Playfair Cipher : An In-Depth Analysis and Time Complexity Evaluation

Heet Dave
Department of Computer Science and Engineering
Institute of Technology
Nirma University
Ahmedabad , Gujarat
23MCD002@nirmauni.ac.in

***Abstract*—This paper is devoted to studying the time complexity evaluation of cipher's workings and its time complexity, shedding light on its strengths, weaknesses, and historical significance. This paper mainly discusses the key points of this paper. The central theme of this paper revolves around the meticulous evaluation of time complexity. It is imperative to gain an in-depth understanding of the time required for the execution of various components of the cipher. This assessment extends to key aspects of the cipher, including the key setup, message preparation, and encryption/decryption processes, allowing for a profound insight into its efficiency.**

***Index Terms*—Playfair Cipher ,Cryptography, Playfair Cipher Time Complexity**

I. Introduction

Cryptography is a study of the principals and methods of transforming meaningful / contextual message to non-contextual message or text , and then retransforming that message back to its original form

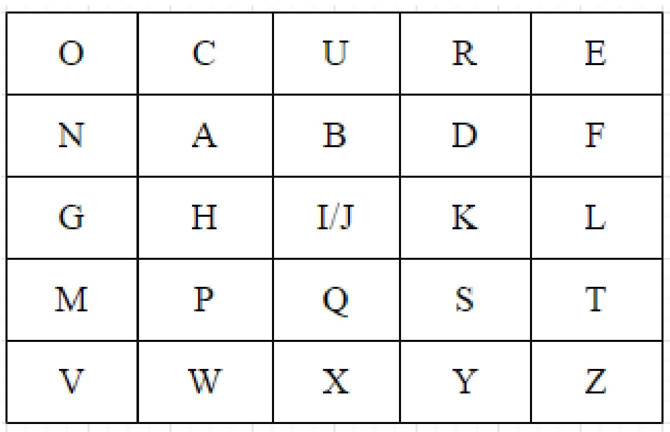
The Playfair Cipher holds a special place in the history cryptography as one of the earliest ciphers to use bigrams, which provided a new dimension of complexity to cryptographic techniques during its time.The Playfair algorithm is based on a 5 × 5 matrix (key) of letters.The Playfair Cipher belongs to symmetric cipher , because it uses single key for both encryption and decryption.It uses substitution technique.The matrix is constructed by filling in the letters of the keyword(minus duplicates) from left to right and from top to bottom, and then filling in the remainder of the matrix with the remaining letters in alphabetic order. The letters I and J count as one letter.It often called block cipher because it encrypt block of message text / plain text with cipher text

The Playfair cipher was considered unbreakable during the war. It is also known as "spy ready cipher," considering its simplicity, ease of use, easy to understand, and no machine required [13].

II. Key Elements of the Playfair Cipher

A. Key Setup

For Key , we create 5X5 grid matrix contains all alphabets , and in one cell we consider i/j both at same place , the generate of this key table is simple , one key text / keyword is provided to us , so the first step is to put all keyword letters



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

Fig. 1. 5X5 Key Table

in each cell of the 5X5 table , without any repetition we have to consider same letter only once , and then rest of the cell's are filled by the remaining letters of alphabets
For Example
Keyword :- OCCURRENCE
Plain Text : - TALL TREES

Here if we Observe Keyword their is multiple repetition of same letter, but it will only consider once in key table formation

B. Message Preparation

Playfair Cipher , treats digrams (two letters) in the plain text as single units and translates these units into cipher text digrams.Make Pairs of letters add filler letter "X" if same letter appears in a pair

Plaintext= TALL TREES
Plaintext= "TA" "LX" "LT" "RE" "ES"

If there is an odd number of letters in plain text, then add uncommon letter to complete digram, a X/Z may be added to the last letter.

C. Encryption Rules

Cipher Text Generation Follows this Rules : -

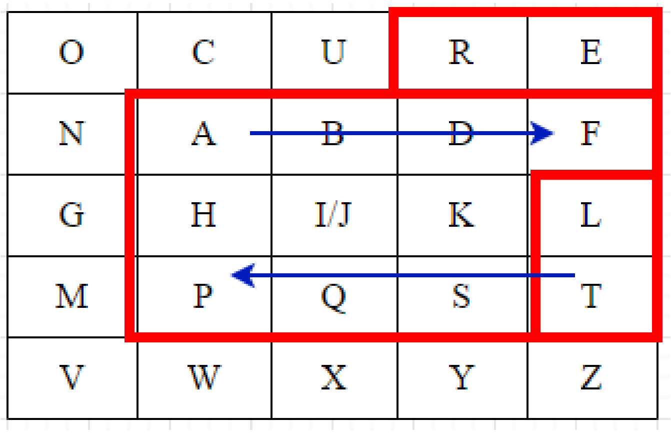


Fig. 2. After Encryption Rules

- If both the letters of selected pair are in same column then ,choose the letter exact below of each one , and if selected letter from pair is lies at last element of column , then go to the top of same column and choose it
- If both the letters of selected pair are in same row then , choose the letter very right of each one , and if selected letter from pair is lies at last element of last row and their is no element on right side of it , then go to the leftmost first element of that row and choose it
- If any of the above rule is not applicable , then the rectangle of selected pair of two letters , and choose the letter on horizontal opposite of corner of selected letter pair

Plain Text :- "TA LX LT RE ES"
Cipher Text:- "PF IZ TZ EO RT"

D. Decryption Process

Set Rules for Generation Plain text from Cipher Text:-

- If both the letters of selected pair are in same column then ,choose the letter exact above of each one , and if selected letter from pair is lies at first element of column , then go to the bottom / last element of same column and choose it
- If both the letters of selected pair are in same row then , choose the letter very left of each one , and if selected letter from pair is lies at first element of row and their is no element on left side of it , then go to the rightmost first element of that row and choose it
- If any of the above rule is not applicable , then the rectangle of selected pair of two letters , and choose the letter on horizontal opposite of corner of selected letter pair

Cipher Text:- "PF IZ TZ EO RT"
Plain Text:-"TA LX LT RE ES"

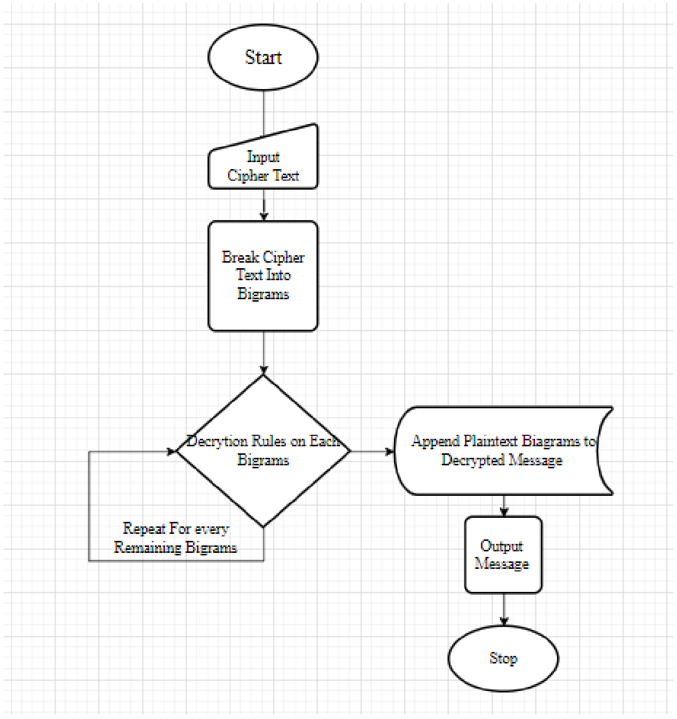


Fig. 3. Decryption Process [7]

III. Merits and Demerits of Playfair Cipher

A. Security Provided BY Playfair Cipher

It operates on pair of letter rather than individual letters , making it more resistant to the frequency analysis attacks [16].Early it was useful but it has some security limitations like known-plain text attacks , brute-force attack ,weak keys etc [10].

many improved version are proposed based on playfair cipher to improve its security and performance by using extended metrices [13] [2] [12] , multi dimension matrices [3] [8] or by improving its algorithm [6] [11]

IV. Time Complexity Analysis

To Understand or for evaluating the time complexity ,we need an algorithm of playfair cipher , basis on the working of this algorithm we can reach to any approx time complexity

For evaluating total time complexity of playfair cipher , thier is mainly three process we do in playfair cipher
Step-1:- Key Setup
Step-2:- Message Preparation
Step-3:- Encryption/Decryption

First we have to Calculate time complexity of all three steps individually by understanding its working of algorithm , and at final we have to sum all this time complexity to get approx time complexity of playfair cipher [4]

A. Time complexity Of Key Setup

Algorithm [1]

TABLE I
COMPARISON WITH OTHER SUBSTITUTION TECHNIQUE CIPHERS [15]

Parameter	Caesar cipher	Playfair Cipher	Hill cipher	Polyalphabetic Cipher
Key Type	Substitution	Substitution	Substitution	Substitution
Block Size	1	2	m	Variable length
Key Size	Fixed Number	Fixed (25!)	Variable	Equal to message Length
Attack Type	Brute Force Attack	Cipher Text Only	Known Plain text Attack	Cipher Text and Plain text attack
Algorithm Strength	Only 25 keys possible	26*26=676 diagrams possible	Hide single letter frequency distribution	multiple cipher text letters for each plaintext letter
Key Factor (Uniqueness) about the technique	Simple substitution with Alphabet	Use pair of letters and substitute with 5×5 matrix designed with key and remaining alphabets	Based on Linear algebra, Convert plaintext into matrix based on ASCII value	Plaintext is written downwards on successive "rails" of an imaginary fence, then moving up when we get

```
                                  cost/Times
# Define the alphabet,
excluding 'j'
alphabet=
'abcdefghiklmnopqrstuvwxyz' ----->c1/O(1)

# Remove whitespace and 'j' from
the key and convert to lowercase
key=key.lower().replace(' ', '')
.replace('j', 'i') -------------->c2/O(K)

# Construct the key square
key_square = ''
for letter in key + alphabet:-->c3/O(K+A)
    if letter not in key_square:
                               ->c4/O(K+A-1)
        key_square += letter
                               ->c5/O(k+A-1)
```

First lets discuss some Expression we have used in algorithm , In O(K) , where K is length of Key / Keyword , in this operation we are just removing blank space and letter j from keyword and making it in lowercase , which depends on the length of k so its complexity is O(K)

Now for Constructing Key Table/Matrix , The for loop starts iteration of every letter in addition of key + alphabets , which gives total length is Length of Key And Length of Alphabets so its execution goes till (K+A) , where A is length of Alphabets and K is Length of Key , so we can say the time required is O(K+A)

Inside that for loop , we have if condition which checks for duplication of letter , if select letter is not duplicate or means occurring very first time then it will be added to the matrix , this condition is going to check for K+A-1 time so , its complexity is O(K+A-1) , if if statement is correct then that much time key sqaure is going to be done , so it also takes O(K+A-1)

So BY Adding Up the complexity of all individual statement we get our complexity for key setup

T(K) = c1 + c2K + c3(K+A) + c4(K+A-1) + c5(K+A-1)
T(K) is time complexity of Key Setup from above equation we can conclude the dominating term is **O(K+A)** which is consider time complexity of Key setup

B. Time Complexity of Message Preparation

Algorithm

```
                                  cost/Times

# Split the plaintext into digraphs,
padding with 'x' if necessary
plaintext = plaintext.lower().
replace(' ', '').
replace('j', 'i')     ------------>c1/O(N)

if len(plaintext) % 2 == 1: -----> c2/O(1)
    plaintext += 'x'

digraphs = [plaintext[i:i+2]
for i in range(0, len(plaintext), 2)]
                                 -------->c3/O(N)
```

Here what we are doing is , In first Statement we are removing blank spaces in plaintext / message and making it lower case and changing letter 'j' with 'i' , this statement will execute for N times , N is length of Plaintext / Message , so its complexity is O(N)

Now we are executing If statement in which it checks whether the plaintext has even length or no if not , then it will add letter 'x' to make its length even , this statement is only execute single time so its complexity is O(1)

At Last we are creating Diagraphs of Plaintext , for which we have to traverse till the length of plaintext , so it will take O(N) time to execute , where N is length of Plaintext

Now By adding up the complexity of this statement

T(M) = c1N + c2 + c3N

T(M) is the Time complexity of message preparation
BY above equation we can conclude that **O(N)** is the dominating term so it the time complexity of Message Preparation

C. Time Complexity of Encryption/Decryption

Algorithm

```
                                  costs/time

# Define the encryption/decryption
functions
function encrypt(digraph): ------->c1/O(1)
```

```
    a, b = digraph
    row_a, col_a =
    divmod(key_square.index(a), 5)
    row_b, col_b =
    divmod(key_square.index(b), 5)
    if row_a == row_b:
        col_a = (col_a + 1) % 5
        col_b = (col_b + 1) % 5
    elif col_a == col_b:
        row_a = (row_a + 1) % 5
        row_b = (row_b + 1) % 5
    else:
        col_a, col_b = col_b, col_a
    return key_square[row_a*5+col_a]
         + key_square[row_b*5+col_b]

function decrypt(digraph):  ----->c2/O(1)
    a, b = digraph
    row_a, col_a =
    divmod(key_square.index(a), 5)
    row_b, col_b =
    divmod(key_square.index(b), 5)
    if row_a == row_b:
        col_a = (col_a - 1) % 5
        col_b = (col_b - 1) % 5
    elif col_a == col_b:
        row_a = (row_a - 1) % 5
        row_b = (row_b - 1) % 5
    else:
        col_a, col_b = col_b, col_a
    return key_square[row_a*5+col_a]
         + key_square[row_b*5+col_b]

# Encrypt or decrypt the plaintext
result = ''
for digraph in digraphs: -------->c3/O(N)
    if mode == 'encrypt':
        result += encrypt(digraph)
    elif mode == 'decrypt':
        result += decrypt(digraph)

# Return the result
return result
```

Here we have two different functions encrypt and decrypt both the function execute in constant time because , at a time they are taking only one Diagraph , and performing Encryption / Decryption rule , this rules are done in constant time because it uses indexing ini key table and mathematical calculation , it does not involve any iteration , so that is why the total time complexity of both functions is constant which is O(1)

Now last we are calling this functions throw for loop which will iterate till the number of Diagraph so it will take O(N) time to execute where N is no of diagraphs

T(E/D) = c1 + c2 + c3N
T(E/D) is the Time complexity of Encryption / Decryption

Here we can say that **O(N)** is dominating term so it will be the time complexity of T(E/D)

Now for The Total time complexity of Playfair Cipher , we have to sum up the time complexity of all the three steps

T(N) = T(K) + T(M) + T(E/D)

T(N) = O(K+A) + O(N) + O(N)

From above We can say O(N) is dominating , Why O(N) is dominating because O(K+A) their is fixed length of alphabets and in Key repetition is not allowed so , if we talk about the worst case the Plaintext length is always much higher than key , key is fixed so that is why O(N) is dominating

So overall time complexity of Play Fair cipher is **O(N)** , Where N is length of Plain text

D. *Factors affecting Tim complexity*

Their are several factors that affects time complexity , they are Length of Key , Length of Message , Number of Diagraphs , Choice of key this are the main factor affecting timecomplexity

V. Future Aspects

Try to reduce the factor affecting time complexity , also try to reduce overall complexity , Also evaluate time complexity for modified version and extended version [9] It can be used as a base for father multi layered encryption approach . Its extended version can be used in Stegnography . Improve user experience because it uses in user facing application . Also make it multi-dimension(3D) approach for enhancing security strength [5].

Integrate with other method make more improved version , for example playfair cipher seed-based color substitution increased security strength by adapting middle square method [14]

A. *The Importance of understanding its time complexity*

The Time complexity of encryption and decryption of algorithm directly affects the speed of cryptographic operation.Some attacks like timing attack or side-channel attack [17]. High complexity may have resource constrained . time complexity also effects its protocol efficiency

VI. Conclusion

This paper aimed to provide an in-depth analysis of the cipher's workings and its time complexity evaluation, shedding light on its strengths, weaknesses, and historical significance.The Significant portion of this paper is talking about the time complexity evaluation.where speed and resource efficiency are essential, a clear comprehension of time complexity is important in making informed decisions regarding its deployment and make it subject of interest for researchers, educators and cryptography enthusiasts.

REFERENCES

[1] Playfair Cipher Implementation in Python - Javatpoint — javatpoint.com. https://www.javatpoint.com/playfair-cipher-implementation-in-python. [Accessed 20-10-2023].

[2] Krishnaraj Bhat, Dindayal Mahto, and Dilip Kumar Yadav. A novel approach to information security using four dimensional (4d) playfair cipher fused with linear feedback shift register. *Indian Journal of Computer Science and Engineering*, 8(1):15–32, 2017.

[3] Subhajit Bhattacharyya, Nisarga Chand, and Subham Chakraborty. A modified encryption technique using playfair cipher 10 by 9 matrix with six iteration steps. *International Journal of Advanced Research in Computer Engineering & Technology*, 3(2):307–312, 2014.

[4] S Gayathri Devi, K Selvam, and SP Rajagopalan. An abstract to calculate big o factors of time and space complexity of machine code. 2011.

[5] Ehsan Elahi, Hasan Raza, and Saqib Ali. A new 3d playfair based secure cipher generation model. In *2017 13th International Conference on Emerging Technologies (ICET)*, pages 1–4. IEEE, 2017.

[6] Safwat Hamad. Novel implementation of an extended 8x8 playfair cipher using interweaving on dna-encoded data. *International Journal of Electrical & Computer Engineering (2088-8708)*, 4(1), 2014.

[7] Piyushi Jain and Surekha Kohle. Variation of playfair cipher. In *2022 5th International Conference on Advances in Science and Technology (ICAST)*, pages 417–421. IEEE, 2022.

[8] Amandeep Kaur, Harsh Kumar Verma, and Ravindra Kumar Singh. 3d (4 x 4 x 4)-playfair cipher. *International Journal of Computer Applications*, 51(2), 2012.

[9] Salman A Khan. Design and analysis of playfair ciphers with different matrix sizes. *International Journal of Computing and Network Technology*, 3(03), 2015.

[10] Lars R Knudsen, Matthew JB Robshaw, Lars R Knudsen, and Matthew JB Robshaw. Brute force attacks. *The Block Cipher Companion*, pages 95–108, 2011.

[11] Ayan Lahiri. *Design and Implementation of an Enhanced Binary Playfair Algorithm using a 4$\times$ 4 Key Matrix*. PhD thesis, 2012.

[12] RI Lanjekar, PS Nevarekar, SP Shinde, and MR Prabhu. Color coded encryption based chatting application. 2018.

[13] Arnold C Licayan, Bobby D Gerardo, and Alexander A Hernandez. Performance analysis of playfair cipher color substitution variants. In *2020 11th IEEE Control and System Graduate Research Colloquium (ICSGRC)*, pages 340–345. IEEE, 2020.

[14] Arnold C Licayan, Bobby D Gerardo, and Alexander A Hernandez. Performance analysis of playfair cipher color substitution variants. In *2020 11th IEEE Control and System Graduate Research Colloquium (ICSGRC)*, pages 340–345. IEEE, 2020.

[15] Preeti Poonia and Praveen Kantha. Comparative study of various substitution and transposition encryption techniques. *Int. J. Comput. Appl*, 145(10):24–27, 2016.

[16] Nikhil Sharma, Himmat Meghwal, Munish Mehta, and Tarun Kumar. A review on playfair substitution cipher and frequency analysis attack on playfair. In *2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI)*, pages 1–9. IEEE, 2018.

[17] François-Xavier Standaert. Introduction to side-channel attacks. *Secure integrated circuits and systems*, pages 27–42, 2010.