ISSN: 0974-5823 Vol. 7 No. 4 April, 2022

International Journal of Mechanical Engineering

ALGORITHM FOR ENCRYPTION AND DECRYPTION IN C++ USING PLANAR GRAPH

M. Anandhi

Assistant Professor, PG and Research Department of Mathematics, Theivanai Ammal College for Women (Autonomous), Villupuram, Tamil Nadu, India.

D. Abarna

PG and Research Department of Mathematics, Theivanai Ammal College for Women (Autonomous), Villupuram, Tamil Nadu, India.

ABSTRACT

In this paper, we develop a new encryption technique that uses graph theory properties to encrypt and decrypt data securely. To construct a complicated cipher text using a shared key, the novel symmetric encryption technique employs the ideas of cycle graph, planar graph and minimal spanning tree.

Keywords: Encryption, Decryption, Planar Graph

1. INTRODUCTION

We consider an undirected graph in this paper G (V, E) where V is the set of vertices and E is set of edges that connect each other. A path is a walk from one vertex to another in which each vertex not appears more than once. A cycle seems when the path(walk) starts from one vertex and return back to the same vertex, when the cycle consist of all vertices in the graph we called it a cycle graph.

An Adjacency list is an array consisting of the address of all the linked lists. Adjacency matrix is a2D array of size V×V where V is the number of vertices in a graph.

Cryptography is the study of secure communications techniques that allow only the sender and intended recipient of a message to view its contents format. Cryptography is an encrypted message in which letters are replaced with other characters. The method of changing text is called a"code" or,more precisely, a "cipher". The changed text is called "cipher text".

2. PRELIMINARIES

Definition: Weighted graph is a graph in which each branch is given a numerical weight.A weighted graph is therefore a special type of labeled graph in which the labels are numbers.

Definition:A cycle graph of order n is a connected graph whose edges form a cycle of length n.

Definition: Let G be a graph with vertex set $\{V_1,V_2,\ldots,V_n\}$ then the adjacency matrix of G is the n×n matrix that has a 1 in the (i,j)-position if there is an edge from V_i to V_j in G and a 0 in the (i,j)-position.

Definition: A planar graph is a graph that can be embedded in the plane,it can be drawn on the plane in such a way that its edges intersect only at their endpoints.

3. PROPOSED ALGORITHM

Encryption Algorithm:

●Add a special character to indicate the starting character (Let A)

●Add vertex for each character in the plain text to the graph.

●Link vertices together by adding an edge between each sequential character in the plain text until us cycle graph

●Weight each edge using the encoding table.Each egde's weight represents the distance between the connected two vertices from the encoding table.

●Adding more edges to form a planar graph M_1,each new added edge has a sequential weight starting from the maximum weight in the encoding table.

●Then find the Minimum Spanning Tree M_2.

●Then store the vertices order in the M_2 matrix in the diagonal places.

●Then we multiply matrices M_1 by M_2 to get M_3.

●After that we multiply M_3 by a predefined Shared-Key K to form C.

●Then the cipher text contains Matrix C and Matrix M_1 line-by-line in a linear format.

Decryption Algorithm:

●The receiver computes M_3 by using the inverse form of the Shared-Key K^{-1}.

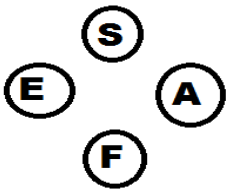
●Then compute M_2 by using the inverse form of M_1.

●Then compute the original text by decodingM_1 using the encoding table.

Example:

Suppose we want to encrypt the plaintext 'SAFE' to send it to the receiver.

The first step is to convert the message to a graph, by converting each character to a vertex.



Convert each character to vertex

Link,each two sequential characters together to form a cycle graph and weight each edge using the below encoding table.

Encoding table

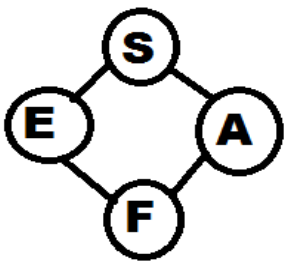
A	**B**	**C**	**D**	**E**	**F**	**…**	**S**	**…**	**W**	**X**	**Y**	**Z**
1	2	3	4	5	6	…	19	…	23	24	25	26

Distance = code(A)-code(S)

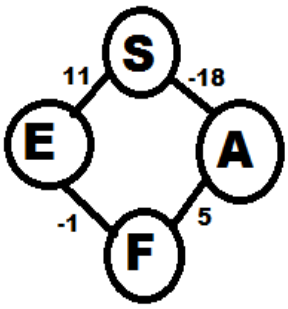
=1-19

= -18

Then the graph will be shown in the above figure:



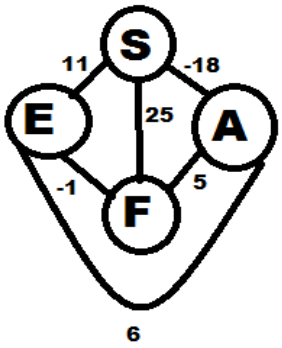
Graph contains plain text character.



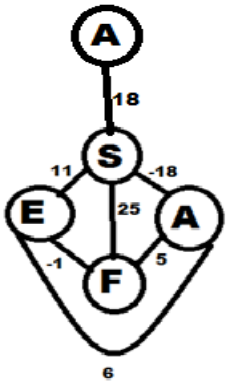
Weighted graph contains plain text characters

The weights of the edges are given by using the encoding table.

We keep adding edges to form a planar graph,each new added edge has a sequential weight starting from the maximum weight in the encoding table(24+1=25).We add a special character before the first character to point to the first character. The planar graph can be represented as a matrix M_1



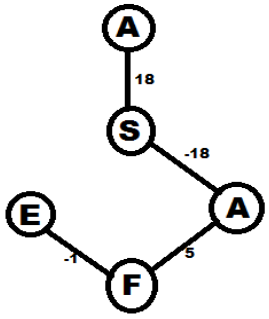
Planar graph



Planar graph with special character

$$M_1 = \begin{bmatrix} 0 & 18 & 0 & 0 & 0 \\ 18 & 0 & -18 & 25 & 14 \\ 0 & -18 & 0 & 5 & 6 \\ 0 & 25 & 5 & 0 & -1 \\ 0 & 14 & 6 & -1 & 0 \end{bmatrix}$$

Then we find the minimum spanning tree



Minimum Spanning Tree

$$M_2 = \begin{bmatrix} 0 & 18 & 0 & 0 & 0 \\ 18 & 0 & -18 & 0 & 0 \\ 0 & -18 & 0 & 5 & 0 \\ 0 & 0 & 5 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 \end{bmatrix}$$

We store the characters order in the diagonal instead of 0's.

Character	**A**	**S**	**A**	**F**	**E**
Order	0	1	2	3	4

$$M_2 = \begin{bmatrix} 0 & 18 & 0 & 0 & 0 \\ 18 & 1 & -18 & 0 & 0 \\ 0 & -18 & 2 & 5 & 0 \\ 0 & 0 & 5 & 3 & -1 \\ 0 & 0 & 0 & -1 & 4 \end{bmatrix}$$

After that, we multiply matrix M_1 by M_2 to formM_3.

$$M_3 = M_1 M_2 = \begin{bmatrix} 324 & 18 & -324 & 0 & 0 \\ 0 & 648 & 89 & -29 & 31 \\ -324 & -18 & 349 & 9 & 19 \\ 450 & -65 & -440 & 26 & -4 \\ 252 & -94 & -245 & 27 & 1 \end{bmatrix}$$

Now, we use the shared-key K to encrypt M_3

Let K= $\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$, so cipher textC= $\begin{bmatrix} 702 & 489 & -571 & 33 & 47 \\ 378 & 471 & -247 & 33 & 47 \\ 378 & -177 & -336 & 62 & 16 \\ 702 & -159 & -685 & 53 & -3 \\ 252 & -94 & -245 & 27 & 1 \end{bmatrix}$

The data to be send is

702 489 -571 33 47 378 471 -247 33 47 378 -177 -366 62 16 702 -159 -685 53 -3 252 -94 -245 27 1

In the receiver side, we get $M_3$by multiplying the cipher text received by the inverse form of the shared key K^{-1}.

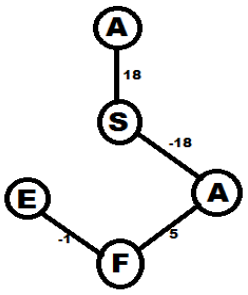
$$M3=K^{-1}C=\begin{bmatrix}702 & 489 & -571 & 33 & 47\\ 378 & 471 & -247 & 33 & 47\\ 378 & -177 & -336 & 62 & 16\\ 702 & -159 & -685 & 53 & -3\\ 252 & -94 & -245 & 27 & 1\end{bmatrix}*\begin{bmatrix}1 & -1 & 0 & 0 & 0\\ 0 & 1 & -1 & 0 & 0\\ 0 & 0 & 1 & -1 & 0\\ 0 & 0 & 0 & 1 & -1\\ 0 & 0 & 0 & 0 & 1\end{bmatrix}$$

$$=\begin{bmatrix}324 & 18 & -324 & 0 & 0\\ 0 & 648 & 89 & -29 & 31\\ -324 & -18 & 349 & 9 & 19\\ 450 & -65 & -440 & 26 & -4\\ 252 & -94 & -245 & 27 & 1\end{bmatrix}$$

Then calculate M_2 by multiplying M_3 by M_1^{-1}

$$M_2 = M_3M_1^{-1} = \begin{bmatrix}0 & 18 & 0 & 0 & 0\\ 18 & 0 & -18 & 0 & 0\\ 0 & -18 & 0 & 5 & 0\\ 0 & 0 & 5 & 0 & -1\\ 0 & 0 & 0 & -1 & 0\end{bmatrix}$$

Here M_2 represents the following final graph



Final graph

Implementation and Experimental Results

To create and test the suggested algorithm, the Microsoft.Net environment was chosen. The method was tested on an Intel® CoreTM i7 CPU with a speed of 2.10 GHz and 6 GB RAM, running on Microsoft Windows 7 Enterprise.64-bit operating system with Service Pack 1.

Experiments on selected plain text representing various length plain texts were conducted to ensure the algorithm's accuracy. Some testing efforts with a random encryption key are shown in the table below (Table 2). The following findings were discovered:

The public key length can be any size because the technique can apply any key length by duplicating it and extending it to the desired length.The amount of the cypher text increases as the plain text grows larger, implying that the algorithm is more efficient when the plain text message is small.Because of the matrix multiplication procedure, the time required to encrypt text increases as well.

4. Implementation of Algorithm

Encryption

```
#include<iostream.h>
//using namespace std:
Intmain()
{
char message[100],ch;
IntI,key;
cout<<"Enter a message to encrypt:";
cin.getline(message,100);
cout<<"Enter key:";
cin>>key;
for(i=0;message[i]!='\0';++1){
ch=message[i];
if(ch>='a'&&ch<='z'){
ch=ch+key
if(ch>'z'){
ch=ch-'z'+'a'-1;
}
message[i]=ch;
}
else if(ch>='A'&&ch<='Z'){
ch=ch+key;
if(ch>'Z'){
ch=ch-'Z'+'A'-1;
}
message[i]=ch;
}
}
cout<<"Encryption message:"<<message;
return 0;
}
```

Output:

Enter a message to encrypt: SAFE

Enter key: 3

Encrypted message: VDIH

Decryption

```
#include<iostream.h>
//using namespace std:
Intmain()
{
char message[100],ch;
intI,key;
```

```
cout<<"Enter a message to decrypt:";
cin.getline(message,100);
cout<<"Enter key:";
cin>>key;
for(i=0;message[i]!='\0';++i)
{
ch=message[i];
if(ch>='a'&&ch<='z'){ch=ch-key;
if(ch>='a'){ch=ch+'z'-'a'+1}message[i]=ch;
else
if (ch>='A'&&ch<='Z'){ ch=ch-key;
if(ch>='a'){ch=ch+'Z'-'A'+1}message[i]=ch;
}
cout<<"Decryption message:"<<message;
return 0;
}
```

Output:

Enter a message to decrypt: VDIH

Enter key: 3

Decrypted message: SAFE

5. CONCLUSION

In this paper,we have discusseda cryptography algorithm using planar graph in C++ programming. In future it can be implemented using any graph in JAVA or Microsoft.Netprogramming language.

6. REFERENCES

1. Corman TH, Leiserson CE, Rivest RL, Stein C. Introduction to algorithms 2nd edition, McGraw-Hill.
2. Yamuna M, Meenal Gogia, Ashish Sikka, Md. Jazib Hayat Khan. Encryption using graph theory and linear algebra. International Journal of Computer Application.ISSN:2250-1797; 2012.
3. Ustimenko VA. On graph-based cryptography and symbolic computations, Serdica. Journal of Computing. 2007;131-156.
4. Paszkiewicz A, et al. Proposals of graph based ciphers, theory and implementations. Research Gate; 2001.
5. Steve Lu, RafailOstrovsky. Daniel Manchala. Visual Cryptography on Graphs, CiteSeerx, COCOON. 2008;225-234.