Finite Fields

Dept. of Computer Science & Engg.
IIT Kharagpur, India

Dipanwita Roy Chowdhury

Introduction

- Finite fields have become increasingly important in cryptography.

- A number of cryptographic algorithms rely heavily on properties of finite fields, such as the AES, Elliptic Curve, IDEA, & various Public Key algorithms.

- Groups, rings, and fields are the fundamental elements of abstract algebra

Group

- A Group {G, .}a set of elements with a binary operation .
- Obeys the following axioms:
  - Closure: If a and b belong to G then a.b is also in G
  - associative law: (a.b).c = a.(b.c)
  - has identity e: e.a = a.e = a
  - has inverses a^{-1}: $a.a^{-1} = e$
- if commutative `a.b = b.a`
  - then forms an **abelian group**

Cyclic Group

- define **exponentiation** as repeated application of operator
  - example: $a^3 = a.a.a$
- and let identity be: $e=a^0$
- a group is cyclic if every element is a power of some fixed element
  - ie $b = a^k$ for some a and every b in group
- a is said to be a generator of the group

Ring

- a set of elements with two operations (addition and multiplication) which form:
- an abelian group with addition operation
- and multiplication:
  - has closure
  - is associative
  - distributive over addition: a(b+c) = ab + ac
- if multiplication operation is commutative, it forms a commutative ring
- if multiplication operation has an identity and no zero divisors, it forms an integral domain

Field

- a set of elements with two operations which form **Integral Domain:**
  - Ring
  - Multiplicative identity
  - No zero divisors

Field:

- Multiplicative inverse:

  there exists a^{-1} in F, $(a)a^{-1} = (a^{-1})a = 1$

Divisors

- say a non-zero number b **divides** a if for some m have $a=mb$ (a,b,m all integers)
- that is b divides into a with no remainder
- denote this $b \mid a$
- and say that b is a **divisor** of a
- eg. all of 1,2,3,4,6,8,12,24 divide 24

Modular Arithmetic

- Modulo operator “a mod n” is remainder when a is divided by n
- Congruent modulo n:

  if (a mod n) = (b mod n) then a ≡ b mod n

  - when divided by *n*, a & b have same remainder
  - e.g. 13 mod 7 = 6; 41 mod 7 = 6 -> 13 ≡ 41 mod 7

b is called a residue of a mod n

- since with integers can always write: a = qn + b
- usually chose smallest positive remainder as residue
  - ie. 0 <= b <= n-1
- process is known as modulo reduction
  - eg. -12 mod 7 = -5 mod 7 = 2 mod 7 = 9 mod 7

Modular Arithmetic Operations

- Exhibits following three properties addition, subtraction & multiplication
  - (a+b)mod n = [(a mod n) + (b mod n)] mod n
  - (a-b)mod n = [(a mod n) - (b mod n)] mod n
  - (axb)mod n = [(a mod n) x (b mod n)] mod n

Modular Arithmetic

- can do modular arithmetic with any group of integers: $Z_n = \{0, 1, \dots, n-1\}$
- form a commutative ring for addition
- with a multiplicative identity
- note some peculiarities
  - if $(a+b) = (a+c) \bmod n$

    then $b = c \bmod n$
  - but if $(a.b) = (a.c) \bmod n$

    then $b = c \bmod n$ only if a is relatively prime to n

Modulo 8 Addition Example

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

Modulo 8 Multiplication Example

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

Additive and Multiplicative Inverses Modulo 8

w	-w	W^{-1}
0	**0**	-
1	7	1
2	6	-
3	5	3
4	4	-
5	3	5
6	2	-
7	1	7

Greatest Common Divisor (GCD)

- a common problem in number theory
- GCD (a,b) of a and b is the largest number that divides evenly into both a and b
  - eg GCD(60,24) = 12
- often want **no common factors** (except 1) and hence numbers are **relatively prime**
  - eg GCD(8,15) = 1
  - hence 8 & 15 are relatively prime

Euclidean Algorithm

- an efficient way to find the GCD(a,b)
- uses theorem that:
  - `GCD(a,b) = GCD(b, a mod b)`
- Euclidean Algorithm to compute GCD(a,b) is:

```
EUCLID(a,b)
1. A = a; B = b
2. if B = 0 return  A = gcd(a, b)
3. R = A mod B
4. A = B
5. B = R
6. goto 2
```

Stallings Book

Example GCD(1970,1066)

1970 = 1 x 1066 + 904	gcd(1066, 904)
1066 = 1 x 904 + 162	gcd(904, 162)
904 = 5 x 162 + 94	gcd(162, 94)
162 = 1 x 94 + 68	gcd(94, 68)
94 = 1 x 68 + 26	gcd(68, 26)
68 = 2 x 26 + 16	gcd(26, 16)
26 = 1 x 16 + 10	gcd(16, 10)
16 = 1 x 10 + 6	gcd(10, 6)
10 = 1 x 6 + 4	gcd(6, 4)
6 = 1 x 4 + 2	gcd(4, 2)
4 = 2 x 2 + 0	gcd(2, 0)

Stallings Book

Galois Fields

- finite fields play a key role in cryptography
- can show number of elements in a finite field **must** be a power of a prime p^n
- known as Galois fields
- denoted $GF(p^n)$
- in particular often use the fields:
  - $GF(p)$
  - $GF(2^n)$

Galois Fields GF(p)

- GF(p) is the set of integers {0,1, … , p-1} with arithmetic operations modulo prime p
- these form a finite field
  - since have multiplicative inverses
- hence arithmetic is “well-behaved” and can do addition, subtraction, multiplication, and division without leaving the field GF(p)

GF(7) Multiplication Example

×	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

Finding Inverses

```
EXTENDED EUCLID($m$, $b$)
1.  (A1, A2, A3)=(1, 0, $m$);
    (B1, B2, B3)=(0, 1, $b$)
2. if B3 = 0
    return A3 = gcd($m$, $b$); no inverse
3. if B3 = 1
    return B3 = gcd($m$, $b$); B2 = $b^{-1}$ mod $m$
4. Q = A3 div B3
5. (T1, T2, T3)=(A1 – Q B1, A2 – Q B2, A3 – Q B3)
6. (A1, A2, A3)=(B1, B2, B3)
7. (B1, B2, B3)=(T1, T2, T3)
8. goto 2
```

Inverse of 550 in GF(1759)

Q	A1	A2	A3	B1	B2	B3
—	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

Polynomial Arithmetic

- can compute using polynomials

  $f(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0 = \sum a_i x^i$

  - nb. not interested in any specific value of x
  - which is known as the indeterminate
- several alternatives available
  - ordinary polynomial arithmetic
  - poly arithmetic with coefficients mod p
  - poly arithmetic with coefficients mod p and polynomials mod m(x)

Ordinary Polynomial Arithmetic

- **add or subtract corresponding coefficients**
- **multiply all terms by each other**
- **eg**

let $f(x) = x^3 + x^2 + 2$ and $g(x) = x^2 – x + 1$

$f(x) + g(x) = x^3 + 2x^2 – x + 3$

$f(x) – g(x) = x^3 + x + 1$

$f(x) \text{ x } g(x) = x^5 + 3x^2 – 2x + 2$

Polynomial Arithmetic with Modulo Coefficients

- when computing value of each coefficient do calculation modulo some value
  - forms a polynomial ring
- could be modulo any prime
- but we are most interested in mod 2
  - ie all coefficients are 0 or 1
  - eg. let $f(x) = x^3 + x^2$ and $g(x) = x^2 + x + 1$

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

$$f(x) \text{ x } g(x) = x^5 + x^2$$

Polynomial Division

- can write any polynomial in the form:
  - $f(x) = q(x)\ g(x) + r(x)$
  - can interpret $r(x)$ as being a remainder
  - $r(x) = f(x) \bmod g(x)$
- if have no remainder say $g(x)$ divides $f(x)$
- if $g(x)$ has no divisors other than itself & 1 say it is **irreducible** (or prime) polynomial
- arithmetic modulo an irreducible polynomial forms a field

Polynomial GCD

- can find greatest common divisor for polys
  - $c(x)$ = GCD($a(x)$, $b(x)$) if $c(x)$ is the poly of greatest degree which divides both $a(x)$, $b(x)$
- can adapt Euclid's Algorithm to find it:

```
EUCLID[$a(x)$, $b(x)$]
1. $A(x) = a(x)$; $B(x) = b(x)$
2. if $B(x) = 0$ return $A(x) = \mathrm{gcd}[a(x), b(x)]$
3. $R(x) = A(x) \bmod B(x)$
4. $A(x)$ ¨ $B(x)$
5. $B(x)$ ¨ $R(x)$
6. goto 2
```

Modular Polynomial Arithmetic

- can compute in field $GF(2^n)$
  - polynomials with coefficients modulo 2
  - whose degree is less than n
  - hence must reduce modulo an irreducible poly of degree n (for multiplication only)
- form a finite field
- can always find an inverse
  - can extend Euclid's Inverse algorithm to find

Example GF(2^3)

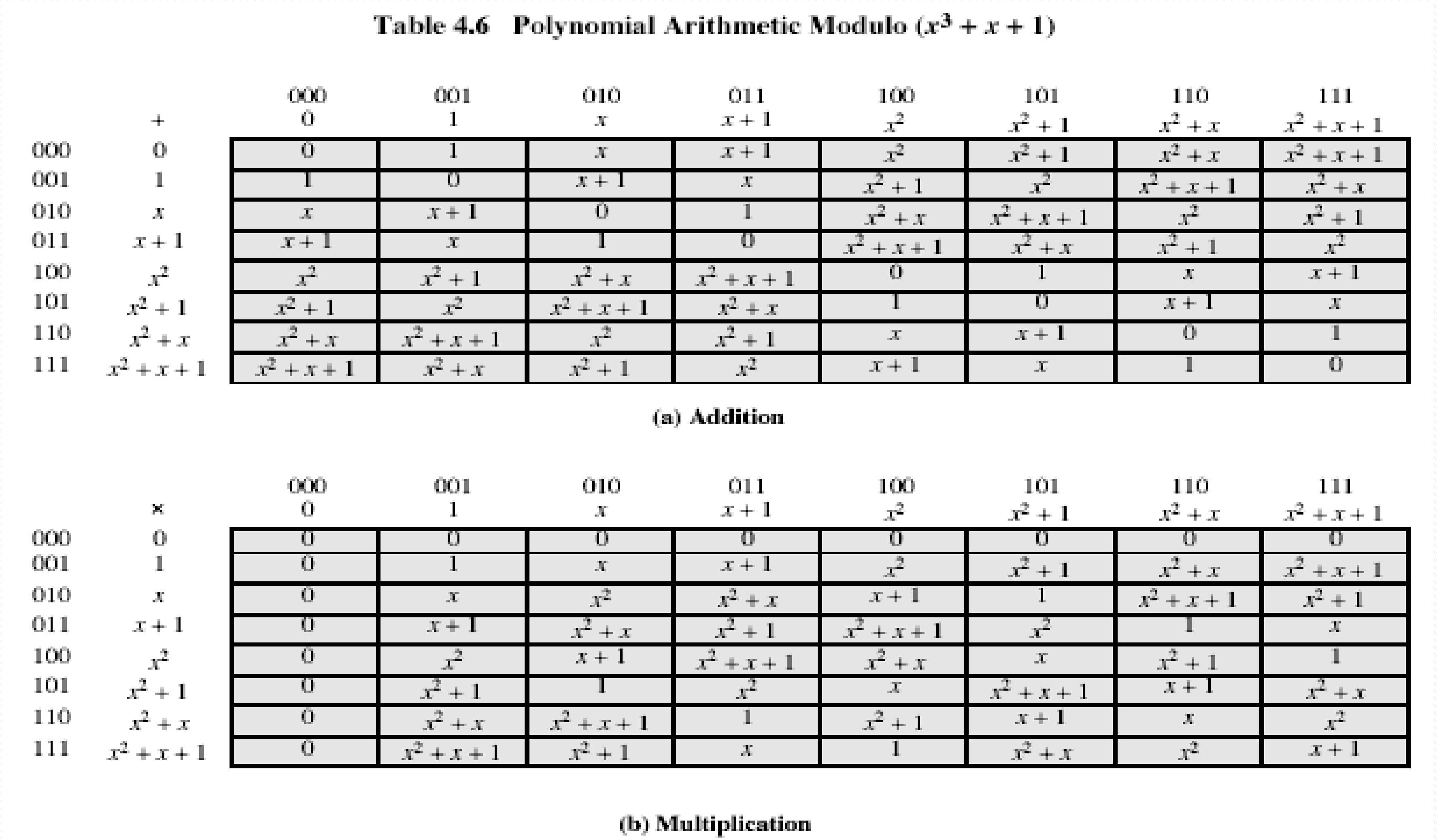

Table 4.6 Polynomial Arithmetic Modulo ($x^3 + x + 1$)

		000	001	010	011	100	101	110	111
	+	0	1	x	$x+1$	x^2	x^2+1	x^2+x	x^2+x+1
000	0	0	1	x	$x+1$	x^2	x^2+1	x^2+x	x^2+x+1
001	1	1	0	$x+1$	x	x^2+1	x^2	x^2+x+1	x^2+x
010	x	x	$x+1$	0	1	x^2+x	x^2+x+1	x^2	x^2+1
011	$x+1$	$x+1$	x	1	0	x^2+x+1	x^2+x	x^2+1	x^2
100	x^2	x^2	x^2+1	x^2+x	x^2+x+1	0	1	x	$x+1$
101	x^2+1	x^2+1	x^2	x^2+x+1	x^2+x	1	0	$x+1$	x
110	x^2+x	x^2+x	x^2+x+1	x^2	x^2+1	x	$x+1$	0	1
111	x^2+x+1	x^2+x+1	x^2+x	x^2+1	x^2	$x+1$	x	1	0

(a) Addition

		000	001	010	011	100	101	110	111
	×	0	1	x	$x+1$	x^2	x^2+1	x^2+x	x^2+x+1
000	0	0	0	0	0	0	0	0	0
001	1	0	1	x	$x+1$	x^2	x^2+1	x^2+x	x^2+x+1
010	x	0	x	x^2	x^2+x	$x+1$	1	x^2+x+1	x^2+1
011	$x+1$	0	$x+1$	x^2+x	x^2+1	x^2+x+1	x^2	1	x
100	x^2	0	x^2	$x+1$	x^2+x+1	x^2+x	x	x^2+1	1
101	x^2+1	0	x^2+1	1	x^2	x	x^2+x+1	$x+1$	x^2+x
110	x^2+x	0	x^2+x	x^2+x+1	1	x^2+1	$x+1$	x	x^2
111	x^2+x+1	0	x^2+x+1	x^2+1	x	1	x^2+x	x^2	$x+1$

(b) Multiplication

Computational Considerations

- since coefficients are 0 or 1, can represent any such polynomial as a bit string
- addition becomes XOR of these bit strings
- multiplication is shift & XOR
  - long-hand multiplication
- modulo reduction done by repeatedly substituting highest power with remainder of irreducible poly (also shift & XOR)

Computational Example

- in GF(2^3) have (x^2+1) is 101_2 & (x^2+x+1) is 111_2
- so addition is
  - $(x^2+1) + (x^2+x+1) = x$
  - 101 XOR 111 = 010_2
- and multiplication is
  - $(x+1).(x^2+1) = x.(x^2+1) + 1.(x^2+1)$

    $= x^3+x+x^2+1 = x^3+x^2+x+1$
  - 011.101 = 1111_2
- polynomial modulo reduction (get q(x) & r(x)) is
  - $(x^3+x^2+x+1\ )$ mod $(x^3+x+1) = 1.(x^3+x+1) + (x^2) = x^2$
  - 1111 mod 1011 = 1111 XOR 1011 = 0100_2

Using a Generator

- equivalent definition of a finite field
- a **generator** g is an element whose powers generate all non-zero elements
  - in F have 0, g^0, g^1, …, g^{q-2}
- can create generator from **root** of the irreducible polynomial
- then implement multiplication by adding exponents of generator