Mathematical Background I

Computer and Network Security

Emilio Coppa

Modular Arithmetic

Goal: computations in finite sets

Most crypto algorithms are based on finite sets of discrete (integer) numbers: this makes it possible to study interesting properties, prove correctness, etc. Modular arithmetic is an approach for reasoning on finite sets. We use modular arithmetic even in our everyday life.

E.g., Hours are a finite set {0, 1, ..., 23}.

"let's meet in 20 hours" We actually mean: (now() + 20) mod 24

For instance, (8 + 20) mod 24 = 28 mod 24 = 4 mod 24

For alphabet is
26

Other examples: weekdays, months, etc.

Modulo operator

Let $a, m, r \in \mathbb{Z}$ with $m > 0$: $a \equiv r \bmod m$

m divides (a - r), m is called modulus, r is called remainder.

Additionally, we can write: $a = q \cdot m + r$ where q is called quotient.

We'd choose

Remark. The remainder is not unique:

$$12 \equiv 3 \bmod 9 \qquad q = 1$$

$$12 \equiv 21 \bmod 9 \qquad q = -1$$

$$12 \equiv -6 \bmod 9 \qquad q = 2$$

By convention, we choose always the smallest non negative remainder. The sets of all remainders form an “equivalence class” modulo m, meaning that they behave equivalent modulo m.

Equivalence classes

For instance, let's consider all the equivalence classes of modulo 5:

$$\{\ldots, -10, -5, 0, 5, 10, \ldots\}$$
$$\{\ldots, -9, -4, 1, 6, 11, \ldots\}$$
$$\{\ldots, -8, -3, 2, 7, 12, \ldots\}$$
$$\{\ldots, -7, -2, 3, 8, 13, \ldots\}$$
$$\{\ldots, -6, -1, 4, 9, 14, \ldots\}$$

Each equivalence class has an infinite number of integers but does not contain all the integers. However, we have "reduced" all integers numbers to just five elements {0, 1, 2, 3, 4}. Our computations will basically involve only this five numbers when working modulo 5. When working with a number different from {0, 1, 2, 3, 4}, you can just replace it with its equivalent number.

Congruence

Two natural number a and b are said to be congruent modulo n if:

$$a \equiv b \bmod n$$

The integer division of *a and n* and *b and n* yield the same remainder. The congruence relation is reflexive, symmetric and transitive, hence it is an equivalence relation.

Properties:

- invariance over addition: $a \equiv b \bmod n \Leftrightarrow (a + c) \equiv (b + c) \bmod n$
- invariance over multiplication: $a \equiv b \bmod n \Leftrightarrow (a \cdot c) \equiv (b \cdot c) \bmod n$
- invariance over exponentiation: $a \equiv b \bmod n \Leftrightarrow a^k \equiv b^k \bmod n$

Congruence (2)

Ex. Some congruences modulo 3:

$$7 \equiv 1 \bmod 3$$

$$-8 \equiv 1 \bmod 3$$

$$-2 \equiv 1 \bmod 3$$

$$7 \equiv -8 \bmod 3$$

$$-2 \equiv 7 \bmod 3$$

Remark. To compute:

x mod m

where x < 0, just add m until you get a positive number, then compute the modulo.

Other way to look at this:

... 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 ...

... -9 -8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9 10 11 12 ...

Properties of modular arithmetic

Modular reduction can be performed at any point during a calculation: e.g., $3^8 \bmod 7$

- First approach (hard): exponentiation followed by modular reduction

$$3^8 = 6561 \equiv 2 \bmod 7$$

- Second approach (easy): exponentiation with intermediate modular reduction

$$3^8 = 3^4 \cdot 3^4 = 81 \cdot 81 \equiv 4 \cdot 4 \bmod 7 = 16 \bmod 7 \equiv 2 \bmod 7$$

Modular division

$$b/a \equiv b \cdot a^{-1} \bmod m$$

The inverse of a is defined as:

$$a \cdot a^{-1} \equiv 1 \bmod m$$

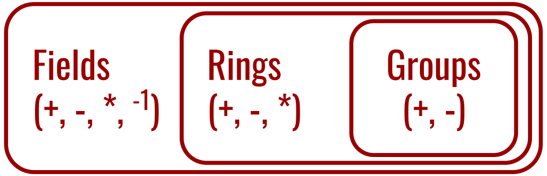
E.g., $5/7 \equiv 5 \cdot 4 = 20 \equiv 2 \bmod 9 \text{ since } 7 \cdot 4 = 28 \equiv 1 \bmod 9$

The inverse of a number a mod m exists only if:

$$gcd(a, m) = 1$$

To compute the greatest common divisor (gcd), we can use the Euclidean algorithm.

Algebraic structures



Fields
(+, -, *, $^{-1}$)

Rings
(+, -, *)

Groups
(+, -)

Given a set:

- group: addition and its inverse
- ring: addition, additive inverse, multiplication
- field: addition, additive inverse, multiplication, multiplicative inverse

Informally, a field is a structure in which we can always add, subtract, multiply, and divide.

Group

A group G is a set of elements and one group operator “○” such that:

1. closure: $\forall a, b \in \mathrm{G} \, : a \circ b \in \mathrm{G}$
2. associativity $\forall a, b, c \in \mathrm{G} \, : (a \circ b) \circ c = a \circ (b \circ c)$
3. neutral element $\exists \, 1 \in \mathrm{G}$ such that $\forall a \in \mathrm{G} \, : a \circ 1 = a$
4. inverse element $\forall a \in \mathrm{G} \, , \exists \, a^{-1} \in \mathrm{G} \, : a \circ a^{-1} = 1$

If abelian group then also:

5. commutative $\forall a, b \in \mathrm{G} \, : a \circ b = b \circ a$

The group operator is often referred, by convention, with the operator “+”.
If we add another group operator, by convention, we refer to it with the operator “*”.

Ring

A ring R is a set with two operations “addition” and “multiplication” such that:

$$\forall a, b, c, d \in \mathrm{R} \;:\; \begin{array}{c} a + b = c \in \mathrm{R} \\ a \cdot b = d \in \mathrm{R} \end{array}$$

Hence, there is closure (the result is in the ring). Additionally:

- addition and multiplication are associative
- addition is commutative
- distributive law holds
- there exists a neutral element for addition
- there exists always an additive inverse for addition given an element in the ring
- there exists a neutral element for multipl.
- the multiplicative inverse exists only for some elements but not for all.

Ring: properties

- abelian group under addition:
  - closure: $\forall a, b \in \mathrm{R} : a + b \in \mathrm{R}$
  - addition is associative $\forall a, b, c \in \mathrm{R} : (a + b) + c = a + (b + c)$
  - addition is commutative $\forall a, b \in \mathrm{R} : a + b = b + a$
  - neutral element for addition $\exists 0 \in \mathrm{R}$ such that $\forall a \in \mathrm{R} : a + 0 = a$
  - inverse for addition $\forall a \in \mathrm{R}, \exists -a \in \mathrm{R} : a + (-a) = 0$
- monoid group under multiplication:
  - closure $\forall a, b \in \mathrm{R} : a \cdot b \in \mathrm{R}$
  - multiplication is associative $\forall a, b, c \in \mathrm{R} : (a \cdot b) \cdot c = a \cdot (b \cdot c)$
  - multiplication is commutative $\forall a, b \in \mathrm{R} : a \cdot b = b \cdot a$
  - distribution law holds $\forall a, b, c \in \mathrm{R} : a \cdot (b + c) = a \cdot b + a \cdot c$
  - neutral element for multiplication $\exists 1 \in \mathrm{R}$ such that $\forall a \in \mathrm{R} : a \cdot 1 = a$

Informally, a ring is a structure in which we can always add, subtract and multiply, but we can only divide by certain elements (namely by those for which a multiplicative inverse exists).

Coprime or relatively prime

An element a has a multiplicative inverse if and only if:

$$gcd(a, m) = 1$$

We say that a is coprime or relatively prime to m.

Finding the inverse is not easy (some effort is required, e.g., Extended Euclidean Algorithm): the set is finite and contain only integers (having real numbers make it easier).

However it is easy see if the inverse of an element exists or not: check the gcd.

Integer ring

We define the “integer ring” $\mathbb{Z}_m = \{0, 1, \ldots, m-1\}$ where we perform modular arithmetic by m:

E.g., $\mathbb{Z}_9 = \{0, 1, 2, 3, 4, 5, 6, 7, 8\}$

Then {0, 3, 6} do not have a multiplicative inverse since they are not coprime (i.e., the gcd is not 1). While:

$$1^{-1} \equiv 1 \bmod 9 \qquad 5^{-1} \equiv 2 \bmod 9$$
$$2^{-1} \equiv 5 \bmod 9 \qquad 7^{-1} \equiv 4 \bmod 9$$
$$4^{-1} \equiv 7 \bmod 9 \qquad 8^{-1} \equiv 8 \bmod 9$$

Field

A field F is a ring where every non negative element has a multiplicative inverse in the field:

$$\forall a \in \mathrm{F}, \exists\, a^{-1} \in \mathrm{F} : a \cdot a^{-1} = 1$$

Element 0 does not need to have the inverse.
A finite field (also called Galois Field) is a field where the set F is finite.

E.g., $\mathbb{Z}_9 = \{0, 1, 2, 3, 4, 5, 6, 7, 8\}$ is not a field.

$\mathbb{Z}_5 = \{0, 1, 2, 3, 4\}$ is a field.

Properties of a field

Theorem. Let F be a field, then for any non negative element a the inverse a^{-1} is unique.

Proof by contradiction. Suppose a^{-1} and b are two inverses of a then:

$ab = 1$
$a^{-1} ab = 1 a^{-1}$
$(a^{-1} a)b = 1 a^{-1}$
$1 b = 1 a^{-1}$
$b = a^{-1}$ (hence the two inverses must be equal and cannot be different)

This property can be exploited in cryptography to implement a scheme based on the relation between an element and its inverse.

Finite fields (or Galois Fields)

Theorem. It can be proved that $\mathbb{Z}_p$ is a field if p is prime.
Theorem. Finite fields exist only if they have p^m elements, where p is a prime and m an integer.

E..g.,

There exists a finite field with 2 elements (called GF(2)): p = 2, m = 1
There exists a finite field with 11 elements (called GF(11)): p = 11, m = 1
There is NOT a finite field with 12 elements: 12 = 3 * 4 = 3 * 2^2
There exists a finite field with 81 elements (called GF(81) = GF(3^4)): p = 3, m = 4
There exists a finite field with 256 elements (called GF(256) = GF(2^8)): p = 2, m = 8

GF(2) is very common, GF(2^8) is used by AES.

Types of Galois Fields

Two types of $GF(p^m)$:

- $m = 1$: prime fields, written as $GF(p)$
- $m > 1$: extension fields

They are both used extensively in cryptography. In particular, Galois Fields with $p=2$, i.e., $GF(2^m)$, are very common and well studied.

Prime Field Arithmetic

The elements of a prime field GF(p) are the integers {0, 1, 2, ..., p-1}

- Addition: $a + b \equiv c \bmod p$
- Subtraction: $a - b \equiv d \bmod p$
- Multiplication: $a \cdot b \equiv e \bmod p$

Hence, for {addition, subtraction, multiplication} is just the operation modulo p. Properties of the ring are satisfied when performing arithmetic in this way.

Prime Field Arithmetic (2)

- division (multiplicative inverse):

$$\forall a \in GF(p) : \exists\, a^{-1} : a \cdot a^{-1} \equiv 1 \bmod p$$

The multiplicative inverse must exist (except for element 0). To compute it, we need to use the Extended Euclidean Algorithm (see later).

Extension field GF(2^m) Arithmetic

The elements of an extension field GF(2^m) are not integers but polynomials:

$$a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + \ldots + a_1 x + a_0 = A(x) \in GF(2^m)$$

We care for the coefficients of the polynomial. For AES, since GF(2^8) we have 8 coefficients for each polynomial. Each coefficient:

$$a_i \in GF(2)$$

Since GF(2) is a prime field, it contains only {0, 1}, i.e., bit valued. Notice that AES is based on GF(2^8) since each byte has 8 bits, which can represent 8 coefficients. GF(2^8) = GF(256) but also with a byte we can represent 256 values.

Extension field GF(2^m): an example

E.g., Let’s consider GF(2^3):

$$A(x) = a_2x^2 + a_1x + a_0$$

Which are the elements of GF(2^3)?

$$\{0, 1, x, 1 + x, x^2, 1 + x^2, x + x^2, 1 + x + x^2\}$$

We can generate them by consider all the possible combination of three bits:

$$\{000_2, 001_2, 010_2, 011_2, 100_2, 101_2, 110_2, 111_2\}$$

Extension field GF(2^m) Arithmetic (2)

- Addition: $C(x) = A(x) + B(x) = \sum_{i=0}^{m-1} c_i\, x^i \text{ where } c_i \equiv a_i + b_i \bmod 2$

- Subtraction: $C(x) = A(x) - B(x) = \sum_{i=0}^{m-1} c_i\, x^i \text{ where } c_i \equiv a_i - b_i \bmod 2$

E.g.,

$$A(x) = 1x^2 + 1x + 1$$
$$B(x) = 1x^2 + 0x + 1$$
$$A(x) + B(x) = (1+1)x^2 + (1+0)x + (1+1)$$
$$A(x) + B(x) = 0x^2 + 1x + 0 = x$$

Remarks. Addition and subtraction in modulo 2 are the same operation.
Addition modulo 2 is equal to a bitwise XOR operation.

Extension field GF(2^m) Arithmetic (3)

- Multiplication:

E.g.,

$$A(x) = 1x^2 + 1x + 1 = x^2 + x + 1$$
$$B(x) = 1x^2 + 0x + 1 = x^2 + 1$$

$$A(x) \cdot B(x) = (x^2 + x + 1)(x^2 + 1) = x^4 + x^3 + x^2 + x^2 + x + 1$$
$$A(x) \cdot B(x) = x^4 + x^3 + (1 + 1)x^2 + x + 1$$
$$A(x) \cdot B(x) = x^4 + x^3 + 0x^2 + x + 1 = x^4 + x^3 + x + 1 = C'(x)$$

Problem: x^4 and x^3 are not in GF(2^3)! In prime fields, when we get something that is not in the field, we perform modulo to compute to obtain the remainder that is in the field. Here is the same but more painful: we need to compute the remainder of a polynomial.

Extension field GF(2^m) Arithmetic (4)

Similarly to prime fields, we need to perform the modulo reduction on C'(x) against a "prime" polynomial, i.e., irreducible polynomial. Similarly to "primes", these polynomial cannot be factorized. For instance, a irreducible polynomial for GF(2^3):

$$P(x) = x^3 + x + 1$$

Hence, multiplication:

$$C(x) = A(x) \cdot B(x) \bmod P(x)$$

In our example:

$$\begin{array}{rlllll} & (x^4 + & x^3 + & & x + 1) & : (x^3 + x + 1) = x \\ - & (x^4 + & & x^2 + & x) & \\ \hline & & x^3 + & x^2 + & 1 & \end{array}$$

Still not in GF(2^3)

Extension field GF(2^m) Arithmetic (5)

We continue with the division:

$$\begin{array}{rl} & (x^3 + x^2 + 1) : (x^3 + x + 1) = x + 1 \\ - & (x^3 + x + 1) \\ \hline & x^2 + x \end{array}$$

This is in GF(2^3). We are done: this is C(x)!

Remark. For every field GF(2^m), there are several irreducible polynomials. E.g., for GF(2^3) another one is:

$$P(x) = x^3 + x^2 + 1$$

Remark. The result of the computation depends on the specific P(x) that is used. AES uses one specific irreducible polynomial and this is dictated by the standard.

Extension field GF(2^m) Arithmetic (5)

- Multiplicative inverse:

$$A(x) \cdot A(x)^{-1} \equiv 1 \bmod P(x)$$

How do we find the multiplicative inverse? Extended Euclidean Algorithm

Euclidean Algorithm

Goal: compute the greatest common divisor gcd(r_0, r_1)

Easy for small numbers:

- factor r_0 and r_1
- take the highest common factor

E.g. $r_0 = 84 = \underline{2} * 2 * \underline{3} * 7$

$r_1 = 30 = \underline{2} * \underline{3} * 5$

$\gcd(r_0, r_1) = \gcd(84, 30) = 6$

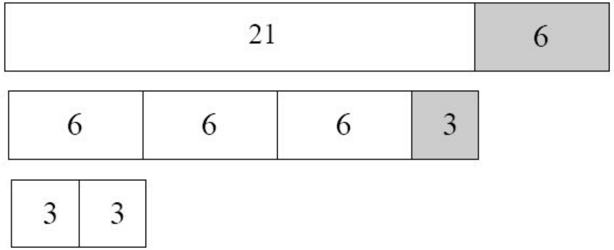
Chapter 6 of Understanding Cryptography by Christof Paar and Jan Pelzl

Euclidean Algorithm (2)

Observation: $\gcd(r_0, r_1) = \gcd(r_0 - r_1, r_1)$ and more in general $\gcd(r_0, r_1) = \gcd(r_0 \bmod r_1, r_1)$

Idea:

- reduce the problem of finding the gcd of two given numbers to finding the gcd of two smaller numbers
- repeat this process recursively
- stop when $r_0 - r_1$ is zero



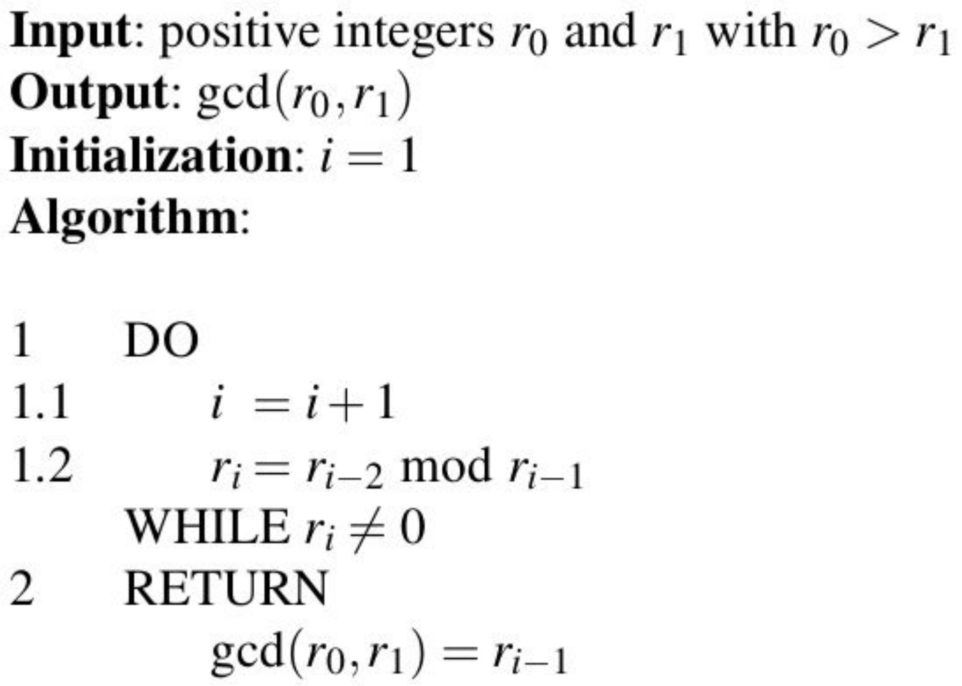
gcd(27, 21) = gcd(1*21+6, 21) = gcd(6, 21) = gcd(21, 6)

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

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

Chapter 6 of Understanding Cryptography by Christof Paar and Jan Pelzl

Euclidean Algorithm (3)



Input: positive integers r_0 and r_1 with $r_0 > r_1$
Output: $\gcd(r_0, r_1)$
Initialization: $i = 1$
Algorithm:

```
1      DO
1.1        $i = i + 1$
1.2        $r_i = r_{i-2} \bmod r_{i-1}$
       WHILE $r_i \neq 0$
2      RETURN
           $\gcd(r_0, r_1) = r_{i-1}$
```

Chapter 6 of Understanding Cryptography by Christof Paar and Jan Pelzl

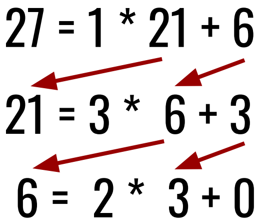
Euclidean Algorithm (4)

We can write as:

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) = 3



27 = 1 * 21 + 6

21 = 3 * 6 + 3

6 = 2 * 3 + 0

Notice that each step we left shift

Euclidean Algorithm (5)

E.g., gcd(973, 301) = ?

$$\overset{r_0}{973} = 3 * \overset{r_1}{301} + \overset{r_2}{70}$$

$$301 = 4 * 70 + \overset{r_3}{21}$$

$$70 = 3 * 21 + \overset{r_4}{7}$$

$$21 = 3 * 7 + 0$$

gcd(973, 301) = 7

Extended Euclidean Algorithm (EEA)

Goal: compute the greatest common divisor gcd(r_0, r_1) and the multiplicative inverse when gcd(r_0, r_1)=1

It computes: $gcd(r_0, r_1) = s \cdot r_0 + t \cdot r_1$ (Bezout's Identity, an example of linear Diophantine equation)

where: $gcd(r_0, r_1) = s \cdot r_0 + t \cdot r_1 = 1$ (existence of multiplicative inverse)

$$0 \cdot r_0 + t \cdot r_1 \equiv 1 \bmod r_0$$

$$t \cdot r_1 \equiv 1 \bmod r_0$$ (t is the inverse that we want to find)

EEA computes s and t. We need t when we are looking for the inverse.

Chapter 6 of Understanding Cryptography by Christof Paar and Jan Pelzl

Extended Euclidean Algorithm (2)

Three main steps:

1. Compute $gcd(r_0, r_1)$ using EA
2. Compute $r_0 = q_1 \cdot r_1 + r_2$
3. Compute $r_2 = s_2 \cdot r_0 + t_2 \cdot r_1$

Repeat this process recursively.

Extended Euclidean Algorithm (3)

$$\begin{array}{lll}
gcd(r_0, r_1) & r_0 = q_1 \cdot r_1 + r_2 & r_2 = s_2 \cdot r_0 + t_2 \cdot r_1 \\
gcd(r_1, r_2) & r_1 = q_2 \cdot r_2 + r_3 & r_3 = s_3 \cdot r_0 + t_3 \cdot r_1 \\
\vdots & \vdots & \vdots \\
gcd(r_{l-2}, r_{l-1}) & r_{l-2} = q_{l-1} \cdot r_{l-1} + r_l & r_l = s_l \cdot r_0 + t_l \cdot r_1
\end{array}$$

r_l is the $gcd(r_0, r_1)$

$t_l = t$ is the multiplicative inverse iff r_l is 1

Extended Euclidean Algorithm (4)

E.g., $gcd(973, 301) = s \cdot 973 + t \cdot 301$

$$i = 2 \qquad \overset{r_0}{973} = 3 \cdot \overset{r_1}{301} + \overset{r_2}{70} \qquad\qquad r_2 = 70 = 1 \cdot 973 + (-3) \cdot 301$$

$$i = 3 \qquad 301 = 4 \cdot 70 + \overset{r_3}{21} \qquad\qquad r_3 = 21 = 1 \cdot 301 + (-4) \cdot 70$$

This is not in the form: $s \cdot 973 + t \cdot 301$

but we can replace the value of 70 with what obtained in the previous line:

$$r_3 = 1 \cdot 301 + (-4) \cdot (1 \cdot 973 - 3 \cdot 301)$$

This is what we need: $r_3 = -4 \cdot 973 + 13 \cdot 301$

Extended Euclidean Algorithm (5)

$$i = 4 \qquad 70 = 3 \cdot 21 + \overset{r_4}{7}$$

$$r_4 = 7 = 1 \cdot 70 + (-3) \cdot 21$$

This is not in the form: $s \cdot 973 + t \cdot 301$

but we can replace the values of 70 and 21 with the what obtained in the previous lines:

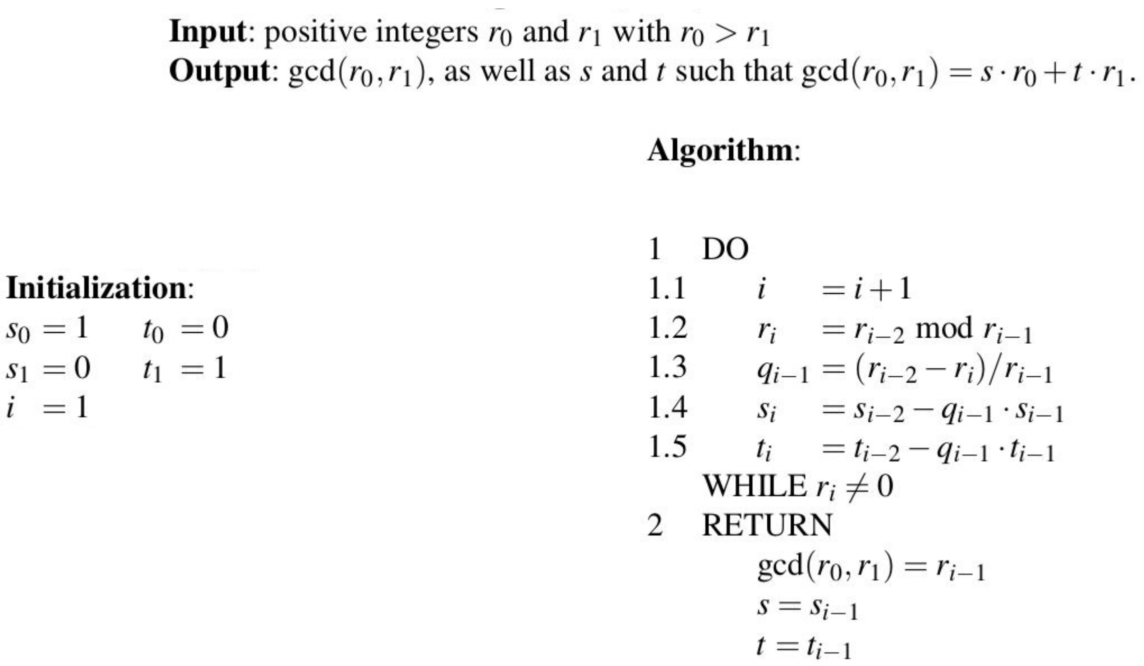
$$r_4 = 1 \cdot (973 - 3 \cdot 301) + (-3) \cdot (-4 \cdot 973 + 13 \cdot 301)$$
$$r_4 = 13 \cdot 973 + (-42) \cdot 301$$

Since r_5 will be zero, we have done: $t_4 = t = -42$

Remark. Since gcd(973, 301) != 1 then t is not the multiplicative inverse of 973 modulo 301.

If we generalize this process...

Extended Euclidean Algorithm (6)



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 $\gcd(r_0, r_1) = s \cdot r_0 + t \cdot r_1$.

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      $s_i = s_{i-2} - q_{i-1} \cdot s_{i-1}$
1.5      $t_i = t_{i-2} - q_{i-1} \cdot t_{i-1}$
     WHILE $r_i \neq 0$
2    RETURN
         $\gcd(r_0, r_1) = r_{i-1}$
         $s = s_{i-1}$
         $t = t_{i-1}$
```

Chapter 6 of Understanding Cryptography by Christof Paar and Jan Pelzl

Extended Euclidean Algorithm (7)

Main application of EEA is computing the inverses modulo n.

$$a \cdot a^{-1} \equiv 1 \bmod n$$

$$gcd(n, a) = 1 = s \cdot n + t \cdot a \equiv t \cdot a \bmod n$$

existence condition of inverse

EEA

inverse

Extended Euclidean Algorithm (8)

For polynomial in GF(2^8), EEA works similarly. However, s and t must be replaced by two polynomials s(x) and t(x), where t(x) will be the inverse that we are looking for.

$$gcd(P(x), A(x)) = 1 = s(x) \cdot P(x) + t(x) \cdot A(x)$$

If we take the modulo:

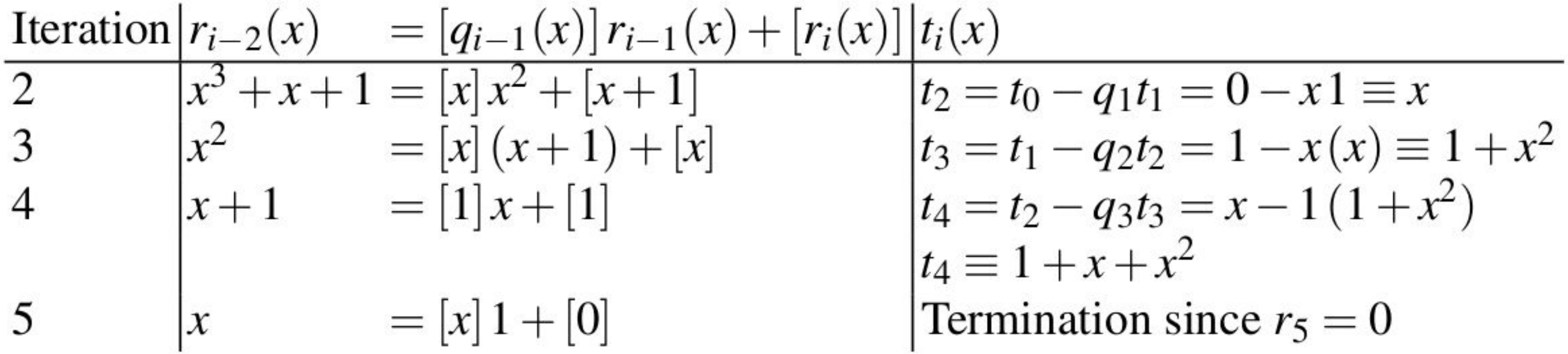
$$s(x) \cdot 0 + t(x) \cdot A(x) \equiv 1 \bmod P(x)$$

$$t(x) = A(x)^{-1} \bmod P(x)$$

Chapter 6 of Understanding Cryptography by Christof Paar and Jan Pelzl

Extended Euclidean Algorithm (9)

E.g., Inverse of $A(x) = x^2$ in GF(2^3) with $P(x) = x^3 + x + 1$



Iteration	$r_{i-2}(x)$	$= [q_{i-1}(x)]\, r_{i-1}(x) + [r_i(x)]$	$t_i(x)$
2	$x^3 + x + 1$	$= [x]\, x^2 + [x+1]$	$t_2 = t_0 - q_1 t_1 = 0 - x1 \equiv x$
3	x^2	$= [x]\,(x+1) + [x]$	$t_3 = t_1 - q_2 t_2 = 1 - x(x) \equiv 1 + x^2$
4	$x+1$	$= [1]\, x + [1]$	$t_4 = t_2 - q_3 t_3 = x - 1\,(1 + x^2)$
			$t_4 \equiv 1 + x + x^2$
5	x	$= [x]\, 1 + [0]$	Termination since $r_5 = 0$

Initially, we assume: $t_0(x) = 1,\ t_1(x) = 0$

Chapter 6 of Understanding Cryptography by Christof Paar and Jan Pelzl

Extension field GF(2^m) Arithmetic (6)

Remark. For a small size finite field, a (precomputed) lookup table is the most efficient method for implementing multiplication and for finding the multiplicative inverse.

Credits

These slides are based on material from:

- Slides of Prof. D’Amore from CNS 2019-2020
- Christof Paar and Jan Pelzl. Understanding Cryptography: A Textbook for Students and Practitioners. Springer. http://www.crypto-textbook.com/
- Wikipedia (english version)