with $r_0 > r_1$
Output : $gcd(r_0,r_1)$, as well as s and t such that $gcd(r_0,r_1) = s*r_0 + t*r_1$.
Initialization :
$s_0 = 1, \ t_0 = 0$
$s_1 = 0, \ t_1 = 1$
$i=1$

Algorithm :

1 DO
1.1 $i = i + 1$
1.2 $r_i = r_{(i-2)} \bmod r_{(i-1)}$
1.3 $q_{(i-1)} = (r_{(i-2)} - r_i) / r_{(i-1)}$
1.4 $s_i = s_{(i-2)} - q_{(i-1)}*s_{(i-1)}$
1.5 $t_i = t_{(i-2)} - q_{(i-1)}*t_{(i-1)}$
WHILE $r_i \neq 0$
2 RETURN
$gcd(r_0,r_1) = r_{(i-1)}$
$s = s_{(i-1)}$
$t = t_{(i-1)}$

Extended Euclidean Algorithm

Now let's assume we want to compute the inverse of $b \bmod a$ where $b < a$.
Recall from the properties of integer rings, the inverse only exists if $gcd(a,b) = 1$.
Hence, if we apply the Extended Euclidean Algorithm, we obtain $s*a + t*b = 1 = gcd(a,b)$.
Taking this equation *modulo a* we obtain :

$s*a + t*b = 1$

$s*0 + t*b = 1 \bmod a$

$b*t = 1 \bmod a$

$t = inv(b) \bmod a$

eg. Given $a = 67$, $b = 12$ and a linear combination $-5*67 + 28*12 = 1$. Find the inverse of 12.

Using *mod 67*, an inverse of 12 = 28 mod 67

Essential Number Theory for Public-Key Algorithms

Euler's Phi Function

Definition.

The number of integers in Z_m relatively prime to m is denoted by $ø(m)$.

eg. Let $m = 6$. The associated set is $Z_6 = \{0,1,2,3,4,5\}$.
$gcd(0,6) = 6$
$gcd(1,6) = 1$
$gcd(2,6) = 2$
$gcd(3,6) = 3$
$gcd(4,6) = 2$
$gcd(5,6) = 1$

Since there are two numbers in the set which are relatively prime to 6, namely 1 and 5, the phi function takes the value 2, ie. $ø(6) = 2$.

Euler's Phi Function

Definition.

The number of integers in Z_m relatively prime to m is denoted by $ø(m)$.

eg. Let $m = 6$. The associated set is $Z\_5 = \{0,1,2,3,4\}$.
$gcd(0,5) = 5$
$gcd(1,5) = 1$
$gcd(2,5) = 1$
$gcd(3,5) = 1$
$gcd(4,5) = 1$

$ø(5) = 4$.

Essential Number Theory for Public-Key Algorithms

Euler's Phi Function

Definition.

The number of integers in *Z_m* relatively prime to m is denoted by *ø(m)*.

eg. Let *m = 6*. The associated set is *Z_5 = {0,1,2,3,4}*.
gcd(0,5) = 5
gcd(1,5) = 1
gcd(2,5) = 1
gcd(3,5) = 1
gcd(4,5) = 1

ø(5) = 4.

Euler’s Phi Function

Theorem 6.3.1 *Let m have the following canonical factorization*

$$m = p_1^{e_1} \cdot p_2^{e_2} \cdot \ldots \cdot p_n^{e_n},$$

where the p_i are distinct prime numbers and e_i are positive integers, then

$$\Phi(m) = \prod_{i=1}^{n} (p_i^{e_i} - p_i^{e_i - 1}).$$

eg. Let *m = 240*.

*m = 240 = 16*15 = 2^4 * 3 * 5*
*ø(m) = (2^4 – 2^3)(3^1 – 3^0)(5^1 – 5^0) = 8 * 2 * 4 = 64.*

This means that 64 integers in the range *{0,1,…,239}* are coprime to *m = 240*.

Essential Number Theory for Public-Key Algorithms

Euler's Phi Function

Theorem. Fermat's Little Theorem

Let a be an integer and p be a prime, then :

$a^p = a \pmod{p}$.

The theorem can be stated in the form : $a^{(p-1)} = 1 \pmod{p}$

$a^{(p-1)} = 1 \pmod{p}$
$a * a^{(p-2)} = 1 \pmod{p}$
$a^{(-1)} = a^{(p-2)} \pmod{p}$

eg. Let $p = 7$ and $a = 2$. We can compute the inverse of a as :

a^(p-2) = 2^5 = 32 = 4 mod 7

This is easy to verify : 2*4 = 1 mod 7

Essential Number Theory for Public-Key Algorithms

Euler's Phi Function

Theorem. Euler's Theorem

Let a and m be integers with $gcd(a,m)$ = 1, then :

$a^{\phi(m)} = 1 \ (mod\ m)$.

eg. Let $m = 12$ and $a = 5$. First, we compute Euler's phi function of m :

$\phi(12) = \phi(2^2 * 3) = \phi(2^2 – 2)(3^1 – 3) = (4 – 2)(3 – 1) = 4$

$5^{\phi(12)} = 5^4 = 25^2 = 625 = 1 \ mod\ 12$

If p is a prime, it holds that $\phi(p) = (p^1 – p^0) = p – 1$. If we use this value for Euler's theorem, we obtain :

$a^{\phi(p)} = a^{(p-1)} = 1 \ (mod\ p)$, which is exactly Fermat's Little Theorem.

Reference

Reference

Preneel, B. (2014) *Understanding cryptography*, Springer.

Thank you