Week-4 : Lecture Notes

Topics :

Moduler Inverse.
Extended Euclid Algorithm.
Fermat's Little Theorem,
Euler Phi Function.
Euler's Theorem, Quadratic Residue.
Polynomial Arithmetic.

Moduler Inverse

see Week-3 lecture note.

Extended Euclid Algorithm

- It was stated earliar that the greatest common divisor of two numbers can be found using Euclid's Algorithm.

- This algorithm can be extended so that it not only finds the greatest common divisor but also calculates the inverse of some number b modulo some other number m (assuming it exists).

- For small values of m it is easy enough to find.

- However, for large numbers this approach is not practical.

- Extended Euclid's algorithm allow us to find the inverse of a number b mod m assuming gcd(m, b) = 1.

	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	0	2	4	6
3	0	3	6	1	4	7	2	5
4	0	4	0	4	0	4	0	4
5	0	5	2	7	4	1	6	3
6	0	6	4	2	0	6	4	2
7	0	7	6	5	4	3	2	1

Fig1: Arithmetic modulo 8

Extended Euclid (m, b):

1. $(A_1, A_2, A_3) \leftarrow (1, 0, m)$; $(B_1, B_2, B_3) \leftarrow (0, 1, b)$
2. if $B_3 = 0$ return $A_3 = \gcd(m, b)$; no inverse
3. if $B_3 = 1$ return $B_3 = \gcd(m, b)$; $B_2 = b^{-1} \pmod m$
4. $Q = \lfloor \frac{A_3}{B_3} \rfloor$
5. $(T_1, T_2, T_3) \leftarrow (A_1 - QB_1, A_2 - QB_2, A_3 - QB_3)$
6. $(A_1, A_2, A_3) \leftarrow (B_1, B_2, B_3)$
7. $(B_1, B_2, B_3) \leftarrow (T_1, T_2, T_3)$
8. Go to 2.

- This is seen to work because $bB_2 = 1 - mB_1$ implies that $bB_2 \equiv 1 \pmod m$. Therefore the values of B_2 is a number when multiplied by b will give a value which is congruent to 1 modulo m (in other words it gives a value that when divided by m will leave a remainder of 1)

- We can therefore say:

$$bB_2 = 1 - mB_1, \text{ i.e, } 1 = mB_1 + bB_2 \quad \text{----(1)}$$

- In other words, we are trying to find two values B_1 and B_2 that solve equation (1). These values will be revealed when another value B_3 is equal to 1 in the above algorithm : $mB_1 + bB_2 = B_3$

- In order to find this multiplicative inverse we need to keep track of A_1, A_2 and A_3 also.

- The values T_1, T_2, T_3 are only used for temporary storage.

- Looking at steps 5 and 7, it can be seen the $B_3 \leftarrow A_3 - QB_3$ - a consequence of Euclid's algorithm and it leaves the remainder when A_3 is divided by B_3 (you are subtracting B_3 away from A_3 as many times as you can, remember $Q = \left\lfloor \frac{A_3}{B_3} \right\rfloor$).

- Throughout the algorithm, the following relationships hold:

$$mT_1 + bT_2 = T_3, \quad mA_1 + bA_2 = A_3,$$
$$mB_1 + bB_2 = B_3.$$

- These equations are why the initial assignments are $(1,0,m)$ and $(0,1,b)$. If you work them out you will get the above for A_3 and B_3. The last equation is the one we are interested in and when $B_3 = 1$, then

$$B_2 = b^{-1} \bmod m.$$

- For example, to find the multiplicative inverse of 550 modulo 1759 we have:

Q	A_1	A_2	A_3	B_1	B_2	B_3
–	1	0	1759	0	1	550
3	0	1	550	1	-3	109
5	1	-3	109	-5	16	5
21	-5	16	5	106	-339	4
1	106	-339	4	-111	355	1

Euler's Theorem

- Euler's Theorem can be stated mathematically as:

$$a^{\phi(m)} \equiv 1 \pmod{m}, \quad \gcd(a,m) = 1$$

- 'a' is any integer and m is the modulus (which again, is restricted to being a positive integer)

- The symbol $\phi(m)$ is known as Euler's phi (or totient) function and is the number of positive integers $\leq m$ and relatively prime to it.

- A few points should be noted about $\phi(m)$:
  - The value of $\phi(1)$ is defined as being equal to 1
  - If p is some prime, then $\phi(p) = p-1$ as there are $p-1$ positive integers $< p$ and relatively prime to p.
  - If p and q are prime numbers and $n = pq$ then $\phi(n) = \phi(p).\phi(q) = (p-1).(q-1)$

- If $n = p_k^{\epsilon_k} \times p_{k-1}^{\epsilon_{k-1}} \times \cdots \times p_1^{\epsilon_1}$ is the prime factorization of n, then

$$\phi(n) = n\left(1-\frac{1}{p_k}\right)\left(1-\frac{1}{p_{k-1}}\right)\cdots\left(1-\frac{1}{p_1}\right)$$

- As an example, take $4^{10} = 1048576 \equiv 1 \pmod{11}$ because $11 \times 95{,}325 + 1 = 4^{10}$

- If n is a product of distinct primes, and if $r \equiv s \pmod{\phi(n)}$, then $a^r \equiv a^s \pmod{n}$ for all integers a. In other words, when working modulo such an n, exponents can be reduced modulo $\phi(n)$.

Fermat's Little Theorem

- Fermat's Little theorem is really a specific case of Euler's Theorem when m is a prime.

- Historically though, Fermat's little theorem was discovered long before Euler's Theorem.

- It can be stated as follows:

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

where m is prime and $m \nmid a$.

- If Euler's theorem is taken to be true, then this can also be seen to work because of the fact that $\phi(m) = m-1$ for a prime number as mentioned above.

Polynomial Arithmetic

- We need to understand some things about arithmetic involving polynomials before continuing.

- A polynomial is an expression of the form:

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots\cdots + a_1 x + a_0$$

$$= \sum_{i=0}^{n} a_i x^i$$

- $a_n \neq 0$
- The degree of the polynomial is equal to the value of the integer $n \geq 0$.

- The co-efficients are the set $S=\{a_n, a_{n-1}, \ldots, a_1, a_0\}$ which is known as the co-efficient set.

- If $a_n = 1$ then the polynomial is said to be monic.

- If $n=0$ then we simply have a constant known as a constant polynomial.

- As an example, we can set $n=8$ and $S=\{1,0,0,0,1,1,1,1,1\}$ and we get the following polynomial:

$$\sum_{i=0}^{8} a_i x^i = x^8 + x^4 + x^3 + x^2 + x + 1$$

- The polynomial has important significance for us as it is used in the AES standard as we shall see later.

- As we are going to use these for cryptographic methods, we will want to do some form of arithmetic on them.

- One can do following three classes of arithmetic :

  1. Ordinary polynomial arithmetic, using the basic rules of algebra.

  2. Polynomial arithmetic in which the arithmetic on the coefficients is performed modulo p ; that is the coefficients are in

     $$Z_p = 0, 1, \cdots, p-1$$

  3. Polynomial arithmetic in which the coefficients are in Z_p and the polynomials are defined modulo a polynomial $m(x)$ having highest power, say n

- Can we operate on polynomials using the four basic arithmetical operations of addition, substraction, multiplication and division ?

- Consider two polynomials $f(x) = x^3 + x^2 + 2$ and $g(x) = x^2 - x + 1$. If we add these we get :

$$f(x) + g(x) = x^3 + 2x^2 - x + 3 .$$

- This would seem to suggest we can add (and it turns out we can).

- If we subtract them,

$$f(x) - g(x) = x^3 + x + 1$$

- These would seem to suggest we can subtract (and it turns out we can).

- If we multiply them:

$$f(x) \times g(x) = x^5 + 3x^2 - 2x + 2$$

- This would seem to suggest we can multiply (it turns out we can).

- The order of the product polynomial is equal to the sum of the orders of the two factors. In this case

$$n_{prod} = n_{f(x)} + n_{g(x)} = 3 + 2 = 5$$

- What about division?

- Divisions requires that the set of coefficients $S = \{a_n, a_{n-1}, \cdots, a_1, a_0\}$ be a field.

- In other words S must satisfy the conditions described earlier.

- In general, division will produce a quotient and a remainder so we can write (for two polynomials $f(x)$ and $g(x)$):

$$\frac{f(x)}{g(x)} = q(x) + \frac{r(x)}{g(x)}$$

i.e $f(x) = q(x) \cdot g(x) + r(x)$

Galois Fields

- Saw earlier that a fields obeys axioms $A_1 \rightarrow M_7$.

- It is possible for a set with an infinite number of elements to be a field.

- For example, the set of real numbers is a field under the usual arithmetic operations.

- In cryptography however we are not interested in infinite fields because they can not be worked

with in practice (due to memory limitation etc.)

- what cryptographers want instead are finite fields. These are simply fields with a finite number of elements.

- It turns out that the order of a finite field must be the power of a prime i.e p^n where $n > 0$.

- The finite field of order p^n is normally written $GF(p^n)$.

- The GF stands for Galois Field in honour of the mathematician who first studied them.

- When $n=1$ Galois fields take on a different structure than when $n>1$.

- We will mainly be interested in $GF(p)$ for some prime p and $GF(2^n)$ (2 being the main prime of interest due to computer operates in binary).

- Addition and multiplication in a Galois fields are done modulo $m(x)$, where $m(x)$ is an irreducible polynomial of degree n.

- A polynomial $m(x)$ over a field F is called irreducible if and only if $m(x)$ can not be expressed as a product of two polynomials both over F, and both of degree lower than that of $m(x)$.

- By analogy to integers, an irreducible polynomial is also called a prime polynomial.

- As an example of a Galois field we can look at $GF(2^3)$.

- This has the following elements: $\{0, 1, x, x+1, x^2, x^2+1, x^2+x, x^2+x+1\}$ and an irreducible polynomial is x^3+x+1

- In binary Galois fields all polynomials are monic due to the fact that the coefficients are taken from the set $S = \{0, 1\}$

- Earlier we stated that Z_m was the set of integers modulo m.
- If $m = p$, where p is some prime, then we have $Z_m = Z_p = \{0, 1, 2, \cdots, p-1\}$
- This together with the arithmetic operations modulo p, form the finite field of order p.
- Since p is the power of a prime, this field is a Galois field.
- The following table shows the properties of moduler arithmetic for integers in Z_m.

Property	Expression
Commutative laws	$(w+x) \bmod n = (x+w) \bmod n$ $(w \times x) \bmod n = (x \times w) \bmod n$
Associative laws	$[(w+x)+y] \bmod n = [w+(x+y)] \bmod n$ $[(w \times x) \times y] \bmod n = [w \times (x \times y)] \bmod n$
Distributive laws	$[w \times (x+y)] \bmod n = [(w \times x)+(w \times y)] \bmod n$ $[w + (x \times y)] \bmod n = [(w+x) \times (w+y)] \bmod n$
Identities	$(0+w) \bmod n = w \bmod n$ $(1 \times w) \bmod n = w \bmod n$
Additive inverse $(-w)$	For each $w \in Z_n$, there exist a z such that $w + z \equiv 0 \bmod n$

Fig: Moduler arithmetic for Z_m.