Introduction to Cycle Cipher

Duc Tri Dang

A project paper under MATH/COMP 4651
Supervised by Dr. Keliher
Mount Allison University

November 23, 2023

Abstract

The "Cycle Cipher" is introduced as a new cryptographic technique that seamlessly combines principles from both stream and block ciphers. This paper investigates the Cycle Cipher's underlying mechanics, focusing on its novel approach to encryption, decryption, and cryptanalysis. The Cycle Cipher represents the effort of designing a new cipher that contributes to the world of cryptography by incorporating the dynamic nature of stream ciphers and the robustness in the inspiration of multiple rounds of encryption and decryption similar to block ciphers.

Contents

1 Introduction

The quest for robust cryptographic techniques remains perpetual in the ever-evolving landscape of information security. As our reliance on digital communication intensifies, the demand for innovative encryption methods becomes increasingly paramount. This paper introduces the "Cycle Cipher," a new cipher designed on purpose for the COMP/MATH 4651 final project under Dr. Liam Keliher's supervision. The Cycle Cipher might not be an incremental advancement; rather, it represents a departure from conventional cryptographic models, offering a unique approach to securing digital information.

In this paper, we examine the Cycle Cipher's core methodology, shedding light on its distinguishing features and underlying principles. We hope to provide a comprehensive overview of the Cycle Cipher's operational framework by understanding the complexities of its encryption and decryption processes. In addition, we conduct a comparative analysis with existing ciphers to place the Cycle Cipher in the cryptographic landscape and explain its potential benefits.

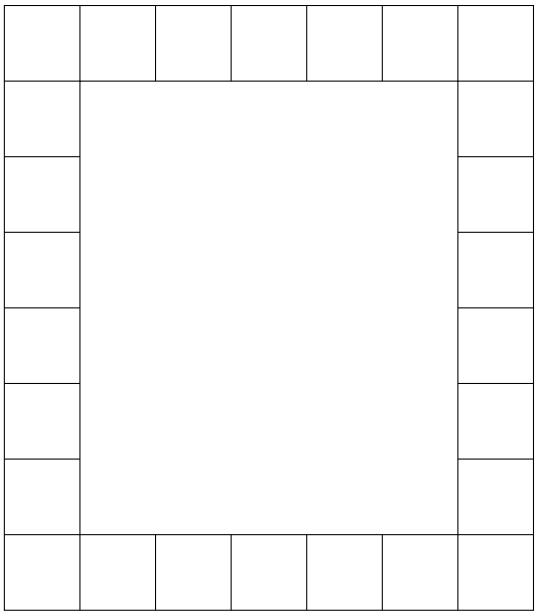
2 The Cycle Cipher

2.1 Background information

The Cycle Cipher is a cipher that encrypts and decrypts by transforming character using a specific symmetric key, based on the main ideas of creating the "fusion" between block cipher and stream cipher.

The key will include three part, an anagram with length 26 of the Alphabet string (ABCDEFGHIJKLMNOPQRSTUVWXYZ), a number from 0 to 25, and an integer of value 1 or -1 to indicate the direction.
For instance, the key of a cipher can be $= \{\text{ZBCDEFGHIJKLMNOPQRSTUVWXYA}, 19, 1\}$.

The string will be filled into a map of 26 rectangles indicates a path surrounded it in a clockwise direction.



If the key includes the anagram such as ZBCDEFGHIJKLMNOPQRSTUVWXYA, then the map will look like this:

$\longrightarrow$

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

The number in the key represents the letter correspond to the index of it in the given anagram (which is letter T). The purpose is that it indicates the beginning square of encrypting progress. The direction value is indicating the encrypting direction,1 for clockwise and -1 for counter clockwise. This will be shown clearly in the encrypting method.

2.2 One Round Encryption Method in Z_{26}

Consider a key that $K = \{\text{ZBCDEFGHIJKLMNOPQRSTUVWXYA}, 19, 1\}$. The character with index 19 in the given anagram is T. Therefore, the map will look like this:

$\longrightarrow$

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

After setting the board, these are the procedure:

1. Transfer the each character in the plaintext to code integers from 0-25.

2. The first encrypted character is formed by moving k steps starting from the chosen character from the key, which k is the integer code of the first character from the plaintext. The direction of moving steps will be based on the direction value in the key, 1 for moving clockwise and -1 for moving counter-clockwise.

3. The next encrypted characters is formed by moving k steps starting from the previous character in the ciphertext, which k is the integer code of the current character from the plaintext. Repeat this until the end.

Let's encrypt the plaintext P = "HELLO".
Let's have the key as $K = \{\text{ZBCDEFGHIJKLMNOPQRSTUVWXYA}, 19, 1\}$.
1. First, we need to transfer the plaintext to integer code.

H	E	L	L	O
7	4	11	11	14

2. The first integer code is 7. Therefore, we need to move 7 steps from the chosen start point T in clockwise direction (because the direction value is 1).

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

Therefore, the first character for the cipher text is Z.

3. Staring from Z, we will move 4 steps (The integer code for E is 4). The second letter for the cipher text is also E.

After repeating those steps, we will have the ciphertext is ZEPZO.

2.3 One Round Decryption Method in $\mathbf{Z}_{26}$

The main idea on decrypting ciphertext for one round using Cycle Cipher is based on the difference in two consecutive character in the ciphertext, starting from the back.

The procedure for the decryption are:

1. Starting from the back of the ciphertext and pick up character at the second last position and the last position. We will call the last character is A and the second last character is B.

2. Using the key to figure out the index of the character A and B on the key.

3. The result from the absolute of subtraction of the index of A and B (#index of $A-$ #index of B)

4. Transform the result into character in $\mathbf{Z}_{26}$. This will be the last character in the plaintext.

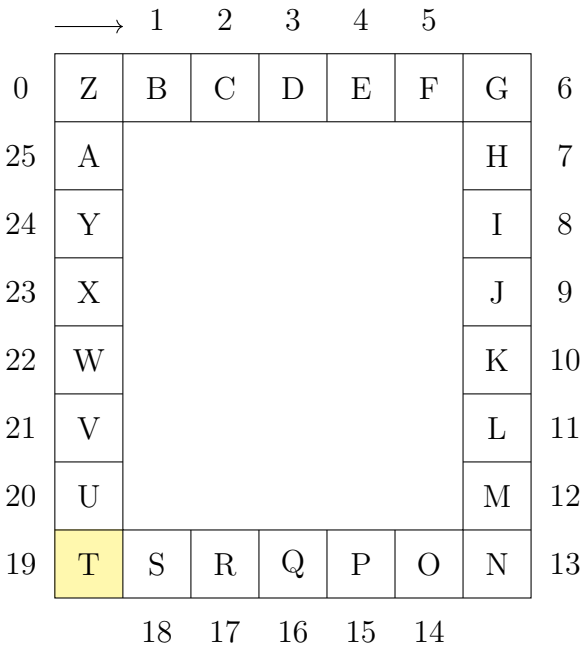
5. Continue picking the next pair (which is the last second character and third second character) until reaching the start of the ciphertext.

6. When reaching the start of the ciphertext, decrypt the pair of the position character included in the key and the first character in the ciphertext.

Note: the procedure here is going from the back to the front of the ciphertext, but it also provides the same result when going from the front to the back.

Consider the decryption process of ciphertext C = "ZEPZO".

The key will be used is $K = \{\text{ZBCDEFGHIJKLMNOPQRSTUVWXYA}, 19, 1\}$.



As the first step, the process start at the last two letters of the ciphertext, which is "Z" and "O", which corresponding to index 0 for "Z" and index 14 for "O" on the key. Therefore, we can figure that it took $14 - 0 = 14$ steps to move from index 0 to index 14. Therefore, the last character of the plaintext is 14, which can be convert to letter in $\mathbf{Z}_{26}$ is O.

After that, it will be continued with the next pair from the end, which is "P" and "Z", which corresponding to index 15 for "P" and index 0 for "Z" on the key. Therefore, we can figure that it took $0 - 15 = -15 = 11 \pmod{26}$ steps to move from index 15 to index 0. Therefore, the next character of the plaintext is 11, which can be convert to letter in $\mathbf{Z}_{26}$ is L.

Next, pairs (E,P) will be decrypted, which corresponding to index 4 for "E" and index 15 for "P" on the key. Therefore, we can figure that it took $15 - 4 = 11$ steps to move from index 4 to index 15. Therefore, the next character of the plaintext is 11, which can be convert to letter in $\mathbf{Z}_{26}$ is L.

Pairs (Z,E) will be decrypted next, which corresponding to index 0 for "Z" and index 4 for "E" on the key. Therefore, we can figure that it took $4 - 0 = 4$ steps to move from index 0 to index 4. Therefore, the next character of the plaintext is 4, which can be convert to letter in $\mathbf{Z}_{26}$ is E.

There is no other pair left, by the procedure, the pair of the starting position of the key, which is 19 (at character "T"), and the first character of the ciphertext, which is "Z" (at

index 0). Therefore, we can figure that it took $0 - 19 = -19 = 7 \pmod{26}$ steps to move from index 19 to index 0. Therefore, the next character of the plaintext is 7, which can be convert to letter in $\mathbf{Z}_{26}$ is H.

After all, the character collected as {O,L,L,E,H}, and it will be come {H,E,L,L,O} when reverse, which is the plaintext: HELLO.

Before moving on, the Keyspace K for one round of the Cycle Cipher can be calculated:

- A Permutation of $\mathbf{Z}_N$ from 0 to N-1, which is resulted to have N! numbers of permutations.
- A starting position starts between 0 and N-1, which contains N total possible cases.
- A direction value of 1 or -1.

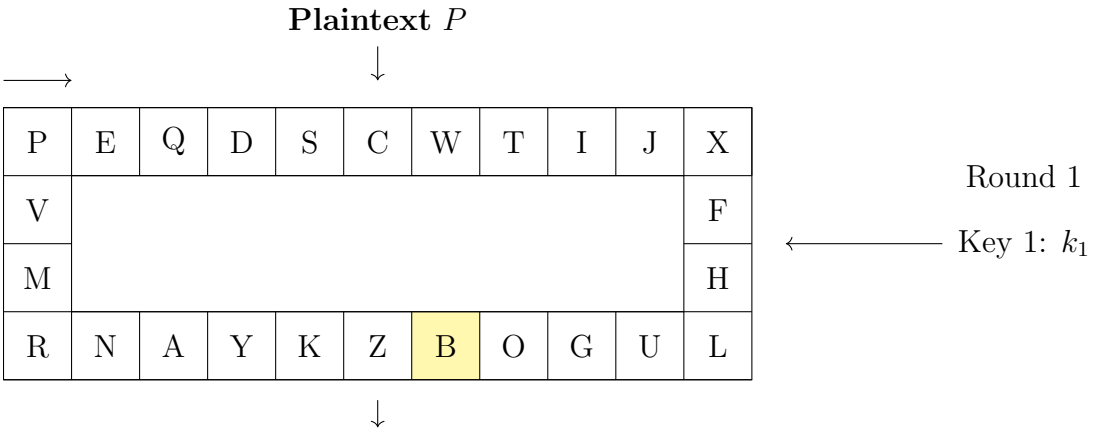
Therefore, the Keyspace K_r for one round of Cycle Cipher is:

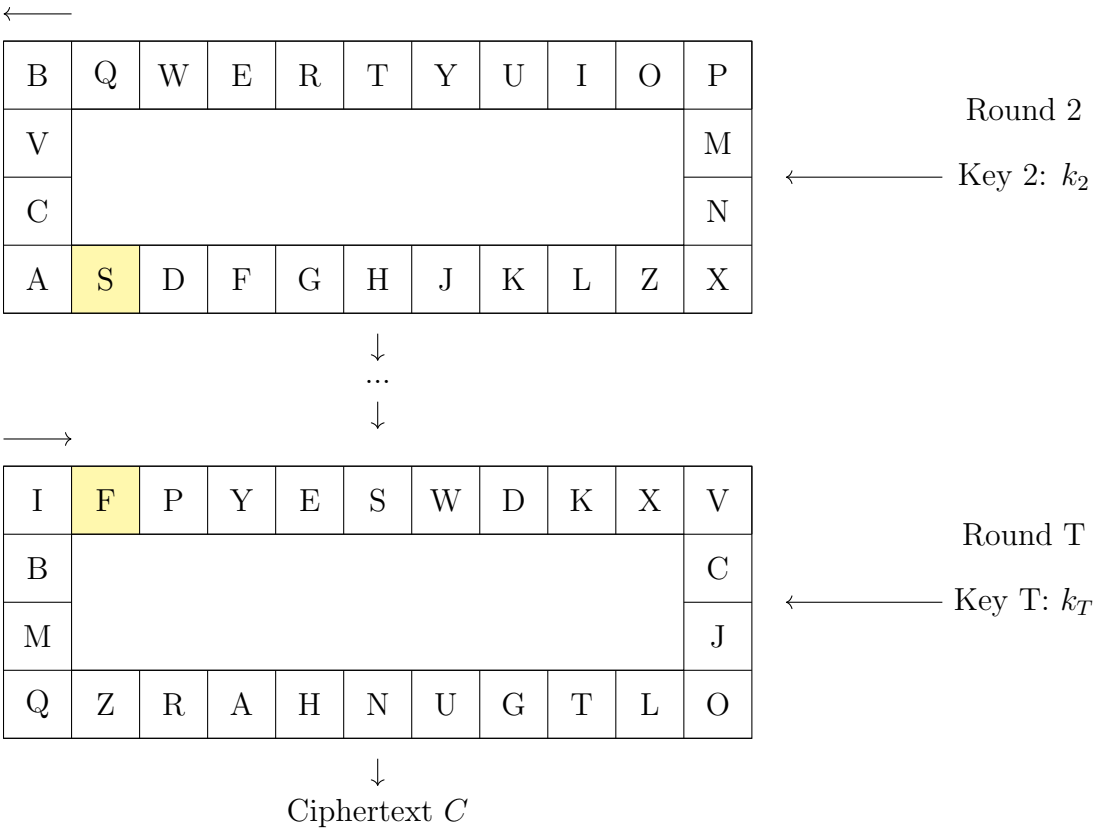
$$|K_r| = N! \times N \times 2$$

2.4 Multiple Rounds Encryption - Block Method

The cipher's vulnerability increases significantly when employing a single encryption round. To enhance its security, a multi-round approach involving iterative encryption and decryption processes is implemented. The fundamental concept revolves around the repetitive encryption of the plaintext.

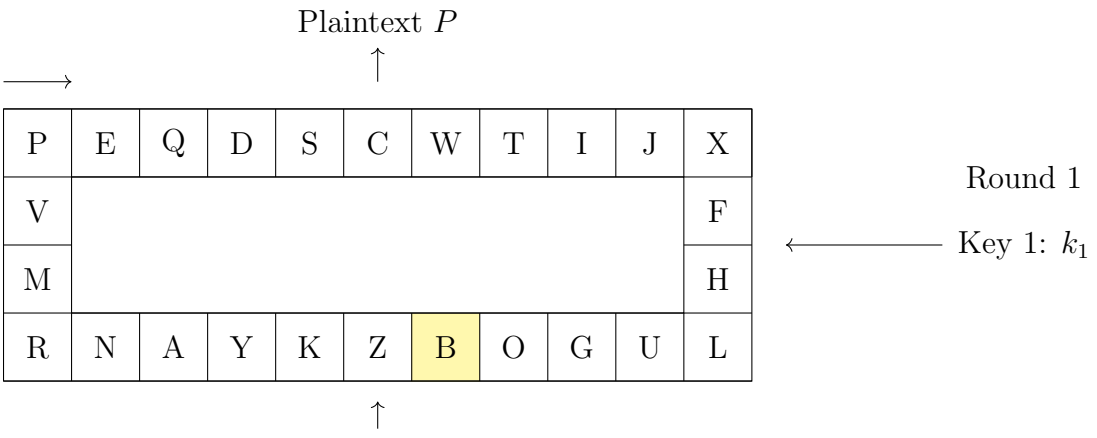
By employing T rounds, each associated with a distinct key, the plaintext undergoes encryption during the initial round. Subsequently, the ciphertext generated from the first round becomes the input for the subsequent round, perpetuating this process until Round T is attained. At this juncture, the final ciphertext is obtained, thereby fortifying the security of the cipher through a multi-layered encryption method.

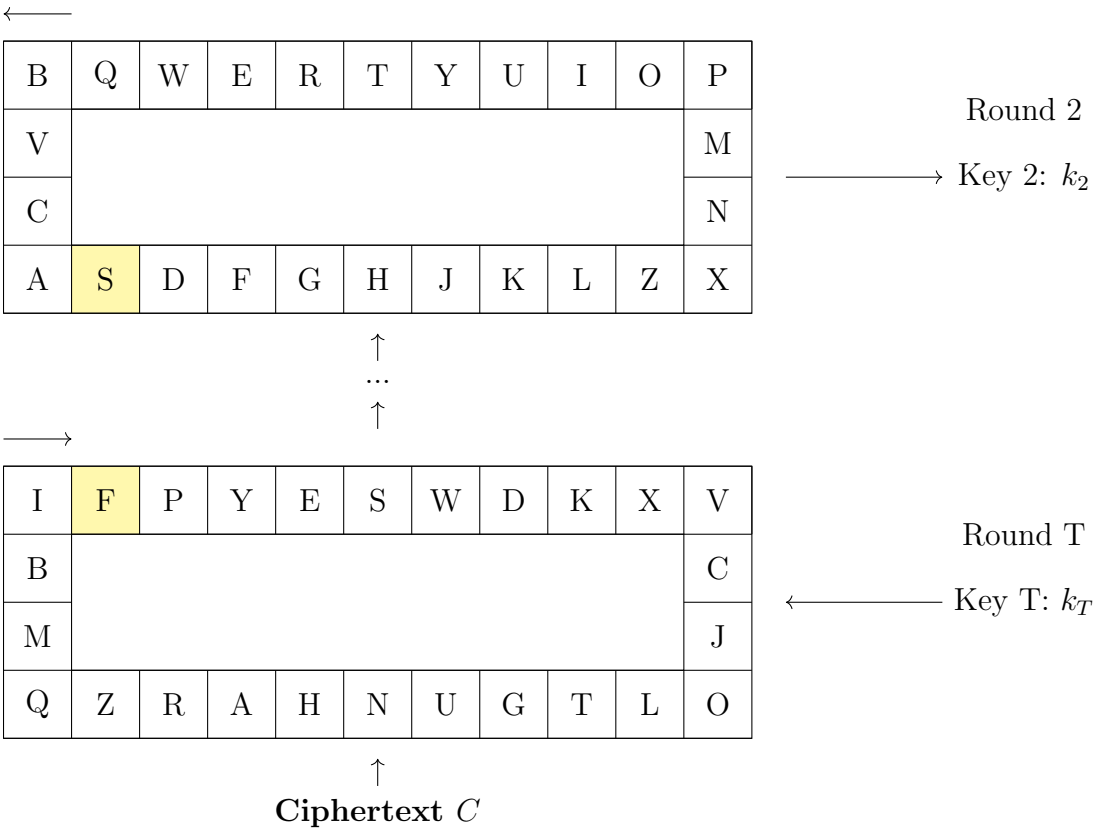




2.5 Multiple Rounds Decryption - Block Method

Similar to the encryption, given T rounds, each associated with a distinct key, the ciphertext will be decrypted round by round. The plaintext decrypted from Round T will be decrypting in the following round until reaching back to Round 1. The final plaintext will be found after the decryption of all rounds.





Because of the implementation of multiple rounds into the cipher, the Keyspace K now become more complicated. As mentioned in section **2.3**, the Keyspace K_r for one round of Cycle Cipher in $\mathbf{Z}_N$ is:

$$|K_r| = N! \times N \times 2$$

Now there are T rounds in the cipher, the total Keyspace K will become:

$$|K| = (N! \times N \times 2)^T$$

2.6 Multiple Rounds Encryption - Character by Character Method

The current iteration of the cipher presents a limitation in its encryption and decryption processes. Specifically, it necessitates access to the entire plaintext for encryption and the complete ciphertext for decryption. In practical scenarios, however, the availability of the entire plaintext or ciphertext may occur incrementally, character by character. Fortunately, a solution exists to address this challenge. The method of a continuous stream encryption method accommodates real-world applications where data is presented gradually, allowing for encryption and decryption to be conducted seamlessly on a per-character basis. The procedure can be considered as:

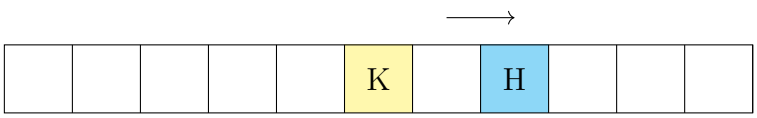
1. Starting at the first character of the plaintext and the key provided for the first round.

2. Start encrypting the first character for the ciphertext after round one and then save the result.

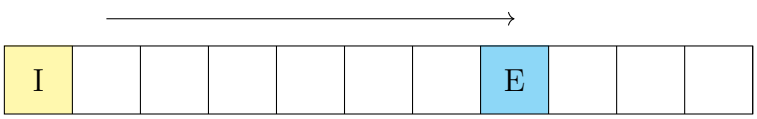
3. Start moving on to the other rounds by encrypting the saved character on the previous round's ciphertext. After that, also saved the answer and keep doing until the last round. After that, the first character will be received.

4. Move on the next character and encrypt base on the character of the plaintext and the last characters of the ciphertexts.

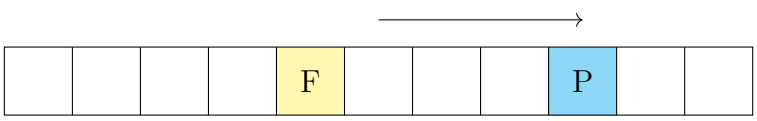
Considering encrypting character "C" in the plaintext at some position given 3 keys. The yellow spots are the last positions in every key.



Move 2 steps ("C" convert to 2)



Move 7 steps ("H" convert to 7)



Move 4 steps ("E" convert to 4)

The final encryption for character 'C" is "P". After that, the next character in the plaintext can be encrypted by starting on the last position each keys (the blue squares).

This solution effectively provides the solution without waiting for the complete assembly of plaintext prior to initiating encryption. By adopting a continuous stream approach, the cipher is adept at encryption without necessitating the entire plaintext or employing a block-to-block encryption method.

2.7 Multiple Rounds Encryption - Character by Character Method

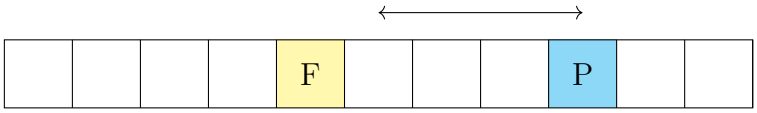
Same to the encryption, the decryption also given to be a solution of waiting a whole ciphertext by decrypting in a continuous stream. Here are the procedure:

1. We will look at the last position of the previous round and the current round. The first character does not have the last position, so we will consider the starting position included in the key.

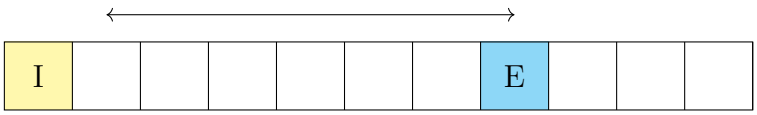
2. Starting from the final round, we will decrypt the current character in the ciphertext. The information for the founded character will help on the way to trace back through the rounds.

3. After going back through the rounds, the character for the plaintext will be decrypted. After that, the decryption can go on by decrypting the next character.

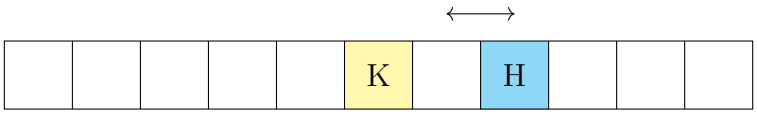
Considering the current character is "P". The last position for each round is saved (marked yellow), so the decryption will start at the last round. The character "P" and character "F" will be considered.



Need 4 steps to move from "F" to "P"
4 can be convert to "E"
"E" will be the next current character



Need 7 steps to move from "I" to "E"
7 can be convert to "H"
"H" will be the next current character



Need 2 steps to move from "K" to "H"
2 can be convert to "C"
"C" will be the final result

As could be seen from the figure, the current character is decrypted. The next character can be decrypted based on the last position for each round (marked blue).

3 Testing and Obsevation in Z_{26}

After deciding the ideas for the cipher, the coding product are quickly conducted to figure out the cryptanalysis of the cipher. More detail on the coding products will be discussed later in the paper.
A several number of tests will be conduct on both encrypting and decrypting. The key for every test include 3 rounds as following:

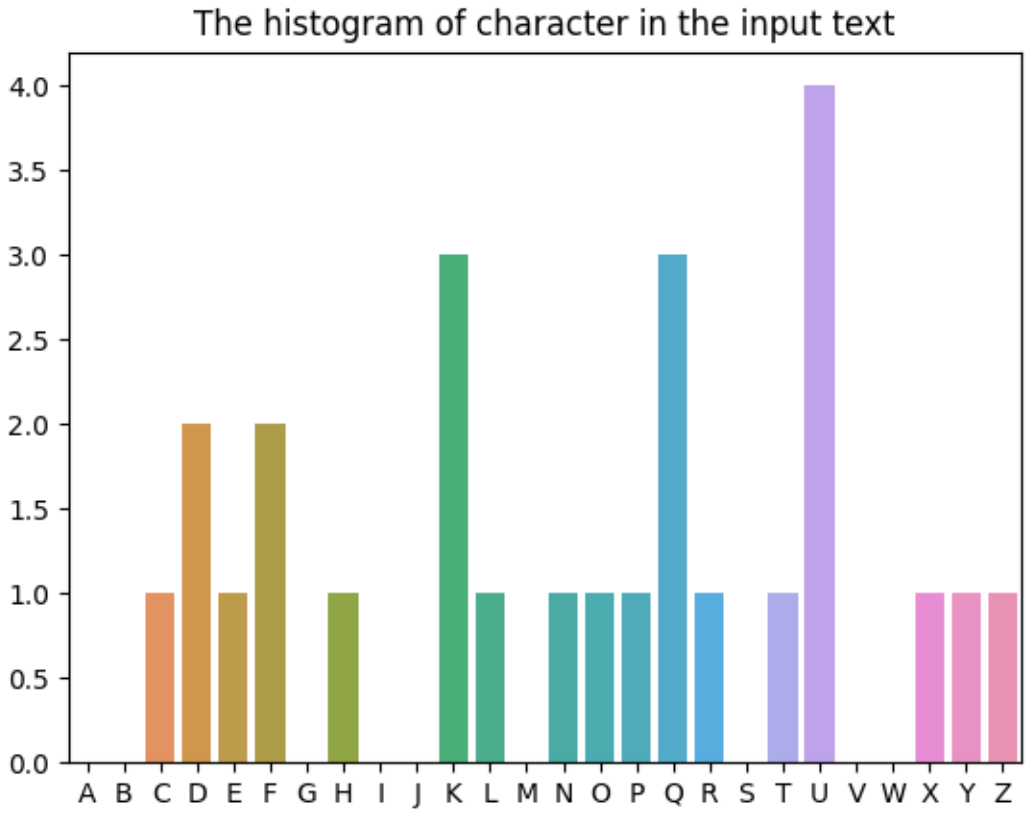
22 16 25 3 19 10 8 13 20 1 5 18 2 6 0 21 15 11 9 7 12 4 24 17 14 23
25
−1
21 0 7 20 11 23 1 19 15 4 8 10 3 17 25 16 13 18 2 14 22 24 12 9 6 5
13
1
19 5 14 4 21 23 22 3 12 24 17 0 1 11 25 15 2 13 9 8 18 20 10 16 7 6
21

–1

There are three rounds in the key, and each round will be demonstrated as three lines.

- The first line is represent an anagram with length 26 of the Alphabet string (ABCDEFGHIJKLMNOPQRSTUVWXYZ)
- A number from 0 to 25, which described the starting point of the key.
- An integer of value 1 or -1 to indicate the direction.

Test 1:
Encrypt the plaintext as the following: ABCDEFGHIJKLMNOPQRSTUVWXYZ
The ciphertext receive is: LDUFUFPHYKQKOXNUUTECQKQDZR
The decrypted text is ABCDEFGHIJKLMNOPQRSTUVWXYZ, which is exactly as the plaintext.
The histogram of the frequency in the ciphertext:



We can see the histogram is not-uniform and seems to not have any rules in the frequency at all. Because the length of the plaintext is quite short, so we have seen some characters are missing.

Test 2:

Encrypt the plaintext as the following:

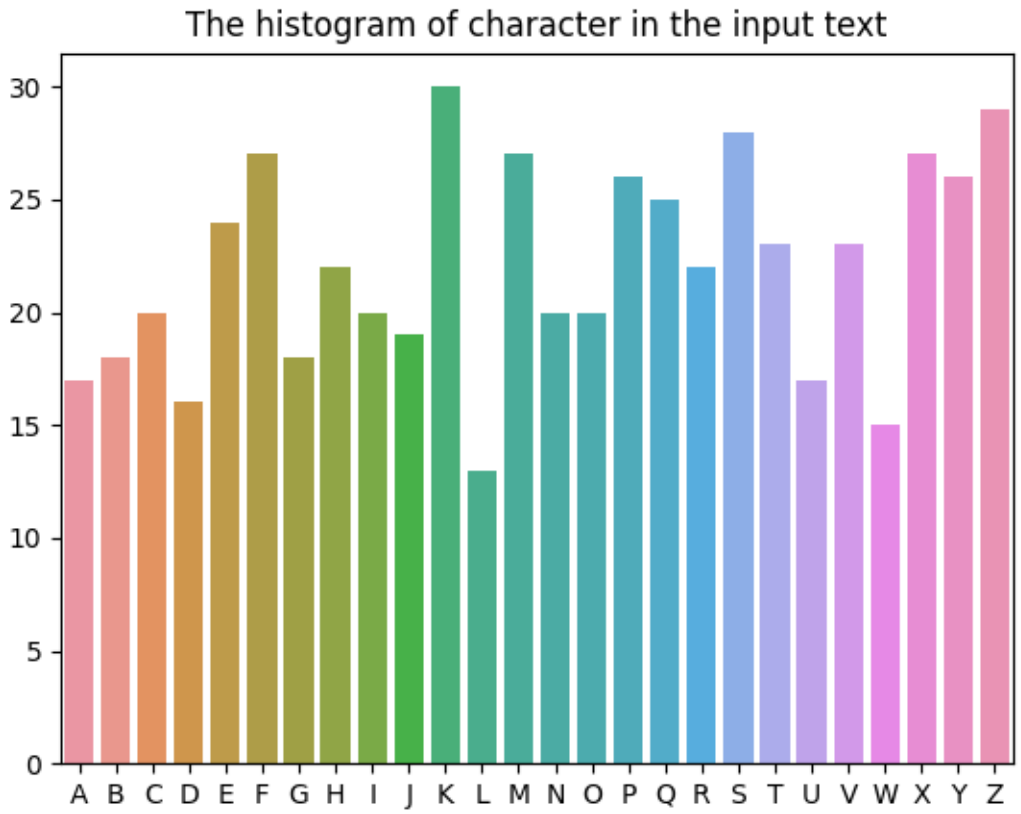
WITHATWINKLEINTHESKYNIGHTEMERGESASACOSMICCANVASSPLATTERED
WITHGLIMMERINGSTARSEACHATINYEMBLEMOFDREAMSIMAGINEASLEEPY
TOWNWHEREEVERYNOOKISACOZYHAVENANDTHENIGHTAIRWHISPERSSOFT
LULLABIESTHROUGHRUSTLINGLEAVESASTHEMOONCASTSAGLOWUPONQUIET
STREETSSLEEPBECOMESAMAGICALJOURNEYINTOAWORLDWHEREDREAMSAN
DREALITYINTERWEAVEPAINTINGVIVIDPICTURESUPONTHECANVASOFTHESLEEPING
MINDINTHISPEACEFULSYMPHONYTHENIGHTEMBRACESEVERYONEOFFERING
AWARMRESPITEFROMTHEDAYSHUSTLEASWEETSLUMBERUNFURLSANDASTHE
WORLDSLEEPSTHENIGHTSLOWLYUNVEILSITSENCHANTMENTSATAPESTRYWOVEN
WITHSTARDUSTANDTHEHUSHEDWHISPERSOFRESTFULDREAMS

The ciphertext receive is:

MBHPPSXISYUQHIOKAKFTIYJYVCHECSQCAMEGEAOYFWIWQBCJGRWT
RHTKXQGUGKDPRTXKDDNMZRNKRMDPMUHYDJHVEYXVOMESKFQHNVGZ
CYXTFZWZCSHVLZQQHWDZSYQPGMJEUJTRYPBZAQQIWYQLKZCGHJXN
IAFFSUFWPTSMEZOUUTLVOHNEEUPKNDZZOBUKVYZITIXQFPZORNTKLE
PRCQCJRHKZZLABESSYXJMZAKPFWMCNPRWIYOZVRIMICHXZJVXVFPJAPB
IREHPFWMVWXHEQETOXSMVJKTSVKJRZJHVROXXKUMWKRANMZQSJAZWFPST
POKGGRCBKBZNDJKZOJLCPOXTWTTEHSGOPNUCKQJKIQAMNSLSBPVUXCX
QFYZXXHBINTETDIVDYYTOBRYDYVCZKNHNSFFZLKJGQHAMEXNKEISNDSSC
FMXORXMVXAGSMUMDKOVBOXFGGCMDKFSIYEPBQKMYWLMNRNYILEFQLESBYM
FJIPKFPMBUSOXBBPVTFKZRFVSCDUYPYTEPUEHFNLTYZIHATFSUGGDLAXFOA
YQVRFQBRGXSAVGCQZQME

The decrypted text is tested as similar to the plaintext.

The histogram of the frequency in the ciphertext:



There are chances for all character to appear, but it is unpredictable to expect which characters will appear. It also seem has no relation between letters and letters, so we will need to have some chosen plaintext to see how it can be different.

Test 3:

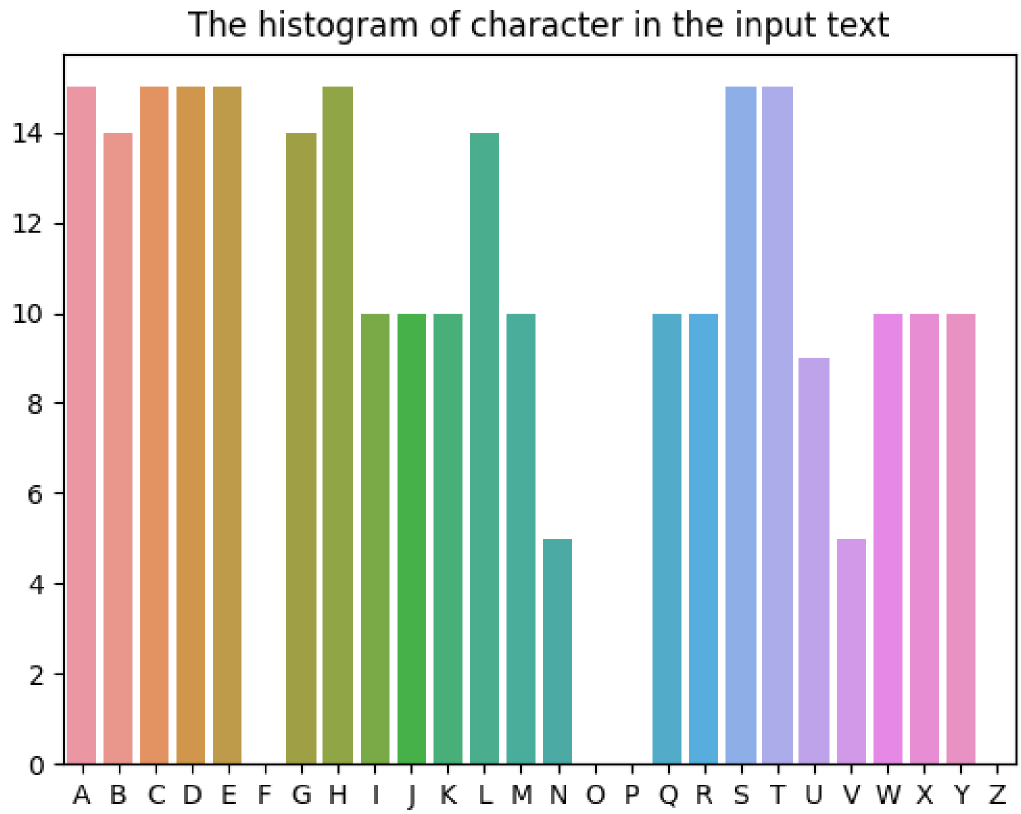
Encrypt the plaintext as 256 A characters.

The ciphertext receive is:

LSYYEXELRWXACDAISRUHNBTBGMTDKKCJCTQIJHESHWDQMAVGLGBU
LSYYEXELRWXACDAISRUHNBTBGMTDKKCJCTQIJHESHWDQMAVGLGBU
LSYYEXELRWXACDAISRUHNBTBGMTDKKCJCTQIJHESHWDQMAVGLGBU
LSYYEXELRWXACDAISRUHNBTBGMTDKKCJCTQIJHESHWDQMAVGLGBU
LSYYEXELRWXACDAISRUHNBTBGMTDKKCJCTQIJHESHWDQMAVG

The decrypted text is tested as similar to the plaintext, which is 256 characters A.

The histogram of the frequency in the ciphertext:



We have seen some repetitions in the ciphertext. It can be explained by the repetition feature of a stream cipher, but it still be quite unpredictable and has no relationship between the characters.

Test 4:

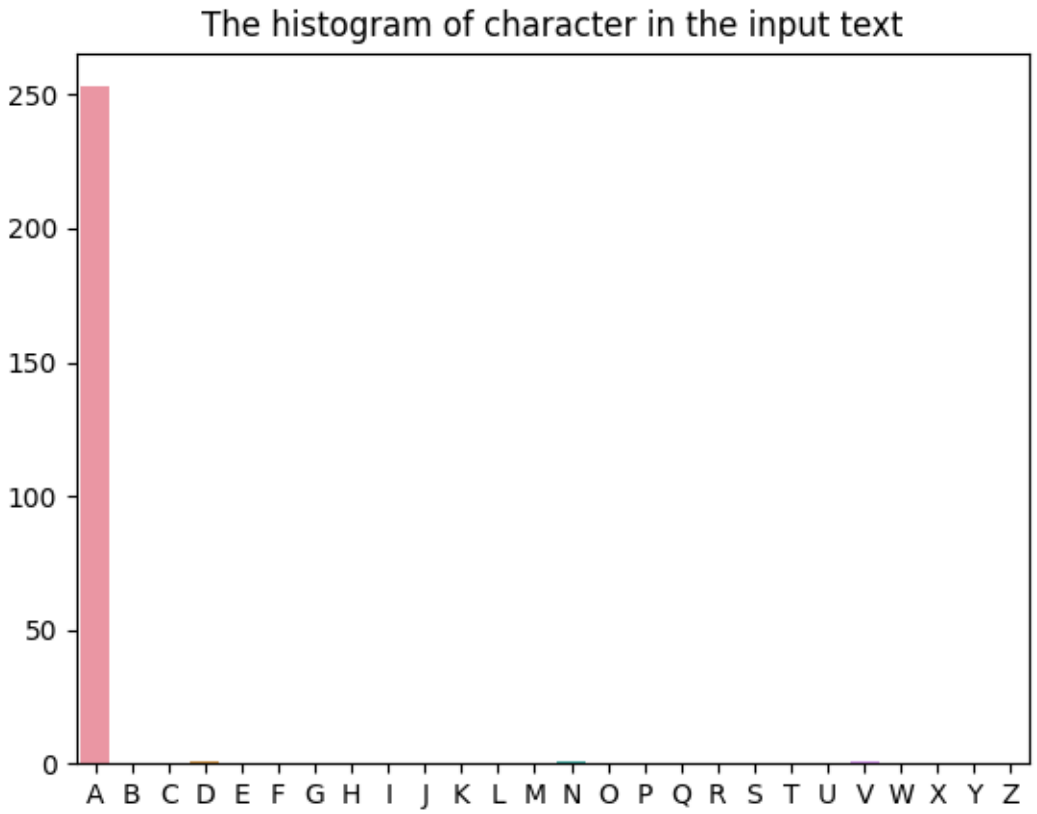
Test 4 is made totally by accident, but it has opened up an opportunity for the cryptanalysis of the cipher. This time, just only the decryption was conducted.

The ciphertext is 256 characters A. We will expect the result of the decrypted message will be somewhat unpredictable, or completely random, just like the encryption.

The decrypted text is: DVNAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA

When encrypt the decrypted message, the exact ciphertext is received, which is 256 characters A.

The histogram of the frequency in the ciphertext:



The observation of three distinct characters coinciding with the three provided rounds presents an intriguing pattern that could potentially serve as a crucial element in deciphering the encrypted message. The identification of any discernible relationship within the decrypted content may indeed pave the way for a successful cryptanalysis.

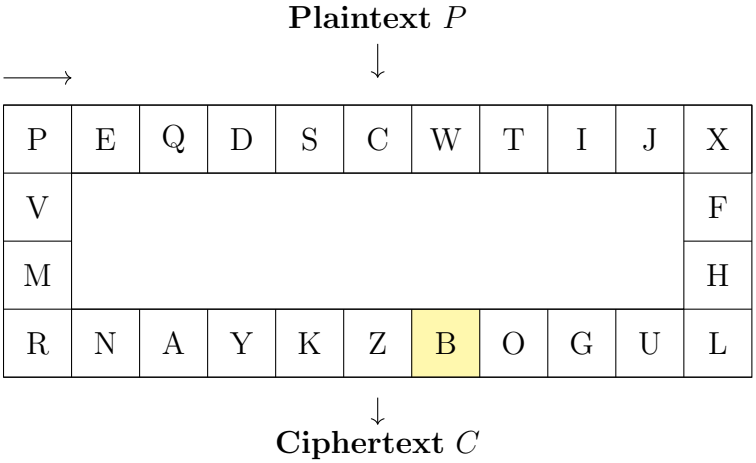
Overall, the conducted assessments affirm the efficacy of the cipher's encryption and decryption processes. Additionally, these findings underscore the significance of embarking upon a series of systematic experiments designed to dealing with cryptanalysis.

4 Cryptanalysis

With a huge Keyspace $|K| = (N! \times N \times 2)^T$, some brute-forcing techniques will be falied, such as Meet-in-the-middle attack. However, the cryptanalysis of the Cycle Cipher can be thought of by using the feature of moving characters around.

4.1 Cryptanalysis On One Round

Given one round of the Cycle Cipher, such as:



As could be seen, it is not really hard to get the key of one round Cycle Cipher. The concept of the chosen-plaintext attack could be used in this case.

The value of every character demonstrates the movement of characters. Therefore, it can be used wisely to move around the board and take the key easily.

To go around the key, we will need a string that includes the character to visit every square once before it repeated. The string with size N that contains the same character will work, especially the character that convert to x that $x \in \mathbb{Z}_{26}^*$, so the characters will be all visited before repeated. Therefore, $x \in \{1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25\}$ is valid, which convert to x can be any in{B,D,E,H,J,L,P,R,T,V,X,Z}. However, the easiest way is choosing B to have a string of size N full with B, so the ciphertext will be exactly the key in order.

For example, encrypt the string of 26 characters B = "BBBBBBBBBBBBBBBBBBBBBB BBBBB", the ciphertext will be the key as "ZKYANRMVPEQDSCWTIJXFHLUGOB".

4.2 Cryptanalysis On Multiple Rounds

4.2.1 Trial on Cryptanalysis Ideas

The cryptanalysis of Cycle Cipher will be experimented base on test 4 in section 3: Testing and Observation in $\mathbf{Z}_{26}$.

A multiple of ciphertexts are chosen, which mostly are the string include multiple numbers of the same character from A to Z. The length of the ciphertexts will need to be equal or greater than the number of rounds.

The key used in this section will be the key in section 3. Here is the table for the result of the experiments:

Plaintext	← After round 1	← After round 2	← After round 3 = Ciphertext
DVNAAA	YQAAAA	KAAAAA	AAAAAA
UKHAAA	KEAAAA	JAAAAA	BBBBBB
FIYAAA	MCAAAA	FAAAAA	CCCCCC
MHSAAA	GIAAAA	OAAAAA	DDDDDD
EQRAAA	EKAAAA	SAAAAA	EEEEEE
YFIAAA	QYAAAA	UAAAAA	FFFFFF
GAFAAA	HHAAAA	WAAAAA	GGGGGG
OLMAAA	SWAAAA	XAAAAA	HHHHHH
PSEAAA	FJAAAA	CAAAAA	IIIIII
XMCAAA	ZPAAAA	DAAAAA	JJJJJJ
QCTAAA	BNAAAA	ZAAAAA	KKKKKK
ACJAAA	XRAAAA	IAAAAA	LLLLLL
WMDAAA	DLAAAA	NAAAAA	MMMMMM
ZPXAAA	WSAAAA	EAAAAA	NNNNNN
RAUAAA	UUAAAA	TAAAAA	OOOOOO
IOPAAA	LDAAAA	GAAAAA	PPPPPP
TTZAAA	IGAAAA	YAAAAA	QQQQQQ
CYLAAA	RXAAAA	LAAAAA	RRRRRR
VPBAAA	TVAAAA	BAAAAA	SSSSSS
SYVAAA	NBAAAA	VAAAAA	TTTTTT
BKAAAA	OAAAAA	AAAAAA	UUUUUU
LQKAAA	AOAAAA	RAAAAA	VVVVVV
KLQAAA	VTAAAA	PAAAAA	WWWWWW
NSGAAA	CMAAAA	QAAAAA	XXXXXX
HIWAAA	JFAAAA	MAAAAA	YYYYYY
JOOAAA	PZAAAA	HAAAAA	ZZZZZZ

As could be seen from the table, after every round except the last round, the last non-A character will turn into an A. This can be explain that because every character is the same, so at the end, the first character after round 2 will reach that the first character in the ciphertext and the As will keep that character to be repeated.

It is worth to say that when look at UUUUUU, it is easy to figure out the key for the final round because the ciphertext of it turned to all As before the final round. However, the attackers know U will be the letter of the starting point in the final key, but the attackers do not know where is letter U in the key. One note that in real attacking, the attackers do not see the result after round 1 and round 2. Therefore, it is clueless to based on this idea to attack the cipher.

Basically, three rounds or more will have the same concept and the computation method. In here, the most basic concept of three rounds will be discuss.

Considering a key of r rounds, note as K_1,K_2, up to K_r, and in this case, $r = 3$ will be mainly discussed. Consider the plaintext $P = x_1x_2x_3AAA...$, which including r different characters will be different, followed by a sequence of As. The ciphertext is $C = yy...y$, with

y is a character from A to Z. In a key K_i, the starting point will be notated as s_i and the direction is notated as d_i. Considering three rounds of the Cycle Cipher:

- First round:
  - The first character in the ciphertext after first round will be:

$$K_1[s_1 + d_1x_1]$$

  - The second character in the ciphertext after first round will be:

$$K_1[s_1 + d_1(x_1 + x_2)]$$

  - The third character in the ciphertext after first round will be A, which is converted to 0 as observation:

$$K_1[s_1 + d_1(x_1 + x_2 + x_3)] = 0$$

  - Everything else will be A, which is converted to 0.

- Second round:
  - The first character in the ciphertext after second round will be:

$$K_2[s_2 + d_2K_1[s_1 + d_1x_1]]$$

  - The second character in the ciphertext after second round will be A, which is converted to 0 as observation:

$$K_2[s_2 + d_2(K_1[s_1 + d_1x_1] + K_1[s_1 + d_1(x_1 + x_2)])] = 0$$

  - Everything else will be A, which is converted to 0.

- Third round:
  - The first character in the ciphertext after third round will be the chosen character y:

$$K_3[s_3 + d_3K_2[s_2 + d_2K_1[s_1 + d_1x_1]]] = y$$

  - Everything else will be the chosen character y.

Overall, here are the main things:

$$\begin{cases} K_1[s_1 + d_1(x_1 + x_2 + x_3)] = 0 \\ K_2[s_2 + d_2(K_1[s_1 + d_1x_1] + K_1[s_1 + d_1(x_1 + x_2)])] = 0 \\ K_3[s_3 + d_3K_2[s_2 + d_2K_1[s_1 + d_1x_1]]] = y \end{cases}$$

Note that: At one specific point (with $y = U$ in this case), we know that things will be a little different:

- After the first round, the second character in the ciphertext will be A, which is converted to 0:

$$K_1[s_1 + d_1(x_1 + x_2)] = 0$$

  In here, x_1 is B, which is converted to 1, and x_2 is K, which is converted to 10.

$$\Leftrightarrow K_1[s_1 + d_1(11)] = 0$$

- After the second round, the first character in the ciphertext will be A, which is converted to 0:

$$K_2[s_2 + d_2K_1[s_1 + d_1x_1]] = 0$$

  In here, x_1 is B, which is converted to 1, and x_2 is K, which is converted to 10.

$$K_2[s_2 + d_2K_1[s_1 + d_1]] = 0$$

There are nothing to be cancelled out or seems to be a nicer way to solve these besides brute-forcing. Brute-forcing approach will be quite long because of the huge Keyspace, and in case of $r = 3$ and $N = 26$, then:

$|K| = (26!\times 26\times 2)^3 = 9222891750255385398361588883735255292604219769585374423848459436032 00...$

It is around 9.2×10^{84} cases to attempt, which is really unrealistic and not efficient. It is really difficult to attack the Cycle Cipher, but what happen if there are only two rounds. We know that there is one cases that when y is the letter of the starting point, it will be a little bit easier. Is there anyway to make the problem less complicated by doing that way?

4.2.2 Cryptanalysis on Two Rounds

Same to the previous section, the experiment will also be conducted with $r = 2$.

Plaintext	← After round 1	← After round 2 = Ciphertext
BKAAAA	OAAAAA	AAAAAA
VQAAAA	TAAAAA	BBBBBB
PWAAAA	FAAAAA	CCCCCC
XOAAAA	ZAAAAA	DDDDDD
ZMAAAA	WAAAAA	EEEEEE
FGAAAA	MAAAAA	FFFFFF
IDAAAA	LAAAAA	GGGGGG
JCAAAA	PAAAAA	HHHHHH
ALAAAA	XAAAAA	IIIIII
URAAAA	KAAAAA	JJJJJJ
DIAAAA	YAAAAA	KKKKKK
CJAAAA	RAAAAA	LLLLLL
HEAAAA	JAAAAA	MMMMMM
WPAAAA	DAAAAA	NNNNNN
MZAAAA	GAAAAA	OOOOOO
KBAAAA	VAAAAA	PPPPPP
NYAAAA	CAAAAA	QQQQQQ
LAAAAA	AAAAAA	RRRRRR
EHAAAA	EAAAAA	SSSSSS
RUAAAA	UAAAAA	TTTTTT
YNAAAA	QAAAAA	UUUUUU
STAAAA	NAAAAA	VVVVVV
GFAAAA	HAAAAA	WWWWWW
OXAAAA	SAAAAA	XXXXXX
TSAAAA	IAAAAA	YYYYYY
QVAAAA	BAAAAA	ZZZZZZ

Same as before, it is easy to figure that the second character in the ciphertext after round 1 will turn to A and the first character in the ciphertext after round 2 will turn to the chosen character y.

$$\begin{cases} K_1[s_1 + d_1(x_1 + x_2)] = 0 \\ K_2[s_2 + d_2 K_1[s_1 + d_1 x_1]] = 0 \end{cases}$$

Especially, there is one case with y is R, which is the character for the starting position for key 2, it can be seen that the first character in the ciphertext after round 1 turn into an A. This can be a huge clue to solve the key for two rounds Cycle Cipher.

$$K_1[s_1 + d_1 x_1] = 0$$

In this case, x_1 is L, which is 11, so then:

$$K_1[s_1 + d_1(11)] = 0$$

However, when constrain it down into one formula, it is quite hard to tell because there are a lot of keys that satisfy $K_1[s_1 + d_1(11)] = 0$. Therefore, it is seems quite difficult to attack the Cycle Cipher.

5 Coding Products

The coding product will be separated into two parts, which are Cycle Ciphers conducting in $\mathbf{Z}_{26}$ and $\mathbf{Z}_{256}$.

5.1 Coding product in $\mathbf{Z}_{26}$

The coding products in $\mathbf{Z}_{26}$ will include a process of encryption, decryption, a key generator, and a program that checking the frequency of the text. In $\mathbf{Z}_{26}$, all plaintexts and ciphertexts just only include capital letters from A to Z. Furthermore, the file to be encrypted and decrypted must end in **.txt**

The encryption process: The program will return a ciphertext file from a file includes the plaintext and the key. The ciphertext file does not need to be created in advance. The complexity of the program should be O(n). The structure of command line will be:

```
python .\Z26_cyclecipher_encryption.py .\plaintext_file.txt
.\ciphertext_file.txt .\key_file.txt
```

The decryption process: The program will return a decrypted text file from a file includes the ciphertext and the key. The decrypted text file does not need to be created in advance. The complexity of the program should be O(n). The structure of command line will be:

```
python .\Z26_cyclecipher_decryption.py .\ciphertext_file.txt
.\decrypted_text_file.txt .\key_file.txt
```

Key generation: The program will return a key file with the number of rounds. The file for the key does not need to be created in advance. The structure of command line will be:

```
python .\key_generator.py .\key_file.txt number_of_rounds
```

The frequency of the text: The program will return a graphic demonstrate the histogram of the text from a given text file. The structure of command line will be:

```
python .\frequency_ciphertext.py .\text_file.txt
```

5.2 Coding product in $\mathbf{Z}_{256}$

The coding products in $\mathbf{Z}_{256}$ will include a process of encryption, decryption, and a key generator. In $\mathbf{Z}_{256}$, the plaintext can be any file to be encrypted to a **.cyc** file, and the

decryption can also turn any **.cyc** file into the original file. However, the key still need to keep in **.txt**.

The encryption process: The program will return a ciphertext file from a file includes the plaintext and the key. The ciphertext file does not need to be created in advance. The complexity of the program should be O(n). The structure of command line will be:

```
python .\Z26_cyclecipher_encryption.py .\plaintext_file
.\ciphertext_file .\key_file.txt
```

The decryption process: The program will return a decrypted text file from a file includes the ciphertext and the key. The decrypted text file does not need to be created in advance. The complexity of the program should be O(n). The structure of command line will be:

```
python .\Z26_cyclecipher_decryption.py .\ciphertext_file.txt
.\decrypted_text_file.txt .\key_file.txt
```

Key generation: The program will return a key file with the number of rounds. The file for the key does not need to be created in advance. The structure of command line will be:

```
python .\key_generator.py .\key_file.txt number_of_rounds
```

There is no frequency check for $\mathbf{Z}_{26}$ because the histogram will be huge and it cannot be indicated through any graphics.

6 Confidentiality And Conclusion

According to KirchHoff's Principle, a cryptosystem should be secure even if everything about the system, except the key, is public knowledge. Applied to the Cycle Ciphers, which appeared even centuries later than the principle, but still include something right about it. The keyspace of the cipher is large enough to be secure within brute-forcing approach. Despite the suspcious ability of attacking on chosen-ciphertext attack, it has demonstrated how hard it is to find a way to attack.

Overall, as a reflection, I had a great time working on this cipher, beginning with days spent staring at a whiteboard in the library to generate ideas for Cycle Cipher and ending with days spent in Halifax and Sackville finishing the code and attempting to attack it. I'd love to have more time to work on and successfully attack the cipher, but that may be left to the readers, or even the next generations of Mount Allison students, when Dr. Keliher mentions it in MATH/COMP 4651 the next time it's offered. In this project, I learned a lot of new things, especially trying on different roles, such as cipher designer trying to come up with ciphers, programmer making the code for this to run, and attacker trying to attack the cipher. It requires me to shift my mindset and adapt well to each position.

Last but not least, I want to acknowledge the support from my course instructor, Dr. Liam Keliher. Furthermore, I also acknowledge the second and third name of the cipher, which is Pizza Cipher from Sawyer Stanley and Monopoly Cipher from Logan Pipes.