

**Notes and Instructions:**

- Some questions may ask you to write a computer program, or you as part of solving a question you might write a computer program to help you. Include the source code of your program in the contents of the PDF / Word file / screenshot that you submit to Crowdmark. We will not run your code, but we may read it and allocate marks for it, so please include a few comments so we can understand what you've done.
- For questions on this assignment where we give you messages/ciphertexts to work with, you can assume the following:
  - All plaintexts are passages of “typical” English text, written in a similar style, without any intentional abnormalities.
  - To encode a message for encryption, all punctuation and whitespace will be removed, so that the plaintext is a string in  $\{A, \dots, Z\}^*$ .
  - If the characters of the plaintext need to be represented as numbers during encryption/decryption operations, they will be represented as integers modulo 26, with the mapping  $A \rightarrow 0, B \rightarrow 1, \dots, Z \rightarrow 25$ .
- Some questions use randomization to customize to you specifically. Please include your max-8-character UW user id (g66xu) at the beginning of your answer so we can look up your custom solution.

1. [8 marks] **Index of coincidence.**

The *index of coincidence* of a string measures the probability that two randomly chosen characters in the string are the same. For some ciphers, the index of coincidence can be used to help determine an unknown block length given just the ciphertext.

Let  $x = x_1x_2 \dots x_n$  be a string of  $n$  characters from the alphabet  $\{A, \dots, Z\}$ . The *index of coincidence* of  $x$  is

$$I_c(x) = \frac{\sum_{i=A}^Z \binom{f_i}{2}}{\binom{n}{2}}$$

where  $f_i$  is the number of occurrences of the letter  $i$  in the string  $x$ . (Recall that  $\binom{k}{2}$  is “ $k$  choose 2” and is equal to 0 if  $k < 2$ , and  $\frac{k(k-1)}{2}$  if  $k \geq 2$ .)

- [3 marks] Write a function in Sage, Python, or a language of your choice that calculates the index of coincidence for a given string. Use it to calculate the index of coincidence for the file `q1-mobydick.txt`, available on LEARN, and include the calculated value in your answer.
- [3 marks] The index of coincidence for “normal” passages of English text will all be very close to each other, and the value for `q1-mobydick.txt` is fairly typical; call this value  $I_c^{\text{english}}$ .

Suppose we are given a ciphertext  $c = c_1c_2 \dots c_n$  encrypted using the Vigenère cipher, but with an unknown block length. We can use the following procedure to assess whether a particular guess of the block length is plausible:

- For each candidate block length  $m$ :
  - Define  $m$  strings  $y_1, \dots, y_m$ , where

$$y_1 = c_1c_{m+1}c_{2m+1} \dots$$

$$y_2 = c_2c_{m+2}c_{2m+2} \dots$$

$$\vdots$$

$$y_m = c_mc_{2m}c_{3m} \dots$$

In other words,  $y_i$  is the string formed by taking all the ciphertext characters at positions  $i \bmod m$ .

- For each  $y_1, \dots, y_m$ , compute the index of coincidence for all of them.
- If these indices of coincidence are all close to  $I_c^{\text{english}}$ , then  $m$  is a plausible guess at the block length. (How “close” is “close”? You’ll know it when you see it.)
- If these indices of coincidence are *not* all close to  $I_c^{\text{english}}$ , then  $m$  is a *not* plausible guess at the block length.

Write a program in Sage, Python, or a language of your choice that you can use to determine plausible block length(s) for the ciphertext in the file `q1-ciphertext.txt` on LEARN; it was encrypted using the Vigenère cipher with a block length somewhere between 1 and 100.

Note that you should *not* assume that the block length is a divisor of the ciphertext length; padding may have been used or a few characters at the end may have been dropped.

- (c) [2 marks] Is this approach effective for identifying the block length of the transposition cipher? Why or why not?

2. [15 marks] **Cryptanalysis of historical ciphers**

Please include your max-8-character UW user id (`g66xu`) at the beginning of your answer so we can look up your custom solution.

For this question, you need to obtain the ciphertexts personalized to you. Download the file [https://www.math.uwaterloo.ca/~dstebila/as/as.cgi/download/co487\\_f23/a1q2ciphertexts.zip](https://www.math.uwaterloo.ca/~dstebila/as/as.cgi/download/co487_f23/a1q2ciphertexts.zip).

Your folder contains 4 ciphertexts, each of which is an encryption of some English plaintext written in a similar style of text. The plaintext may start in the middle of a sentence and may end in the middle of a word.

Each plaintext is different and each is encrypted with a different cipher. The four ciphers used, in random order in your set, are:

- shift cipher
- substitution cipher
- Vigenère cipher, for an unknown block length between 6 and 13
- transposition cipher, for an unknown block length between 6 and 13

Your task is to determine which cipher was used for each ciphertext, and what the corresponding plaintext is, using the following steps. You may write software to do so in any language of your choice, or use online tools to assist, but please indicate which tool(s) you used. If you write your own software to help you solve, please include that in your submission.

- (a) [4 marks] For each ciphertext, using either a table or a histogram, present the single character frequencies and a description of the relevant properties of the single character frequencies.
- (b) [4 marks] Explain clearly how the single character frequencies are *expected* to look for the ciphertext from each of the four cipher algorithms used.

Use the experimental observations from part (a) to argue which ciphertext comes from which of the four ciphers.

- (c) [4 marks] Explain, with direct reference to the statistics that you found, the procedure you can use to cryptanalyze each cipher. You are not expected to perform the cryptanalysis in this part – you need to explain how it can be done in principle for each type of cipher and how the measured statistics can help.
- (d) [3 marks] Using whatever technique or tools you like, obtain the key and plaintext for the ciphertexts encrypted using the shift, substitution, and Vigenère cipher. It is okay if you only give the first 50 or so characters of the plaintext. It is okay if the first or last few characters of your plaintext are nonsensical, as the plaintext may have been interrupted in the middle of a word and may not be a multiple of the block length.

3. [7 marks] **Cryptanalysis of an LFSR**

Please include your max-8-character UW user id (g66xu) at the beginning of your answer so we can look up your custom solution.

In the lectures on stream ciphers, we noted that stream ciphers do not use the output of linear feedback shift registers (LFSRs) directly, instead they apply some non-linear filter or combiner prior to generating the output. In this question, we will see why, by developing a technique to cryptanalyze an LFSR with unknown feedback coefficients using a known-plaintext attack.

Let  $(k_0, \dots, k_{m-1})$  be the  $m$ -bit key that is used to initialize the LFSR, in other words,

$$s_0 = k_0, s_1 = k_1, \dots, s_{m-1} = k_{m-1}.$$

Let the LFSR be defined by the recurrence relation

$$s_{i+m} = \sum_{j=0}^{m-1} c_j s_{i+j} \bmod 2$$

for  $i \geq 0$ , where  $c_0, \dots, c_{m-1} \in \mathbb{Z}_2$ .

Suppose we know an  $n$ -bit plaintext  $x_0, x_1, \dots, x_{n-1}$  and the corresponding ciphertext  $y_0, y_1, \dots, y_{n-1}$ , in other words,

$$y_i = x_i \oplus s_i \tag{1}$$

for  $i \geq 0$ .

If  $n \geq 2m$ , we can derive a system of  $\geq m$  linear equations in  $m$  unknowns, which can then be solved.

Suppose Eve obtains the ciphertext

0011110010110101

corresponding to the plaintext string

0001111101100011

and suppose we know that this was generated a 4-stage LFSR with unknown feedback coefficients, namely

$$s_{i+4} = c_0 s_i + c_1 s_{i+1} + c_2 s_{i+2} + c_3 s_{i+3}$$

- (a) [1 marks] Compute the keystream used to encrypt.
- (b) [1 marks] What is the initial state of the LFSR?
- (c) [2 marks] Write a matrix equation for the unknown feedback coefficients  $c_0, c_1, c_2, c_3$  in terms of the keystream.
- (d) [2 marks] Solve the linear system for the feedback coefficients.<sup>1</sup>
- (e) [1 marks] Draw LFSR with the initial state filled in.

4. [9 marks] **Hill cipher.**

The Hill cipher is another historical cipher; it is based on linear algebra.

Let  $\ell$  be the block length.

The message space will be strings over the alphabet  $\{A, \dots, Z\}$  whose length is a multiple of the block length. To encrypt/decrypt using the Hill cipher, we will encode a message of length  $n\ell$  as

---

<sup>1</sup>This calculator may be helpful for computing matrix inverses modulo 2: <https://planetcalc.com/3324/>

an  $n \times \ell$  matrix over  $\mathbb{Z}_{26}$  (that is, the entries of the matrix are integers modulo 26). For example, the message **ABCDEF** will be represented by the matrix

$$\begin{pmatrix} 0 & 1 \\ 2 & 3 \\ 4 & 5 \end{pmatrix}$$

The key space will be the set of all  $\ell \times \ell$  invertible matrices over  $\mathbb{Z}_{26}$ .

To encrypt a message  $m \in \{A, \dots, Z\}^{n\ell}$  of length  $n\ell$ , represent it as an  $n \times \ell$  matrix  $M$  of integers mod 26, then compute the ciphertext using matrix multiplication:

$$C = MK \bmod 26$$

You may choose to keep the ciphertext as a matrix of integers modulo 26, or convert it back to a string of letters.

To decrypt a ciphertext  $C$  represented as a matrix of integers modulo 26, compute

$$CK^{-1} = (MK)K^{-1} = M \bmod 26$$

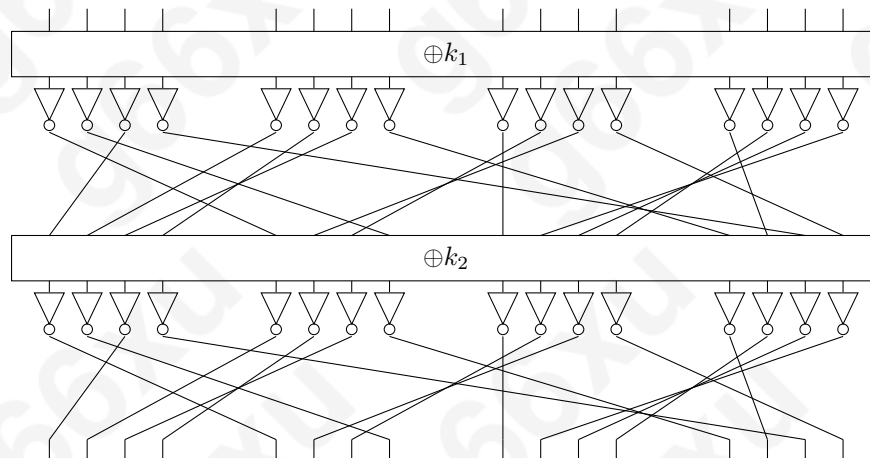
and then convert the matrix of integers  $M$  back to a string of letters  $m$ .

- (a) [2 marks] Encrypt the message **COOKIE** using key  $K = \begin{pmatrix} 13 & 16 \\ 8 & 23 \end{pmatrix}$ .
- (b) [3 marks] Assume that you know the block length  $\ell$ . Explain how you could recover the key if you know a plaintext-ciphertext pair  $(m, c)$ . What is the minimum number of blocks that the message  $m$  needs to have? Do you need any specific conditions on these blocks?
- (c) [2 marks] Explain how you could proceed if you don't know  $\ell$ . What is the minimum number of blocks needed now? You may assume  $\ell$  is relatively small.
- (d) [2 marks] Assume  $\ell = 2$ . Suppose that you only have access to a ciphertext  $C$  corresponding to a long English plaintext but don't know this plaintext. Explain how you could use the list of most common digrams in English below to recover the key.

*TH, HE, IN, ER, AN, RE, ED, ON, ES, ST, EN, AT,*  
*TO, NT, HA, ND, OU, EA, NG, AS, OR, TI, IS, ET,*  
*IT, AR, TE, SE, HI, OF, ...*

5. [6 marks] **Substitution-permutation networks.**


In this question, we define a new block cipher, following the substitution-permutation network design. We will call our block cipher NVS (Not Very Substitute-y), because the S-boxes will just be bit-flip operations. The resulting cipher looks something like this:



⋮

(14 more rounds for a total of 16 rounds)

It takes as input a 16-bit message block and a key, and outputs a 16-bit ciphertext. It expands the 16-bit key into sixteen 16-bit round keys  $k_1, \dots, k_{16}$ . It has 16 rounds. In each round  $i = 1, \dots, 16$ :

- the round key  $k_i$  is XORed into the state
- then each bit is flipped using the NOT operation, denoted by the following gate: 
- then a permutation is applied to rearrange the bits; the same permutation is used for all rounds.

For the purposes of this question you can assume that the permutation is sufficiently “good”, for any definition of good you need. I want you to focus on the impact of replacing the S-boxes with bit flips.

- (a) [3 marks] Describe how an adversary can totally break the 16-round NVS cipher using a single known plaintext/ciphertext pair, in much less time than a generic brute force attack on the keyspace.
- (b) [1 marks] Is a key-recovery attack possible (i.e., can the adversary recover the secret key)? Explain.
- (c) [1 marks] Would the cipher be more secure if it had 1,000,000 rounds? Why or why not?
- (d) [1 marks] Would the cipher be more secure if it had 128-bit keys and operated on 128-bit blocks (assuming a new sufficiently “good” permutation across all 128 bits)? Why or why not?

6. [8 marks] **IND-CPA security of symmetric key encryption.**

Let  $(E, D)$  be a symmetric key encryption scheme with key space  $\mathcal{K} = \{0, 1\}^\ell$ , message space  $\mathcal{M} = \{0, 1\}^m$ , and ciphertext space  $\mathcal{C} = \{0, 1\}^n$ .

We will consider the security impact of adding some checksum digits. Why would someone want to add checksum digits? Perhaps we want to ensure that any communication errors get detected.

- (a) [1 marks] Suppose  $E$  is a deterministic algorithm. Explain why  $(E, D)$  cannot be IND-CPA-secure. (This implies that, in order to be IND-CPA-secure,  $E$  must be non-deterministic.)
- (b) [3 marks] Let  $(E', D')$  be a symmetric key encryption scheme built from  $(E, D)$  as follows.
  - The key space of  $(E', D')$  is the same as for  $(E, D)$ :  $\mathcal{K}' = \mathcal{K} = \{0, 1\}^\ell$ .
  - The message space of  $(E', D')$  is the same as for  $(E, D)$ :  $\mathcal{M}' = \mathcal{M} = \{0, 1\}^m$ .
  - The ciphertext space of  $(E', D')$  is  $\mathcal{C}' = \{0, 1\}^{n+1}$ .
  - The encryption algorithm  $E'_K(M)$  is as follows:
    - 1:  $Z \leftarrow M_1 \oplus M_2 \oplus \dots \oplus M_m$  // XOR together all the message bits
    - 2:  $C \leftarrow E_K(M)$  // encrypt the message using  $K$
    - 3: **return**  $C \| Z$  // append the checksum  $C$  to the ciphertext
 Note: the notation  $c \| z$  means “concatenation”: append the bitstring  $z$  to the bitstring  $c$ .
  - The decryption algorithm  $D'_K(C \| Z)$  is as follows:
    - 1:  $M \leftarrow D_K(C)$  // decrypt the first  $n$  bits of the input
    - 2:  $Z' \leftarrow M_1 \oplus M_2 \oplus \dots \oplus M_m$  // XOR together all the message bits
    - 3: **if**  $Z = Z'$  **then return**  $M$  // check if the received checksum = the computed checksum
    - 4: **else return** “failure”

Even if  $(E, D)$  is IND-CPA-secure, it is not the case that  $(E', D')$  is IND-CPA-secure. Show how an attacker can win the IND-CPA security game for  $(E', D')$  with certainty.

(c) [4 marks] Let  $(E'', D'')$  be a symmetric key encryption scheme built from  $(E, D)$  as follows.

- The key space of  $(E'', D'')$  is the same as for  $(E, D)$ :  $\mathcal{K}'' = \mathcal{K} = \{0, 1\}^\ell$ .
- The message space of  $(E'', D'')$  is the same as for  $(E, D)$ :  $\mathcal{M}'' = \mathcal{M} = \{0, 1\}^m$ .
- The ciphertext space of  $(E'', D'')$  is  $\mathcal{C}'' = \{0, 1\}^{n+1}$ .
- The encryption algorithm  $E''_K(M)$  is as follows:
  - 1:  $C \leftarrow E_K(M)$
  - 2:  $Z \leftarrow C_1 \oplus C_2 \oplus \dots \oplus C_n$
  - 3: **return**  $C \| Z$
- The decryption algorithm  $D''_K(C \| Z)$  is as follows:
  - 1:  $Z' \leftarrow C_1 \oplus C_2 \oplus \dots \oplus C_n$
  - 2: **if**  $Z = Z'$  **then return**  $D_K(C)$
  - 3: **else return** “failure”

If  $(E, D)$  is IND-CPA-secure, it is the case that  $(E'', D'')$  is IND-CPA-secure. Prove this.

You can do this proof using the contrapositive. In other words, to show that “if  $(E, D)$  is IND-CPA-secure, then  $(E'', D'')$  is IND-CPA-secure”, we could equivalently show “if there exists an adversary SuperEve that can win the IND-CPA game for  $(E'', D'')$ , then there exists an adversary Eve that can win the IND-CPA game for  $(E, D)$ .”

Here’s the approach.

Imagine that Eve’s task is to win the IND-CPA game for  $(E, D)$  – in other words, Eve must pick two messages  $m_0$  and  $m_1$ , and then figure out whether the challenge ciphertext  $c^*$  is the encryption of  $m_0$  or  $m_1$  while being able to use a chosen-plaintext attack. Furthermore, imagine Eve has a friend SuperEve who does know how to win the IND-CPA game for  $(E'', D'')$  – in other words, SuperEve knows how to pick two messages  $m'_0$  and  $m'_1$ , and, if she is given a ciphertext  $c^{*''}$  which is the encryption of either  $m'_0$  or  $m'_1$ , SuperEve can figure whether it was  $m'_0$  or  $m'_1$ . (SuperEve might make use of *her* chosen-plaintext attack powers to have some messages encrypted using  $E''$ ). Explain how Eve can make use of SuperEve to successfully construct two messages  $m_0$  and  $m_1$  and then figure out whether  $c^*$  is the encryption of  $m_0$  or  $m_1$ , such that Eve’s runtime is small (excluding SuperEve’s runtime). In other words, explain how Eve can set things up so that SuperEve effectively solves the problem for Eve.

---

## Academic integrity rules

You should make an effort to solve all the problems on your own. You are also welcome to collaborate on assignments with other students in this course. However, solutions must be written up by yourself. If you do collaborate, please acknowledge your collaborators in the write-up for each problem. *If you obtain a solution with help from a book, paper, a website, or any other source, please acknowledge your source. You are not permitted to solicit help from other online bulletin boards, chat groups, newsgroups, or solutions from previous offerings of the course.*

---

## Due date

The assignment is due via Crowdmark by 8:59:59pm on September 28, 2023. Late assignments will not be accepted.