Cryptologyt part 2 Keränen/Teeriaho Ramk 2006

2. Classical cryptosystems

■ 2.1 Types of classical cryptosystems

▪ Division according to the number of images of a character

A) **Monoalphabetic cipher** is a cipher, where each character has a fixed image character . (example : Caesar cipher).

B) **Polyaphabetic cipher** is a cipher, where characters can have many image characters in different positions of the message. . (Example: Vigenère cipher)

Polyalphabetic ciphers are usually more difficult to break than monoalphabetic ciphers.,

▪ Division according to the block size

Many classical algorithms encrypt one character at a time. . In others the message is first divided into 2 , 3 , .. character blocks (bigrams, trigrams,...), which are then encrypted.

2.2 Caesar cipher - cyclic permutation of alphabet

One of the oldest algorithms is a cipher used by Romans during the rule of Caesar. It was based on the rotation of alphabet. The key k is the rate of rotation of the alphabet. Caesar's army possessed encryption discs, which were used for encryption and decryption. They consisted of two discs with alphabet written on them. Encryption was made rotating the discs with respect to each other.

Example.
A substitution table of key $k = 5$.

$$\begin{pmatrix} a & b & c & d & e & f & g & h & i & j & k & l & m & n & o & p & q & r & s & t & u & v & w & x & y & z \\ f & g & h & i & j & k & l & m & n & o & p & q & r & s & t & u & v & w & x & y & z & a & b & c & d & e \end{pmatrix}$$

When the word "rovaniemi" is encrypted, the corresponding image is looked up from the substitution table: The cipher is *"wtafsnjrn"***.**

In Caesar - cipher decryption is done reading the table in reverse order..

Cyclic permutation

The idea of rotation, which we call also "cyclic permutation" of alphabet is still used as one of the steps in modern ciphers.

2.3 Cryptoanalysis of Caesar cipher

Caesar cipher is obviously easy to break even without computers.

Brute force - attack

The size of the keyspace is very small. It we use 26 character alphabet , there exists 25 possible keys.
Trying all possible keys, we get 25 possible messages, from which usually only one makes sense.
We call the method of searching all possible keys "Brute force attack".

Example of brute force attack with *Mathematica*

We create a *Mathematica* - function, which performs Caesar - encryption for message m using key k.
It encodes first characters using ASCII character codes to numbers 0 - 25 , then performs rotation adding the key k and taking the remainder mod 26, and finally decodes numbers back to characters.

```
CaesarE[m_, k_] :=
 FromCharacterCode[Mod[ToCharacterCode[m] - 97 + k, 26] + 97]
```

Assume the ciphertext is "wtafsnjrn"

```
c = "wtafsnjrn";
```

Use **brute - force** searching through the 25 possible keys

```
Table[CaesarE[c, -k], {k, 1, 25}]

{vszermiqm, urydqlhpl, tqxcpkgok, spwbojfnj, rovaniemi,
 qnuzmhdlh, pmtylgckg, olsxkfbjf, nkrwjeaie, mjqvidzhd,
 lipuhcygc, khotgbxfb, jgnsfawea, ifmrezvdz, helqdyucy,
 gdkpcxtbx, fcjobwsaw, ebinavrzv, dahmzuqyu, czglytpxt,
 byfkxsows, axejwrnvr, zwdivqmuq, yvchupltp, xubgtokso}
```

The 5th pre-image "rovaniemi" is the only one that makes sense, whence k = 5

Frequency analysis

Frequency analysis is a method, where we compare the frequencies of the characters in the message with the known frequencies of characters in the language.

A cipher, which encrypts one character at a time in a way that each character has a fixed image character, is easy to break with frequency analysis. One should require at least that the image of each character varies in different parts of the plaintext.

Example of frequency analysis

Assume, that the ciphertext c is from Caesar algorithm and the message language is English.

```
c = "yfxmpcespzcjtdfdpqfwqzcpyntaspctyrxpddlrpd"

yfxmpcespzcjtdfdpqfwqzcpyntaspctyrxpddlrpd
```

First we change the message into a list of characters

```
charlist = Characters[c]

{y, f, x, m, p, c, e, s, p, z, c, j, t, d, f, d, p, q, f, w, q,
 z, c, p, y, n, t, a, s, p, c, t, y, r, x, p, d, d, l, r, p, d}
```

The frequencies can be calculated using statistical package of Mathematica:

```
<< Statistics`DataManipulation`
```

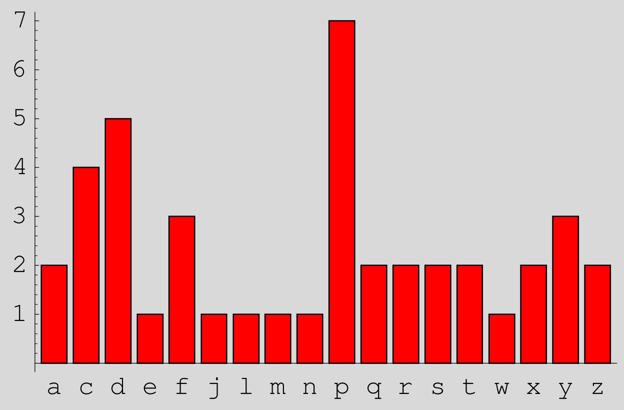
```
tab = Frequencies[charlist]

{{2, a}, {4, c}, {5, d}, {1, e}, {3, f}, {1, j}, {1, l}, {1, m}, {1, n},
 {7, p}, {2, q}, {2, r}, {2, s}, {2, t}, {1, w}, {2, x}, {3, y}, {2, z}}
```

The frequencies can also be presented graphically :

```
<< Graphics`Graphics`
```

```
BarChart[tab]
```



The relative frequencies of characters in different languages are known

English	*e*	*t*	*a*	*o*	*n*	*i*	*s*	*r*	*h*
	12.3	9.6	8.1	7.9	7.2	7.2	6.6	6.0	5.1
Finnish	*a*	*i*	*t*	*n*	*e*	*s*	*l*	*o*	*k*
	12.1	10.6	9.8	8.6	8.1	7.8	5.9	5.5	5.2

In our cipher letter p occurs most often. On the other hand in English letter e has the biggest relative frequency. Assuming that e maps to p in the Caesar encryption, we guess that k = 11. Lets test this hypothesis:

```
CaesarE[c, -11]

numbertheoryisusefulforencipheringmessages
```

This makes sense: "Number Theory is Useful For Enciphering Messages" .

2.4 Variations of Caesar cipher

There exists variations from Caesar in which rotation of alphabet is replaced by more complicated permutations. Also the key space is bigger.

Affine cipher $c \equiv a \times m + b \pmod n$

Let n be the size of alphabet.

1. Encode characters into elements of Z_n : $0, 1, ..., n-1$
2. Encrypt using an affine mapping $Z_n \to Z_n$

$$c \equiv a \times m + b \pmod n,$$

where a and $b \in Z_n$ and $\gcd(a, n) = 1$.

3. Because $c \equiv a \times m + b \Rightarrow a \times m \equiv c - b \Rightarrow m \equiv a^{-1} \times c - a^{-1} \times b$,

decryption is done by applying

$$m \equiv a^{-1} \times c - a^{-1} \times b \pmod n,$$

where a^{-1} is the inverse of a in Z_n.

In brute force attack all pairs (a^{-1}, b) must be tried. Their total number is $\varphi(n) \times n$.

Frequency analysis breaks affine cipher easily.

Example Encrypt "helsinki" with keys a = 5 and b = 11. Modulus n is 26.
First we encode the message into numbers.

```
m = ToCharacterCode["helsinki"] - 97

{7, 4, 11, 18, 8, 13, 10, 8}
```

Then we encrypt the message and finally decode numbers back to characters

```
a = 5; b = 11; n = 26;
c = Mod[a * m + b, n]
FromCharacterCode[c + 97]

{20, 5, 14, 23, 25, 24, 9, 25}
```

```
ufoxzyjz
```

For decryption the inverse of **a** mod 26 is needed.
In *Mathematica* we calculate inverse by

```
a ' = PowerMod[a, -1, n]

21
```

Decryption

```
opened = Mod[a ' * c - a ' * b, n]

{7, 4, 11, 18, 8, 13, 10, 8}
```

This is same as the original message

■ 2.5 Simple substitutions

The *key space* of permutations can be very large if we use a random permutation of alphabet.

Example:

$$\begin{pmatrix} a & b & c & d & e & f & g & h & i & j & k & l & m & n & o & p & q & r & s & t & u & v & w & x & y & z \\ n & d & s & w & a & q & f & i & p & l & g & y & b & m & c & v & k & x & r & j & t & o & e & z & u & h \end{pmatrix}$$

The key is the permutation table itself. The size of key space is 26!

```
26! // N

4.03291 × 10^26
```

Brute Force can not be used**.** A super computer performs 36 million MIPS = $36*10^{12}$ instructions per second.
A year has 365*24*60*60 seconds. Time spent on brute force is

$$\frac{26!}{365. * 24 * 60 * 60 * 36 * 10^{12}} \text{ years}$$

```
355230. years
```

F**requency analysis** works well. It is enough to quess less that ten characters right . The one already finds some word like "the" , "and". With reasonable computing power the cipher is broken.

■ 2.6 The significance of key size

Key space size is proportional to the breaking time of a *brute force* - attack.
The key space of 8 character password (lowercase + uppercase letters + 0 - 9 = 62)

```
62^8 // N

2.1834×10^14
```

Time if a complete search with super computer is

$$\frac{62^8}{36.*10^{12}}\ \mathbf{sec}$$

```
6.065 sec
```

If the password is 12 characters

$$\frac{62^{12}}{365*24*60*60*36.*10^{12}}\ \mathbf{years}$$

```
2.84178 years
```

When computing power is increased, we have just to use longer passwords

■ 2.7 Vigenère cipher

Blaise de Vigenere in the court of Henry III in 16th century France invented a cipher, which was for long considered to be safe. If the keyword length is long enough, it is difficult to break.
Vigenere system can be implemented by using modular arithmetics, which is the basic tool in modern cryptology.

Vigenère cipher: (using modular arithmetic)
Encryption:
1. Encode the message into numbers : 0 - 25 (a = 0, b = 1, ... , z = 25)
2. Encode keyword into numbers repeating it until the length of the message is reached
3. Encryption is done by addition mod 26
4. Decode numbers back to characters.

Decryption
Use same algorithm, but instead of addition , use subtraction mod 26.

Example: Encrypt. "Meet you tomorrow" with keyword k = "turvaketju"

$$\begin{pmatrix} m & e & e & t & y & o & u & t & o & m & o & r & r & o & w \\ t & u & r & v & a & k & e & t & j & u & t & u & r & v & a \end{pmatrix}$$

$$\begin{pmatrix} 12 & 4 & 4 & 19 & 24 & 14 & 20 & 19 & 14 & 12 & 14 & 17 & 17 & 14 & 22 \\ 19 & 20 & 17 & 21 & 0 & 10 & 4 & 19 & 9 & 20 & 19 & 20 & 17 & 21 & 0 \\ - & - & - & - & - & - & - & - & - & - & - & - & - & - & - \\ 5 & 24 & 21 & 14 & 24 & 24 & 24 & 12 & 23 & 6 & 7 & 11 & 8 & 9 & 22 \end{pmatrix}$$

addition mod 26

As characters this is "fyvoyyymxghlijw"

Decryption:

$$\begin{pmatrix} 5 & 24 & 21 & 14 & 24 & 24 & 24 & 12 & 23 & 6 & 7 & 11 & 8 & 9 & 22 \\ 19 & 20 & 17 & 21 & 0 & 10 & 4 & 19 & 9 & 20 & 19 & 20 & 17 & 21 & 0 \\ - & - & - & - & - & - & - & - & - & - & - & - & - & - & - \\ 12 & 4 & 4 & 19 & 24 & 14 & 20 & 19 & 14 & 12 & 14 & 17 & 17 & 14 & 22 \end{pmatrix}$$

subtraction mod 26

Security of Vigenère cipher

For keyword of length n, the key space size is 26^n. In the example n = 10. and key space size is $26^{10} = 1.4 * 10^{14}$ - for a super computer a few minutes job.
Notice that the keyword length is not known, which makes the brute force attack much more complex.

Kasiski method
1863 Prussian major **Kasisky** found a system to find the length of the keyword:
If a long ciphertext is available, we make statistics of occurences of same characters in a cipher. If there exists some distance which occurs more often than others, it is very probably the multiple of keyword length.

When the keyword length n is known, then Vigenere cipher is equivalent to n Caesar ciphers and can be broken with frequency analysis.

2.8 One time pad - "the only unbreakable cipher"

If Vigenère cipher uses a randomly generated , unique key for every session and if the keyword is as long as the message, this cipher is provably unbreakable It is called ***"One Time Pad"*** .

It is unbreakable because searching through the key space we get the set of all possible messages from the given cipher. Actually the ciphertext gives no information about the message.

The easiest version of One Time Pad uses binary character alphabe. It is called **Vernam cipher.**

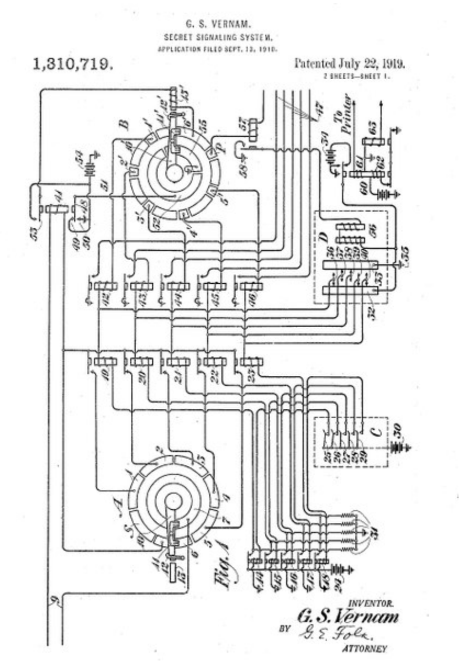
Addition mod 2 is called XOR (symbol $\oplus$) : $0 \oplus 0 = 0$, $0 \oplus 1 = 1$, $1 \oplus 0 = 1$ ja $1 \oplus 1 = 0$.

One Time Pad
Encryption:
1. Present the message as a binary string *m*.
2. Generate a random binary key *k* with the same length as the message
3. Encryption is done by bitwise XOR : $\boldsymbol{c = m \oplus k}$
4. Send the key to the receiver using a secure channel

Decryption:
Use same algorithm: Add the key again to the cipher to get the message. $\boldsymbol{m = c \oplus k}$

Never use the same key twice !

Gilbert Vernam patented 1919 in USA an electric encryption machine.



Mathematica code

```
m = {1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1};
k = {0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0};
c = Mod[m + k, 2]

{1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1}
```

Decryption is done similarly, because c + k = (m + k) + k = m + (k + k) = m.:

```
Mod[c + k, 2]

{1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1}
```

Moscow Washington hot line used One Time Pad during the cold war.
The key was different for all sessions.

One Time Pad has two problems: changing key length and difficulty of delivering the key.

GSM - encryption algorithm A5 imitates One Time Pad. However A5 key is not random , but pseudorandom. Pseudorandom bit sequence is completely deterministic when the initial state of the generator is given.

■ 2.9 Enigma

In World War II the German army had an encryption machine called Enigma.
Polish and British managed to break it (in the group Alan Turing).
Enigma was based on permutations and substitutions.