MATH187A: Introduction to Cryptography

Course Webpage

October 15, 2021

Abstract

Warning: This is only a piece of lecture notes written by a careless scribe. So just be careful with and tolerant of any possible typos or misunderstandings when you read [0.1]. The scribe does not intend to make anyone to be driven by his stupidity! Also, the professor's explanation is extremely helpful as he discusses a lot about the interpretable ideas behind the dull scripts. So watch the lecture before reading this. If you have any suggestions (e.g. typos, typography, logistics), please do not hesitate contacting the scribe!

Without specifications, the notation use is as the following

- $\mathbb{R}, \mathbb{C}, \mathbb{Q}, \ldots$: real, complex, quadratic, and so on

[0.1] Especially '$\cap$' and '$\cup$' are often mistaken because of typos.

Contents

1 Introduction to Cryptography 3
1.1 Introduction 3
2 Early Ciphers 5
2.1 Early Ciphers 5
3 Modular Arithmetic 8
4 Invertible Modulo 9
5 Berlekamp Algorithm 11
6 Probability 13

MATH187A **Fall 2021**

Lecture 1: Introduction to Cryptography

Lecturer: Alina Bucur *Scribes: Rabbittac*

1.1 Introduction

Cryptography: from Greek kryptos (hidden, secret) and graphen (writing).

- **Cryptography**: the art of secret writing.
- **Plaintext**: text to be encoded for secrecy.
- **Ciphertext**: encoded text.
- **Cipher**: a method of secret writing.
- n**-gram**: a string of n-letters.
- **Encipherment / Encryption**: the process of encoding plaintext into ciphertext.
- **Decipherment / Decryption**: the process of decoding ciphertext back into plaintext.
- **Sender**: the person that is to send the encrypted message.
- **Receiver**: the person which is to receive and decrypt the message
- **Opponent**: the person which intercepts the message and attempts the unauthorized decipherment.
- **Cryptographic system / Encipherment scheme**: a family of ciphers, where each member of the family is determined by a particular key.
- **Key**: the information, usually a sequence of digits or symbols, used to determine the algorithm by which plaintext is to be transformed into ciphertext.
- **Message space**: the collection of all messages that may occur in a particular cryptographic transaction.
- **Key space**: the collection of all keys that may occur in a given cryptographic system.
- **Cryptanalysis**: the process by which the opponent attempts to recover the original plaintext from the intercepted ciphertext.
- **Code breaking**: the process by which a cryptographic system is made vulnerable to cryptanalysis.
- **Onetime pad / Ephemeral key**: a key to be used only once.

In a typical set up, the sender and the receiver choose a cryptographic system and, at some time before the message is to be sent, the sender chooses the key. This determines which transformation of the system will be used to encrypt the

message. The key is then sent to the receiver by some safe path. Upon obtaining the key, the receiver determines which transformation of the system is to be used to decrypt the message.

The security of the message is not usually expected to be achieved through the opponent's ignorance of the encryption system but rather from lack of knowledge as to which particular key has been used in the encryption.

Two common methods of encryption are

1. **Substitution**: when individual letters or n-grams of plaintext are replaced by letters or n-grams of ciphertext.
2. **Transposition**: when the characters or words of the original message are rearranged according to some particular pattern.

The objective of this course is to learn how encrypted data, in reasonably large amounts, can be made vulnerable to computer attacks.

MATH187A **Fall 2021**

Lecture 2: Early Ciphers

Lecturer: Alina Bucur *Scribes: Jacky Wang*

The cryptographic transaction can be viewed as twp-person games between the sender-receiver on one side and the opponent on the other side. And such game is often under a few rules:

1. **Ciphertext only attack**: the opponent is to recover plaintext only through knowledge of ciphertext.
2. **Known plaintext attack**: the opponent may have access to some information concerning the original plaintext.
3. **Chosen plaintext attack**: the opponent is in a position to acquire ciphertext corresponding to plaintext of his selection.

In **classical cryptography**, only ciphertext-only or known-plaintext attacks are allowed.

2.1 Early Ciphers

e.g.1.

- The scytale — roll a paper at a stick
- XII century templars — symbols corresponding to letters
- Alberti's wheels

Rectangular transposition: Given plaintext "the baboons are coming for you", first divide it into five groups of 5-grams. Then perform rectangular transposition and break it up into 3-gram.

e.g.2.

3	5	1	4	2		1	2	3	4	5
T	H	E	B	A		E	A	T	B	H
B	O	O	N	S	$\longrightarrow$	O	S	B	N	O
A	R	E	C	O		E	O	A	C	R
M	I	N	G	F		N	F	M	G	I
O	R	Y	O	U		Y	U	O	O	R

Table 2.1: example of row version of rectangular transposition

Caesar code: Moving each letter by some letters. There are 26^k keys of length k.

e.g.3. KHUH LV DQ HADPSOH $\longrightarrow$ HERE IS AN EXAMPLE.

ABCDEFGHIJKLMNOPQRSTUVWXYZ
DEFGHIJKLMNOPQRSTUVWXYZABC

Table 2.2: Caesar code

e.g.4.

ABCDEFGHIJKLMNOPQRSTUVWXYZ
GHIJKLMNOPQRSTUVWXYZABCDEF
OPQRSTUVWXYZABCDEFGHIJKLMN
LMNOPQRSTUVWXYZABCDEFGHIJK
FGHIJKLMNOPQRSTUVWXYZABCDEF

Table 2.3: Caesar cicular shifts for the key "GOLF"

SEND FOOD SEND AMMUNITION $\longrightarrow$ YSYILCZIYSYIGAXZ TWENUB.

Playfair's Cryptosystem: The key is 5×5 box of the alphabet, with IJ in the same box. The first n characters are initialized by a n-letter word and the rest is arranged from A to Z, excluding any repeating character. For each plaintext, break it into 2-grams and use x to break up repeated letters. Then encrypt the plaintext using key: if 2 letters are in the same column, shift each down; if 2 letters are in the same row, shift each right; if 2 letters are at opposite corners of a rectangle, replace each with the other corner on the same row.

e.g.5.

M	O	N	K	E
Y	L	V	A	B
C	D	F	G	H
I/J	P	Q	R	S
T	U	W	X	Z

Table 2.4: Playfair's cryptosystem with key "MONKEY"

"plaintext takes time" $\longrightarrow$ "pl ai nt ex tx ta ke st im eq" $\longrightarrow$ "DU YR MW KZ UZ XY EM IZ TY NS".

ADFGVX system: The key consists of 6×6 gird of the letters and digits and a permutation of $1, 2, \ldots, n$. Replace each letter of plaintext with its coordinates. Then rearrange text into n columns and then read down the column labeled 1, then 2, ... Finally break into 5-grams.

e.g.6.

	A	D	F	G	V	X
A	C	O	8	X	F	4
D	M	K	3	A	Z	9
F	N	W	L	0	J	D
G	5	S	I	Y	H	U
V	P	1	V	B	6	R
X	E	Q	7	T	2	G

Table 2.5: ADFGVX system

"HQ REQUESTS FRONT LINE SITUATION BY TELEGRAM. HQ 7TH CORPS." ⟶ GFGVV VAGFG XGADV GAGXX XVXXX XXVGX DAAAD XDXFV VVFGF GFFDG GAGVA AAGAA XXXXA GGGXF DXGAG XFDXA DGGXD XFFXX AFDGA DDGDX

MATH187A **Fall 2021**

Lecture 3: Modular Arithmetic

Lecturer: Alina Bucur *Scribes: Rabbittac*

Monoalphabetic cipher: Each letter takes a corresponding letter to replace. There are 26! possible keys.

Homophonic substitution: Randomly label 1–26 to A–Z for 4 times and arrange them into 4 rows. Then add $25(i-1)$ to the i-th row for $1 \leq i \leq 4$.

e.g.1.

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
20	21	22	23	24	25	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
38	39	40	41	42	43	44	45	46	47	48	49	50	26	27	28	29	30	31	32	33	34	35	36	37
66	67	68	69	70	71	72	73	74	75	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65
96	97	98	99	100	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95

"The box will arrive by train" $\longrightarrow$ 59 78 24 97 54 17 34 46 5 81 20 29 87 74 91 42 67 18 31 87 66 3 7; 84 47 21 8 26 52 24 29 $\longrightarrow$ "OK BOOMER".

We can use 1–26 to represent A–Z in Caesar code. So the encryption is to add the given shift to each letter and the decryption is to subtract the given shift from each letter. Then take $\mod 26$.

Vernam encryption: Let M be the plaintext, C be the ciphertext, W be the key. $C = (c_1, c_2, \ldots, c_n) = (m_1, m_2, \ldots, m_n) + (w_1, w_2, \ldots, w_n) \mod 26$ and $M = (m_1, m_2, \ldots, m_n) = (c_1, c_2, \ldots, c_n) - (w_1, w_2, \ldots, w_n) \mod 26$.

Joseph Mauborgne showed that the Vernam one-time pad was vulnerable. One of the recipes is to generate characters by a purely random process.

Consider $A = Bq + r$, where q is the quotient and $0 \leq r < B$ is the remainder. The modular arithmetic is formulated $r = A \mod B$.

MATH187A **Fall 2021**

Lecture 4: Invertible Modulo

Lecturer: Alina Bucur *Scribes: Rabbittac*

We say a^{-1} is **invertible modulo** m if

$$\exists a : aa^{-1} \equiv 1 \mod m$$

so $aa^{-1} = km + 1$ for some k, which implies a^{-1} and m are coprime. To compute inverse p in modular arithmetic, for small p, we use the multiplication modulo p table; for large p, we use the Euclidean algorithm.

e.g.1. $3^{-1} \equiv 9 \mod 26, 5^{-1} \equiv 21 \mod 26, 7^{-1} \equiv 15 \mid 26.$

e.g.2. Solve the linear equation modulo 26: $7x + 5 \equiv y \mod 26$.
$7x + 5 \equiv y \mod 26 \implies 7x \equiv y - 5 \mod 26 \implies x \equiv 7^{-1}(y - 5) \mod 26 \implies x \equiv 15y + 3 \mod 26$.

Affine cipher with modulus p and key a, b has the encryption function $E(x) = ax + b \mod p$ and the decryption function $D(y) = a^{-1}(y - b) \mod p$, where A∼Z is mapped to $0 \sim 25$.

Algorithm 4.1 Affine cipher

Encryption

1: Given a, b, p.
2: Map A∼Z with $0 \sim 25$.
3: Encrypt each character by $E(x) = ax + b \mod p$.

Decryption

1: Given a ciphertext y.
2: Decrypt each character by $D(y) = a^{-1}(y - b) \mod p$.

e.g.3. Let $a = 7, b = 5, p = 26$. Then $\mathrm{A} \to 0 \to 0 \times 7 + 5 \mod 26 = 5 \to \mathrm{F}; \mathrm{B} \to 1 \to 1 \times 7 + 5 \mod 26 = 12 \to \mathrm{M}; \mathrm{A} \to 2 \to 2 \times 7 + 5 \mod 26 = 19 \to \mathrm{T}$

If p is a prime, then all $x > 0$ have an inverse mod p.

The **superincreasing sequences** can be used for ciphering. Let $s_1, s_2, \ldots, s_n$ be a sequence with $s_j > \sum_{i=1}^{j-1} s_i$. To see whether T can be expressed as a subset sum, repeatedly search r using $s_1, s_2, \ldots, s_{k-1}$ with following procedure: if $T < s_1$ or $T > \sum_{i=1}^{n} s_i$, then no such subset exists; otherwise find the largest k such that $s_k \leq T$ and find r such that $T = r + s_k$.

e.g.4. Write 55 as the sum of distinct powers of 2.
$55 = 32 + 23 = 32 + 16 + 7 = 32 + 16 + 4 + 3 = 32 + 16 + 4 + 2 + 1.$

Merkle Hellman Knapsack cipher chooses a superincreasing sequence $s_1, s_2, \ldots, s_n$ and p to be a large prime number such that $p > \sum_{i=1}^{n} s_i$. Let a be

a random number between 1 and $p-1$ and publicly announce $t_1, \ldots, t_n$ where $t_j \equiv as_j \mod p$.

Knapsack cipher is used to encode a message $(x_1, \ldots, x_n)$ consisting of 0 and 1 and the sender sends the single number $C = \sum_{i=1}^{n} x_i t_i$.

For decryption, we solve the subset sum problem for $M = a^{-1}C \mod p$. That is, we need to find $x_1, \ldots, x_n$ such that $M = \sum_{i=1}^{n} x_i s_i$.

e.g.5. Let $\{3, 5, 12, 21, 43\}$ be the sequence, $p = 89$, and $a = 15$. Then $T = \{45, 75, 2, 48, 22\}$. So we encode 01101 and get $C \equiv 10 \mod 89$. To decode, since $a^{-1} \equiv 6 \mod 89$, we have $M = a^{-1}C = 60 = 17 + 43 = 5 + 12 + 43 = 0 \times 3 + 1 \times 5 + 1 \times 12 + 0 \times 21 + 1 \times 43$.

Algorithm 4.2 Algorithm for generating random number.

1: Initialize a start sequence R_0 as seed.
2: **for** $i = 1, 2, \ldots, 30$ **do**
3: Update by $R_i = R_{i-1} \times 2^{27} \mod 277998721$.
4: **end for**
5: **if** $k \geq 31$ **then**
6: Update by $R_k = R_{k-1} + R_{k-31} \mod 277998721$.
7: **end if**

Hill cipher uses a key of $k \times k$ matrix $A = \begin{pmatrix} a_{11} & \ldots & a_{1k} \\ \vdots & \ddots & \vdots \\ a_{k1} & \ldots & a_{kk} \end{pmatrix}$ and a modulus p. The plaintext is cut into k-grams and each k-gram is converted into a code vector $x_1 x_2 \ldots x_k$. Encode by the matrix multiplication $\begin{pmatrix} y_1 \\ \vdots \\ y_k \end{pmatrix} = A \begin{pmatrix} x_1 \\ \vdots \\ x_k \end{pmatrix} \mod p$. Then convert $y_1 y_2 \ldots y_k$ into a k-gram.

For decryption, to construct the inverse matrix in mudulo p arithmetic, we first obtain the adjoint matrix A^* where $A \cdot A^* = (\det A)I$. Then if $\det A \not\equiv 0 \mod p$, we compute its inverse by $c \cdot (\det A) \equiv 1 \mod p$. The inverse matrix is $A^{-1} = cA^*$.

Algorithm 4.3 Hill cipher

Encryption

1: Given key $A = \begin{pmatrix} a_{11} & \ldots & a_{1k} \\ \vdots & \ddots & \vdots \\ a_{k1} & \ldots & a_{kk} \end{pmatrix}$ and modulus p.
2: Divide the plaintext into k-grams.
3: Each k-gram is converted into a code vector $(x_1, x_2, \ldots, x_k)^T$.
4: Encrypt the code vector by $\begin{pmatrix} y_1 \\ \vdots \\ y_k \end{pmatrix} = A \begin{pmatrix} x_1 \\ \vdots \\ x_k \end{pmatrix} \mod p$. Then convert $y_1 y_2 \ldots y_k$.
5: Convert the ciphered code vector into k-grams.

Decryption

1: Compute A^* by $A \cdot A^* = (\det A)I$.
2: **if** $\det A \not\equiv 0 \mod p$ **then**
3: Compute inverse by $c \det A \equiv 1 \mod p$.
4: **end if**
5: The inverse matrix is $A^{-1} = cA^*$.

MATH187A **Fall 2021**

Lecture 5: Berlekamp Algorithm

Lecturer: Alina Bucur *Scribes: Rabbittac*

e.g.1. Encrypt $A = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$ with the Hill cipher.

$A = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right) \implies A^* = \left(\begin{smallmatrix} d & -b \\ -c & a \end{smallmatrix}\right) \mod p = \left(\begin{smallmatrix} d & \bar{b} \\ \bar{c} & a \end{smallmatrix}\right)$ where $\bar{e} \equiv -e \mod p$. Then $\det A = ad - bc \mod p$. Now $A \cdot A^* = \left(\begin{smallmatrix} ad-bc & 0 \\ 0 & ad-bc \end{smallmatrix}\right) \mod p$. Then $A^{-1} = (ad-bc)^{-1}\left(\begin{smallmatrix} d & \bar{b} \\ \bar{c} & a \end{smallmatrix}\right) \mod p$.

e.g.2. Given the plaintext PD and $T = \left(\begin{smallmatrix} 11 & 13 \\ 5 & 6 \end{smallmatrix}\right)$. Map PD into $0 \sim 25$ so $\text{PD} = \left(\begin{smallmatrix} 15 \\ 3 \end{smallmatrix}\right)$. Then $T\left(\begin{smallmatrix} 15 \\ 3 \end{smallmatrix}\right) \equiv \left(\begin{smallmatrix} 1 \\ 6 \end{smallmatrix}\right) \mod 29 \to \text{BG}$.

To decrypt it, $T^* = \left(\begin{smallmatrix} 6 & 16 \\ 24 & 11 \end{smallmatrix}\right)$ so $\det T = 66 - 65 = 1$ Then $T^{-1} = 1 \cdot T^* = \left(\begin{smallmatrix} 6 & 16 \\ 24 & 11 \end{smallmatrix}\right)$. $T^{-1}\left(\begin{smallmatrix} 1 \\ 6 \end{smallmatrix}\right) \mod 29 = \left(\begin{smallmatrix} 15 \\ 3 \end{smallmatrix}\right)$.

The Euclidean algorithm is used to get the greatest common divisor d of two given integers A and B.

Algorithm 5.1 Recursion Implementation of Euclidean algorithm

Given two integers A, B.

```
procedure EuclideanAlgorithm(A, B)
    if B == 0 then
        return A
    end if
    return EuclideanAlgorithm(B, A mod B)
end procedure
```

Berlekamp modified the Euclidean algorithm to compute $\gcd(A, B)$ 5.2. Let a_k be the quotient and r_k be the remainder. The iterations force the following identities when $r_k = 0$, and since r_k decreases at least by 1 in each iteration, we eventually obtain $r_k = 0$.:

1. $q_k p_{k-1} - p_k q_{k-1} = (-1)^k$
2. $A = r_{k-1} p_k$
3. $B = r_{k-1} q_k$

Proof: $\left(\begin{smallmatrix} p_k & p_{k-1} \\ q_k & q_{k-1} \end{smallmatrix}\right) = \left(\begin{smallmatrix} a_0 & 1 \\ 1 & 0 \end{smallmatrix}\right)\left(\begin{smallmatrix} a_1 & 1 \\ 1 & 0 \end{smallmatrix}\right) \dots \left(\begin{smallmatrix} a_k & 1 \\ 1 & 0 \end{smallmatrix}\right)$. Since the determinant of a product the product of the determinants of the factors, we get $q_k p_{k-1} - p_k q_{k-1} = (-1)^k$. Then the iterations yield $\left(\begin{smallmatrix} p_k & p_{k-1} \\ q_k & q_{k-1} \end{smallmatrix}\right)\left(\begin{smallmatrix} r_{k-1} \\ r_k \end{smallmatrix}\right) = \left(\begin{smallmatrix} A \\ B \end{smallmatrix}\right)$. When $r_k = 0$, this reduces to $\left(\begin{smallmatrix} p_k & p_{k-1} \\ q_k & q_{k-1} \end{smallmatrix}\right)\left(\begin{smallmatrix} r_{k-1} \\ 0 \end{smallmatrix}\right) = \left(\begin{smallmatrix} A \\ B \end{smallmatrix}\right)$. So we get $A = r_{k-1} p_k$ and $B = r_{k-1} q_k$.

Multiplying (2) by q_{k-1} and (3) by p_{k-1} and subtracting gives $Bp_{k-1} - Aq_{k-1} = (-1)^k r_{k-1}$. Thus $r_{k-1} = \gcd(A, B)$. If r_{k-1}, then $Bp_{k-1} - Aq_{k-1} = (-1)^k \implies B(-1)^k p_{k-1} = 1 + A(-1)^k q_{k-1}$ and we get the inverse of B modulo A. ∎

Algorithm 5.2 Berlekamp Algorithm

Given two integers A, B.
1: Set $r_{-2} = A, r_{-1} = B, p_{-2} = 0, p_{-1} = 1, q_{-2} = 1, q_{-1} = 0$.
2: **while do**$r_k \neq 0$
3: $r_{k-2} = a_k r_{k-1} + r_k$
4: $p_k = a_k p_{k-1} + p_{k-2}$
5: $q_k = a_k q_{k-1} + q_{k-2}$
6: **end while**

ASCII is the American Standard Code for Information Interchange

e.g.3. 01000001 — A, 01100001 — a, 01011010 — Z, 01111010 — z, 00110000 — 0, 00111001 — 9.

e.g.4. Enter the ASCII code for letter b using $t_1, \ldots, t_8$ published by Alice: 120, 300, 540, 323, 826, 955, 216, 612.

ASCII code for b is 01100010. Then $C = 0\times120+1\times300+\cdots+0\times612 = 1056$.

e.g.5. Alice picked the superincreasing sequence $\{2, 5, 9, 22, 47, 99, 203, 409\}$, the prime $p = 997$, and the encryption factor $a = 60$. Bob uses the published numbers t_i to encrypt a message and sends the result $C = 1255$ to Alice. Use Alice's key to encrypt Bob's message.

Since p is big, we use Berlekamp's algorithm with $A = 997$ and $B = 60$. We compute a_i as $997 = 60 \times 16 + 37$ so $a_0 = 16, \ldots, a_1 = \cdots = a_6 = 1$, and $a_7 = 4$. Now iteratively compute $p_0 = 16 \times 1 + 0 = 16, q_0 = 16 \times 0 + 1 = 1, \ldots, p_7 = 4 \times 216 + 133 = 997, q_7 = 4 \times 13 + 8 = 60$. Then $60 \times 216 - 997 \times 13 = (-1)^7 = -1$ so $60^{-1} \equiv -216 \mod 97 \equiv 981 \mod 997$.

Now compute $M \equiv a^{-1}C \mod p \equiv 104 \mod 997$ and write M in terms of the superincreasing sequence: $M = 104 = 1 \times 99 + 5 = 0 \times 2 + 1 \times 5 + \cdots + 0 \times 409$. So the ASCII code is 01000100 — D.

MATH187A **Fall 2021**

Lecture 6: Probability

Lecturer: Alina Bucur *Scribes: Rabbittac*

Motivation: count the frequency of each letter appears in daily English.
Studied elementary probability in scribe's every class so he is lazy.

The same excuse holds.