MA080G Cryptography Summary of Block 1 Theory

Viktor Rosvall

April 8, 2019

Symmetric Ciphers

Symmetric Ciphers are divided into **Stream Ciphers** and **Block Ciphers**.

Block Ciphers encrypt entire blocks of bits at a time with the same key, such as AES-128. Block Ciphers are used more often on the Internet.

Stream Ciphers encrypts bits individually. This is done by adding (the same as XOR) a bit from the *key stream* to a plaintext bit (modulus 2).

The One-Time Pad

The One-Time Pad (OPD) is a perfect cipher. Meaning, unconditional security cipher: it cannot be broken even with infinite computational resources.

Defintion: OTP

A stream cipher for which

1. the key stream $s_0, s_1, s_2, ...$ is generated by a *true* random number generator (TNRG)
2. the key stream is only known to the legitimate communicating parties
3. every key stream bit s_i is used only once

The need for a TNRG means that the device has to have the capability to produce truly random bits, such as from a dice roll or white noise.

To ensure that the key stream isn't leaked, we must have a perfectly secure channel of communication.

The fact that each key bit is only used ones means that there is an equal amount of plaintext as keys bits

Improving substitution ciphers

Substitution ciphers can be solved using frequency analysis, so we need to improve the cipher.

This can be done by inserting additional symbols into the ciphertext with no meaning. Use a different alphabet to represent the ciphertext. Combining transposition with substitution.

The Vigenère cipher

A different *Ceasar Shift* is applied to each letter of the plaintext.

The key to this cipher is a sequence of numbers, explaining the shift length. If the plaintext is longer than the sequence, it repeats.

Example: The letters "enemy" $\rightarrow$ "JBBQQ" encrypted using the sequence (5,14,23,4,18).

The trick to remembering the sequence, is to shift the letter "a" to these numbers, for example: "aaaaa" "FOXES"

Breaking the Vigenère cipher

If we know the sequence length, we can split the ciphertext into multiple strings of the sequence length, and individually crack them.

Suppose we have the plaintext encoded by numbers: $x_0, x_1, x_2, ...$ And the key $k_0, k_1, k_2, ..., k_{n-1}$ of length n

Using the formula we can get the ciphertext y_i:

$$y_i = (x_i + k_{iMODn})MOD26$$

For example: Assume we have the key: "fishy", which corresponds to the sequence (5,8,18,7,24).

If we want to decrypt the cipher text "KQJZR", which in numeric corresponds to the "10,16,9,25,17", we would apply the formula 5 times:

$$\begin{aligned} 10 &= (x_0 + 5_{0MOD5})MOD26 \\ 10 &= (x_0 + 5_0)MOD26 \\ 5 &= x_0 MOD26 \\ x_0 &= 5 \end{aligned}$$

So the first letter "K" $\rightarrow$ "f". Continuing this process, "KQJZR" $\rightarrow$ "first".

Kasiski method for guessing the key length in a Vigenère cipher

Using the *Kasiski method* we can guess the key length n, by finding the greatest common divisor of the most common digrams and trigrams.

After we have guessed the key length n, we can divide the cipher text into n substrings. We can find the key letter by doing a frequency analysis on each column (representing each key letter).

Friedman's method for computing the key length in a Vigenère Cipher

We need to compute the *Index of Coincidence* (IOC) I, before we can calculate the key length.

Computing the IOC for a ciphertext

Let $n_0, n_1, n_2, ..., n_25$ be the number of As,Bs,Cs,...,Zs in the ciphertext. And n be the total length of the ciphertext.

We can calculate the IOC I by obtaining the number of identical letters.

$$I = \frac{\sum_0^{25} n_i(n_i - 1)}{n(n-1)}$$

Using the IOC to estimate the key length of a Vigenère cipher

With a ciphertext of length n, the key length k, can be approximated using this formula:

$$k \approx \frac{0.0265n}{(0.065 - I) + n(I - 0.0385)}$$

Where I is the IOC calculated above.

0.065 is the probability of choosing 2 identical letters in a English plaintext. 0.0385 is the probability of choosing 2 identical letters randomly selected. And $0.0265 = 0.65 - 0.0385$

The Chinese Remainder

Theorem. Given pairwise coprime positive integers $n_1, n_2, ..., n_k$ and arbitrary integers $a_1, a_2, ..., a_k$, the system of simultaneous congruences

$$\begin{aligned} x &\equiv a_1 (mod\ n_1) \\ x &\equiv a_1 (mod\ n_1) \\ &. \\ &. \\ &. \\ x &\equiv a_1 (mod\ n_1) \end{aligned}$$

has a unique solution modulo $N = n_1, n_2, ..., n_k$.

Example:

$$\begin{aligned} x &\equiv 6 (mod\ 7) \\ x &\equiv 4 (mod\ 8) \end{aligned}$$

We begin by calculating N, and new n_1, n_2

$$N = 7 * 8 = 56$$
$$n_1 = N/7 = 8$$
$$n_1 = N/8 = 7$$

So x becomes:

$$x = 6 * 8 + 4 * 7 = 76$$
$$x = [76]_{56} = [20]_{56}$$
$$x \equiv 20(mod\ 56)$$

Summary of Complexity Theory

Complexity Theory is the study of *time complexity*, how long it will take to compute functions, and the cost of computation.

We use the **Big-O notation**, O(n), to measure the time complexity of functions and algorithms.

Example: Let $f : R \to R$ be given by $f(x) = 10x + 30$. Then f is $O(x)$, on $x \geq 1$:

$$|f(x)| = |10x + 30| \leq |10x + 30x| = |40x| = 40|x|$$

because the constant 40 is insignificantly small on large x.

Complexity classes

Complexity class P, is polynomial-time solvable. Which means a Turing machine could solve it.

Complexity class ExpTime, is exponential-time solvable, again with a Turing machine.

Complexity class NP, is non-deterministic polynomial-time solvable.