EE 480F

Review Summary

May 5, 2005

1 Modular Arithmetic

1.1 Groups

A Group is any set G with some operation defined on them, such that these four rules hold:

1. *Closure:* if $g \in G$ and $h \in G$, then $gh \in G$.
2. *Associativity:* For $f, g, h \in G$, $f(gh) = (fg)h$.
3. *Identity:* There is an identity $1 \in G$, so that $g1 = 1g = g$.
4. *Inverses:* Every element $g \in G$ has an inverse $g^{-1} \in G$, such that $gg^{-1} = g^{-1}g = 1$.

Examples of groups: the set of real numbers $\mathbb{R}$ under addition; the set of nonzero real numbers $\mathbb{R} - \{0\}$ under multiplication; the set of integers $\mathbb{Z}$ under addition.

1.2 The integers modulo N

Two integers are *congruent modulo* N if their difference is a multiple of N. For example, 22 and 30002 are congruent modulo 10. We write $22 \equiv 30002 \pmod{10}$.

The set $\mathbb{Z}_N$ is a set of N elements $\{0, 1, 2, \cdots, N-1\}$, with addition and multiplication defined modulo N.

$\mathbb{Z}_N$ is a group under addition, but not under multiplication. The set $\mathbb{Z}_N^\times$ is the subset of $\mathbb{Z}_N$ containing all the elements with multiplicative inverses. For example, $\mathbb{Z}_8^\times = \{1, 3, 5, 7\}$. A number a has a multiplicative inverse modulo N, if a and N have no common factors.

1.3 The Euclidean Algorithm

The Euclidean algorithm is used to find the greatest common divisor (GCD) of two numbers N and M. It is also used to find the inverse of one number modulo another. What the algorithm finds is the smallest positive integer K such that this equation:

$$Na + Mb = K$$

...has a solution for some integers a and b. The number K is the GCD of N and M. If the GCD is 1, the extra numbers a and b tell you the inverses: a is the inverse of $N \pmod{M}$; b is the inverse of $M \pmod{N}$.

The algorithm works as follows: start with the system of equations

$$\begin{aligned} x &= N \\ y &= M \end{aligned}$$

Assume $M < N$. Suppose $N = kM + R$: then subtract k times the bottom equation from the top, to get

$$\begin{aligned} y &= M \\ x - ky &= R \end{aligned}$$

...Now $R < M$. Repeat this process for as long as the right side of the equations are nonzero. Eventually you will have an equation of the form $Ax + By = R$. R is the GCD, and if the GCD is 1, then $AN \equiv 1 \pmod{M}$.

2 The RSA algorithm

2.1 Euler's Phi function

The number $\phi(N)$ is the size of $\mathbb{Z}_N^\times$, or the number of integers from $1 \cdots N$ with no factors in common with N. For example, $\mathbb{Z}_{15}^\times = \{1, 2, 4, 7, 8, 11, 13, 14\}$, so $\phi(15) = 8$.

The formula for $\phi(N)$ is

$$\phi(N) = N \cdot \prod_{p|N} (1 - 1/p)$$

The productis taken over every distinct prime p that divides N. Example: $\phi(100) = 100 \cdot (1 - 1/2) \cdot (1 - 1/5) = 40$.

2.2 Euler's Generalization of Fermat's Little Theorem

The theorem is very simple to state: if a, N have no factors in common,

$$a^{\phi(N)} \equiv 1 \pmod{N}$$

This says that as we multiply and exponentiate modulo N, the exponents will behave like an additive group modulo $\phi(N)$.

An example application: what is the last digit of $m = 7^{1234567}$? Well, we want to know $m \pmod{10}$, and the formula tells us $\phi(10) = 4$. $7^4 \equiv 1$, so

$$7^{1234567} = 7^{4 \cdot K + 3} = (7^4)^K \cdot 7^3 \equiv 1^K \cdot 7^3$$

So the last digit is a "3", because $7^3 = 343$.

2.3 The RSA algorithm

Factoring is hard (see below), so for two appropriately chosen primes p and q, we can compute $N = pq$ but nobody can determine p or q from N. We can also compute $\phi(N) = (p-1)(q-1)$, but nobody else can.

In other words, if we know the factors of N we know how exponents behave modulo N. Other people don't.

The RSA algorithm: pick exponents e and d that are inverses modulo $\phi(N)$. Encryption is

$$C = M^e \pmod{N}$$

and decryption is

$$C^d \pmod{N} \equiv (M^e)^d \equiv M^{ed=1+k\phi(N)} \equiv M \cdot 1^k \equiv M$$

3 Other Topics

3.1 Diffie-Hellmann-Merkle key exchange

The discrete log problem is difficult: if I have a large prime p, and a generator g (meaning that every element of $\mathbb{Z}_p^\times$ can be written as g^a for some a), then it is easy to compute $H = g^h \pmod{p}$, but difficult to derive h given H. Exponentiation becomes a one-way function.

So, given a public prime p and generator g, Alice and Bob can each choose a random secret h_A, h_B, and compute their exponents $H_A = g^{h_A} \pmod{p}$, $H_B = g^{h_B} \pmod{p}$.

If Alice and Bob want to communicate, Bob can raise Alice's public number to the power of his secret exponent: $k = H_A^{h_B} = g^{h_A h_B} \pmod{p}$.

Alice does the same, and arrives at the same number $k = H_B^{h_A} = g^{h_A h_B} \pmod{p}$. Nobody else can compute this number without either possessing one of the secret values h_A or h_B, or having some efficient algorithm for computing discrete logarithms.

3.2 The El Gamal Encryption Algorithm

This encryption algorithm is simply the DHM key exchange rephrased as an encryption method.

Alice constructs a public key consisting of $\{p, g, H_A = g^{h_A} \pmod{p}\}$, keeping the secret value h_A as her private decryption key. Instead of p, g being public, they are part of Alice's key.

When Bob wants to send a message m, he chooses a random h_B and computes $c = mH_A^{h_B} \pmod{p}$. He sends the pair $\{c, H_B = g^{h_B} \pmod{p}\}$ to Alice. The secret h_B is randomly chosen for every message.

3.3 Complexity Theory

You should know what P and NP mean:

- P is the set of computational problems solvable in polynomial time.
- NP is the set of computational problems whose solution can be confirmed in polynomial time.

"Solvable in polynomial time" means that the running time of the algorithm, as a function of the input *size*, is bounded from above by some polynomial function.

3.4 Steganography

The goal of steganography is undetectable communication. Unlike cryptography, whose goal is secrecy of the message, the goal in steganography is secrecy of the channel. This is much harder to achieve.

The usual approach of steganography is disguising a message as something innocent, such as an image or audio clip. This innocent object is called the *cover message*.

The adversary in a steganographic problem is called the *warden*. The warden tries to either catch or prevent covert communication, and falls into three types:

- The passive warden just listens in on all communications, in search of anything suspicious.

- The active warden can add noise to the channel, for example recompressing audio clips or subtly warping images, to prevent a stego algorithm from being used. Message meaning can not be changed.
- The malicious warden is allowed to drastically change messages, and even forge messages from one person to another.

For covert communication, the embedder hopes that no statistical test can distinguish a natural message from a cover message. Since manipulation of natural data can induce all sorts of unusual statistics, there is no provable way to embed undetectably.

One application of steganography is *watermarking*. Watermarks are short labels embedded in multimedia for purposes of tracking, access control, and copyright. Watermarks need to be imperceptible, and also robust to compression, noise, AD/DA conversion and other innocent operations. Ideally they would also be immune to deliberate attempts to wipe away the watermark, but in most realistic scenarios that kind of robustness is impossible.

A final note: if covert communication is possible using a shared secret key, then it is possible to construct a "public-key" stego system. That is, it is possible to perform key exchange without the warden noticing. In 1998, it was discovered that covert key exchange is possible even under an active warden–although an efficient key exchange was not discovered until 2004.