15-499: Algorithms and Applications

Cryptography II
- Number theory (groups and fields)

Cryptography Outline

Introduction: terminology and background
Primitives: one-way hash functions, trapdoors, ...
Protocols: digital signatures, key exchange, ...
Number Theory: groups, fields, ...
Private-Key Algorithms: Rijndael, DES, RC4
Cryptanalysis: Differential, Linear
Public-Key Algorithms: Knapsack, RSA, El-Gamal,
Blum-Goldwasser
Case Studies: Kerberos, Digital Cash

Number Theory Outline

Groups
- Definitions, Examples, Properties
- Multiplicative group modulo n
- The Euler-phi function

Fields
- Definition, Examples
- Polynomials
- Galois Fields

Why does number theory play such an important role?

It is the mathematics of finite sets of values.

Groups

A Group \((G,\cdot, I)\) is a set \(G\) with operator \(\cdot\) such that:
1. Closure. For all \(a, b \in G\), \(a \cdot b \in G\)
2. Associativity. For all \(a, b, c \in G\), \(a \cdot (b \cdot c) = (a \cdot b) \cdot c\)
3. Identity. There exists \(I \in G\), such that for all \(a \in G\), \(a \cdot I = I \cdot a = a\)
4. Inverse. For every \(a \in G\), there exist a unique element \(b \in G\), such that \(a \cdot b = b \cdot a = I\)

An Abelian or Commutative Group is a Group with the additional condition
5. Commutativity. For all \(a, b \in G\), \(a \cdot b = b \cdot a\)
### Examples of groups
- Integers, Reals or Rationals with Addition
- The nonzero Reals or Rationals with Multiplication
- Non-singular \( n \times n \) real matrices with Matrix Multiplication
- Permutations over \( n \) elements with composition \([0\rightarrow 1, 1\rightarrow 2, 2\rightarrow 0] \circ [0\rightarrow 1, 1\rightarrow 0, 2\rightarrow 2] = [0\rightarrow 0, 1\rightarrow 2, 2\rightarrow 1]\)

We will only be concerned with finite groups, i.e., ones with a finite number of elements.

### Groups based on modular arithmetic
The group of positive integers modulo a prime \( p \)
\( \mathbb{Z}_p^* = \{1, 2, 3, \ldots, p-1\} \)
\( \ast_p = \text{multiplication modulo } p \)
Denoted as: \((\mathbb{Z}_p, \ast_p)\)

**Required properties**
3. Identity. 1.
4. Inverse. Yes.

**Example:** \( \mathbb{Z}_7 = \{1, 2, 3, 4, 5, 6\} \)
\( 1^{-1} = 1, 2^{-1} = 4, 3^{-1} = 5, 6^{-1} = 6 \)

### Key properties of finite groups
**Notation:** \( a^j = a \ast a \ast a \ast \ldots \ast j \text{ times} \)

**Theorem (Fermat's little):** for any finite group \((G, \ast, I)\) and \( g \in G, g^{|G|} = I \)

**Definition:** the order of \( g \in G \) is the smallest positive integer \( m \) such that \( g^m = I \)

**Definition:** a group \( G \) is cyclic if there is a \( g \in G \) such that \( \text{order}(g) = |G| \)

**Definition:** an element \( g \in G \) of order \( |G| \) is called a generator or primitive element of \( G \).

### Other properties
\( |\mathbb{Z}_p^*| = (p-1) \)
By Fermat’s little theorem: \( a^{(p-1)} = 1 \text{ (mod } p) \)

**Example of \( \mathbb{Z}_7 \)**

<table>
<thead>
<tr>
<th>( x )</th>
<th>( x^2 )</th>
<th>( x^3 )</th>
<th>( x^4 )</th>
<th>( x^5 )</th>
<th>( x^6 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>6</td>
<td>4</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>6</td>
<td>1</td>
<td>6</td>
<td>1</td>
</tr>
</tbody>
</table>

**Generators**

For all \( p \) the group is cyclic.
What if n is not a prime?
The group of positive integers modulo a non-prime n
\( \mathbb{Z}_n = \{1, 2, 3, ..., n-1\} \), n not prime
*\( _p \) = multiplication modulo n
**Required properties?**
1. Closure. ?
2. Associativity. ?
3. Identity. ?
4. Inverse. ?
How do we fix this?

---

Groups based on modular arithmetic
The multiplicative group modulo n
\( \mathbb{Z}^*_n = \{m : 1 \leq m < n, \text{gcd}(n,m) = 1\} \)
*\( _n \) = multiplication modulo n
Denoted as \( (\mathbb{Z}^*_n, \ast) \)
**Required properties:**
- Closure. Yes.
- Associativity. Yes.
- Identity. 1.
- Inverse. Yes.
**Example:** \( \mathbb{Z}^*_6 = \{1, 2, 4, 7, 8, 11, 13, 14\} \)
  \( 1_6 \ast 1 = 1, 2_6 \ast 8 = 4, 4_6 \ast 4 = 7, 7_6 \ast 13 = 11, 11_6 \ast 11 = 14_6 \ast 14 = 14 \)

---

The Euler Phi Function
\( \phi(n) = |\mathbb{Z}^*_n| = n \prod_{p \mid n}(1 - 1/p) \)
If n is a product of two primes p and q, then
\( \phi(n) = pq(1 - 1/p)(1 - 1/q) = (p-1)(q-1) \)
Note that by Fermat's Little Theorem:
\( a^{\phi(n)} = 1 \pmod{n} \) for \( a \in \mathbb{Z}^*_n \)
Or for \( n = pq \)
\( a^{(p-1)(q-1)} = 1 \pmod{n} \) for \( a \in \mathbb{Z}^*_{pq} \)
This will be very important in RSA!

---

Generators
Example of \( \mathbb{Z}^*_10 = \{1, 3, 7, 9\} \)

<table>
<thead>
<tr>
<th>x</th>
<th>x^2</th>
<th>x^3</th>
<th>x^4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>9</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>9</td>
<td>1</td>
</tr>
</tbody>
</table>

For all n the group is cyclic.
**Operations we will need**

**Multiplication**: $a \times b \pmod{n}$
- Can be done in $O(\log^2 n)$ bit operations

**Inverse**: $a^{-1} \pmod{n}$
- Euclid's algorithm $O(\log n)$ steps, $O(\log^3 n)$ bit ops

**Power**: $a^k \pmod{n}$
- The power method $O(\log n)$ steps, $O(\log^3 n)$ bit ops
  
  ```
  fun pow(a, k) =
  if (k = 0) then 1
  else if (k mod 2 = 1)
    then a * (pow(a, k/2))^2
  else (pow(a, k/2))^2
  ```

---

**Euclid's Algorithm**

Euclid's Algorithm:

```math
\gcd(a, b) = \gcd(b, a \mod b) \\
\gcd(a, 0) = a
```

"Extended" Euclid's algorithm:

- Find $x$ and $y$ such that $ax + by = \gcd(a, b)$
- Can be calculated as a side-effect of Euclid's algorithm.
- Note that $x$ and $y$ can be zero or negative.

This allows us to find $a^{-1} \pmod{n}$, for $a \in \mathbb{Z}_n^*$
In particular return $x$ in $ax + ny = 1$.

---

**Euclid's Algorithm**

```math
\text{fun euclid}(a, b) = 
  \text{if } (b = 0) \text{ then } a \text{ else euclid}(b, a \mod b)
```

```math
\text{fun ext_euclid}(a, b) = 
  \text{if } (b = 0) \text{ then } (a, 1, 0) \text{ else} 
  \text{let } (d, x, y) = \text{ext_euclid}(b, a \mod b) \text{ in} 
  (d, y, x - (a/b) y) \end
```

The code is in the form of an inductive proof.

**Exercise**: prove the inductive step.

---

**Discrete Logarithms**

If $g$ is a generator of $\mathbb{Z}_n^*$, then for all $y$ there is a unique $x \pmod{\phi(n)}$ such that

- $y = g^x \mod n$

This is called the **discrete logarithm** of $y$ and we use the notation

- $x = \log_g(y)$

In general finding the discrete logarithm is conjectured to be hard...as hard as factoring.
**Fields**

A **Field** is a set of elements $F$ with binary operators $\ast$ and $+$ such that

1. $(F, +)$ is an **abelian group**
2. $(F \setminus I, \ast)$ is an **abelian group**
3. **Distribution**: $a \ast (b + c) = a \ast b + a \ast c$
4. **Cancellation**: $a \ast I = I$.  

The **order** of a field is the number of elements.

A field of finite order is a **finite field**.

The reals and rationals with $+$ and $\ast$ are fields.

$\mathbb{Z}_p$ (p prime) with $+$ and $\ast \mod p$, is a finite field.

---

**Polynomials over $\mathbb{Z}_p$**

$\mathbb{Z}_p[x]$ = polynomials on $x$ with coefficients in $\mathbb{Z}_p$.

- Example of $\mathbb{Z}_5[x]$: $f(x) = 3x^4 + 1x^3 + 4x^2 + 3$
- $\text{deg}(f(x)) = 4$ (the **degree** of the polynomial)

Operations: (examples over $\mathbb{Z}_5[x]$)

- Addition: $(x^3 + 4x^2 + 3) + (3x^2 + 1) = (x^3 + 2x^2 + 4)$
- Multiplication: $(x^3 + 3) \ast (3x^2 + 1) = 3x^5 + x^3 + 4x^2 + 3$
- $I = 0, I = 1$
- $+$ and $\ast$ are associative and commutative
- Multiplication distributes and 0 cancels

Do these polynomials form a field?

---

**Division and Modulus**

Long division on polynomials ($\mathbb{Z}_5[x]$):

\[
\begin{array}{c|ccccc}
  & x^3 & + & 4x^2 & + & 0x + 3 \\
\hline
x^2 + 1 & \overline{x^3} & + & 4x^2 & + & 0x + 3 \\
        & - & x^3 & + & 0x^2 & + 1x + 0 \\
        & \hline
        & 4x^2 & + & 4x & + & 4 \\
        & - & 4x^2 & + & 0x & + 4 \\
        & \hline
        & 4x & + & 4
\end{array}
\]

$(x^3 + 4x^2 + 3)/(x^2 + 1) = (x + 4)$

$(x^3 + 4x^2 + 3) \mod (x^2 + 1) = (4x + 4)$

$(x^2 + 1)(x + 4) + (4x + 4) = (x^3 + 4x^2 + 3)$

---

**Polynomials modulo Polynomials**

How about making a field of polynomials modulo another polynomial? This is analogous to $\mathbb{Z}_p$ (i.e., integers modulo another integer).

e.g. $\mathbb{Z}_5[x] \mod (x^2+2x+1)$

Does this work?

Does $(x + 1)$ have an inverse?

**Definition**: An **irreducible polynomial** is one that is not a product of two other polynomials both of degree greater than 0.

e.g. $(x^2 + 2)$ for $\mathbb{Z}_5[x]$  

Analogous to a prime number.
**Galois Fields**

The polynomials
\[ Z_p[x] \mod p(x) \]
where
\[ p(x) \in Z_p[x], \]
\[ p(x) \text{ is irreducible,} \]
\[ \text{and } \deg(p(x)) = n \]
form a finite field. Such a field has \( p^n \) elements. These fields are called **Galois Fields** or \( GF(p^n) \).
The special case \( n = 1 \) reduces to the fields \( Z_p \).
The multiplicative group of \( GF(p^n)/(0) \) is cyclic (this will be important later).

**GF(2^n)**

Hugely practical!
The coefficients are bits \((0,1)\).
For example, the elements of \( GF(2^8) \) can be represented as a byte, one bit for each term, and \( GF(2^{64}) \) as a 64-bit word.
- e.g. \( x^6 + x^4 + x + 1 = 01010011 \)
How do we do addition?
**Addition** over \( Z_2 \) corresponds to xor.
- Just take the xor of the bit-strings (bytes or words in practice). This is dirt cheap

**Multiplication over GF(2^n)**

If \( n \) is small enough can use a table of all combinations.
The size will be \( 2^n \times 2^n \) (e.g. 64K for \( GF(2^8) \)).
Otherwise, use standard shift and add (xor)

**Note:** dividing through by the irreducible polynomial on an overflow by 1 term is simply a test and an xor.

\[ \begin{align*}
\text{e.g. } 0111 / 1001 &= 0111 \\
1011 / 1001 &= 1011 \text{ xor } 1001 = 0010 \\
&\text{^ just look at this bit for } GF(2^3)
\end{align*} \]

**Typedef signed char uc;**

\[
\begin{align*}
typedef &\text{ unsigned char uc;} \\
u &\text{ mult(uc a, uc b) } \{ \\
&\text{ int p = a; } \\
&\text{ uc r = 0; } \\
&\text{ while(b) } \{ \\
&\text{ if (b & 1) r = r ^ p; } \\
&\text{ b = b >> 1; } \\
&\text{ p = p << 1; } \\
&\text{ if (p & 0x100) p = p ^ 0x11B; } \\
&\text{ } \}
&\text{ return r; } \\
&\} \\
\end{align*}
\]
Finding inverses over \( GF(2^n) \)

Again, if \( n \) is small just store in a table.
- Table size is just \( 2^n \).
For larger \( n \), use Euclid’s algorithm.
- This is again easy to do with shift and xors.

Polynomials with coefficients in \( GF(p^n) \)

Recall that \( GF(p^n) \) were defined in terms of coefficients that were themselves fields (i.e., \( Z_p \)).
We can apply this recursively and define:

\[ GF(p^n)[x] = \text{polynomials on} \ x \text{ with coefficients in} \ GF(p^n). \]
- Example of \( GF(2^3)[x] \): \( f(x) = 001x^2 + 101x + 010 \)
  Where 101 is shorthand for \( x^2+1 \).

Polynomials with coefficients in \( GF(p^n) \)

We can make a finite field by using an irreducible polynomial \( M(x) \) selected from \( GF(p^n)[x] \).
For an order \( m \) polynomial and by abuse of notation we can write: \( GF(GF(p^n)^m) \), which has \( p^{nm} \) elements.
Used in Reed-Solomon codes and Rijndael.
- In Rijndael \( p=2, n=8, m=4 \), i.e. each coefficient is a byte, and each element is a 4 byte word (32 bits).
Note: all finite fields are isomorphic to \( GF(p^n) \), so this is really just another representation of \( GF(2^{32}) \).
This representation, however, has practical advantages.