PLANAR GRAPH IN DATA ENCRYPTION

M. Yamuna[1] and A.Elakkiya[2]

[1,2] *School of Advanced Sciences, VIT University, Vellore, (India)*

ABSTRACT

Transfer of data and its safety is an important issue in this information world. Planar graphs are not in wide use in data encryption. In this paper we propose a method of encryption using planar graphs. We also have developed a program for the proposed algorithm using MATLAB.

Keywords: Decryption, Encryption, Graph.

I INTRODUCTION

Before the modern era, cryptography was concerned solely with message confidentiality (i.e., encryption) - conversion of messages from a comprehensible form into an incomprehensible one and back again at the other end, rendering it unreadable by interceptors or eavesdroppers without secret knowledge (namely the key needed for decryption of that message). Encryption was used to ensure secrecy in communications, such as those of spies, military leaders, and diplomats. Yadhu Ravinath et.al have proposed a selective encryption mechanism using message specific key and spanning tree concept of graph theory. The mechanism provides protection of privacy in communication as it avoids the formation of self-loops and parallel edges and key is exchanged only among the authenticated neighbours only [1]. Esam Suliman Mustafa Ahmed et.al proved the effect of encryption delay on TCP based application is discussed. Increasing the encryption delay and then comparing the effect of that delay on TCP protocol through different scenarios is the methodology of the study, using OPNET [2]. Hussein Th.Khamees et.al used the stream cipher which is the best way with the algorithm Geffe generator with a specific length to Encryption the information from the plain text in the first compute [3]. Graph theory has contributed to the development of various encryption techniques. In this paper we propose a method using graph duals.

II PRELIMINARY NOTE

In this section we provide the basic results of graph theory which are requested for proposed encryption scheme.

2.1 Graph

In the most common sense of the term, a graph is an ordered pair G = (V, E) compromising a set V of vertices or nodes together with a set E of edges or links, which are 2 – elements subset of V (that is an edge is related with two vertices, and the relation is represented as an unordered pair of the vertices with respect to the particular edge)[4].

2.2 Weighted Graph

A 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 (which are usually taken to be positive) [5]. The graph in Fig. 1 is an example of a weighted graph [6].

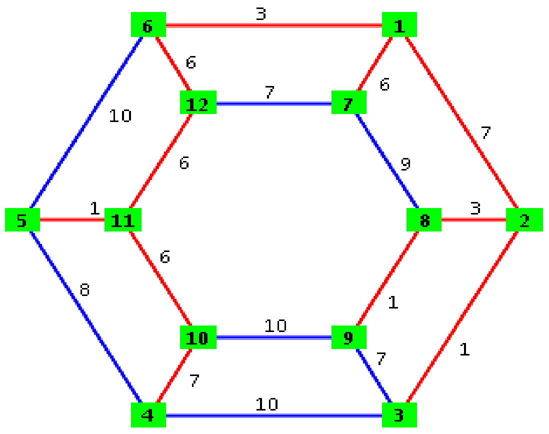


Fig. 1

2.3 Simple Graph

A simple graph, also called a strict graph, is an unweighted, undirected graph containing no graph loops or multiple edges [7]. In Fig. 2 G1 [8] represents a simple graph and G2 a multiple graph [9].

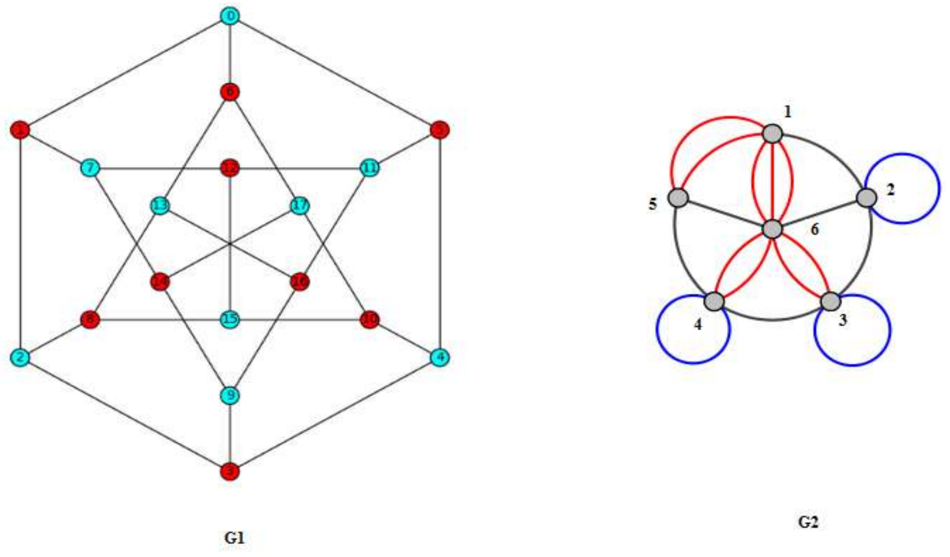


Fig. 2

2.4 Planar Graph

In graph theory, a planar graph is a graph that can be embedded in the plane, i.e., it can be drawn on the plane in such a way that its edges intersect only at their endpoints. In other words, it can be drawn in such a way that no edges cross each other. Such a drawing is called a plane graph or planar embedding of the graph [10]. The graph in Fig. 1 is an example of a weighted graph [11]

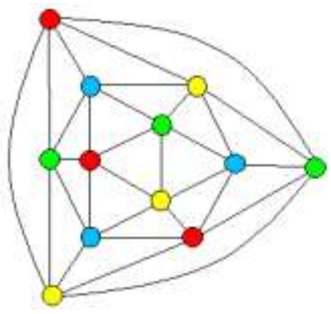


Fig. 3

2.5 Dual Graph

The dual graph of a plane graph G is a graph that has a vertex corresponding to each face of G, and an edge joining two neighboring faces for each edge in G. In Fig. the red graph is the dual graph of the blue graph [12].

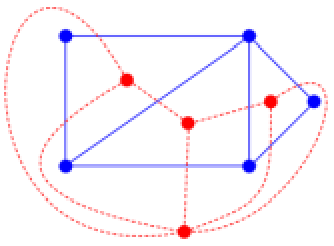


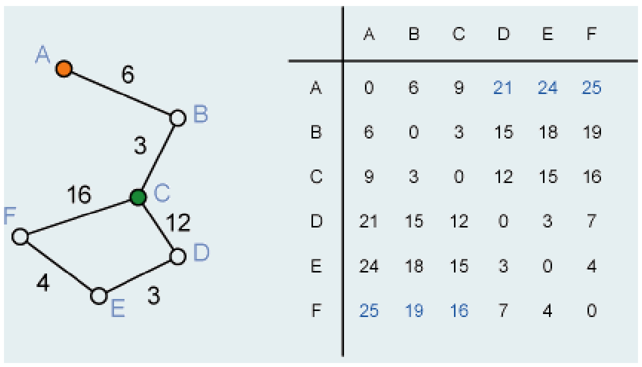
Fig. 4

2.6 Degree

In graph theory, the degree of a vertex of a graph is the number of edges incident to the vertex, with loops counted twice. The degree of a vertex v is denoted deg (v). The maximum degree of a graph *G*, denoted by Δ(*G*), and the minimum degree of a graph, denoted by δ(*G*), are the maximum and minimum degree of its vertices In Fig.2 the degree of all the vertices in G1 has degree 3 and the degree of vertices 1, 2, 3, 4, 5 in G2 are 6, 5, 6, 6, 4, 9 respectively [13].

Distance Matrix

In mathematics, computer science and graph theory, a distance matrix is a matrix (two-dimensional array) containing the distances, taken pairwise, of a set of points. This matrix will have a size of n x n where n is the number of points, nodes or vertices (often in a graph)[14]. Snapshot – 1 provides a weighted graph and its related distance matrix [15].



	A	B	C	D	E	F
A	0	6	9	21	24	25
B	6	0	3	15	18	19
C	9	3	0	12	15	16
D	21	15	12	0	3	7
E	24	18	15	3	0	4
F	25	19	16	7	4	0

Snapshot – 1

III PROPOSED METHOD

In this paper we aim to encrypt any message S using a graph G and its dual G^*. If there is a vertex of degree two in a planar graph G, then there will more than one edge between a pair of vertices in G*. So while we decrypt, more than one edge between a vertex pair creates a dilemma in picking the right edge. If G is a simple planar graph with degree of each vertex atleast 3, then G* is always simple. So, we use only graphs where G, G^* are simple and degree of every vertex in G is atleast 3. The basic idea is that we pick a random edge sequence in graph G. We then pick the corresponding edge sequence in the dual G* and assign weights to the edges and use the graph for encryption.

3. 1 Encryption Chart

We can use any normal encryption chart. Table – 1 provides a sample chart.

A	B	C		Z	Blank space
↕	↕	↕	• • •	↕	↕
1	2	3		26	27

Table – 1

We can include any number of characters depending on the need of the message to be encrypted.

3. 2 Encryption Algorithm

Let | S | = K. Let us label the edges in G as e_1, e_2, … and the corresponding edges in G^* as e_1^*, e_2^*, …. Let **S = SECRET** be the message to be encrypted.

Step 1 Choose a planar graph G with atleast K edges and construct G^*.

Since | S | = 6 we choose a graph with atleast 6 edges as seen in Fig. 5. The dual graph G* is also seen in Fig. 5.

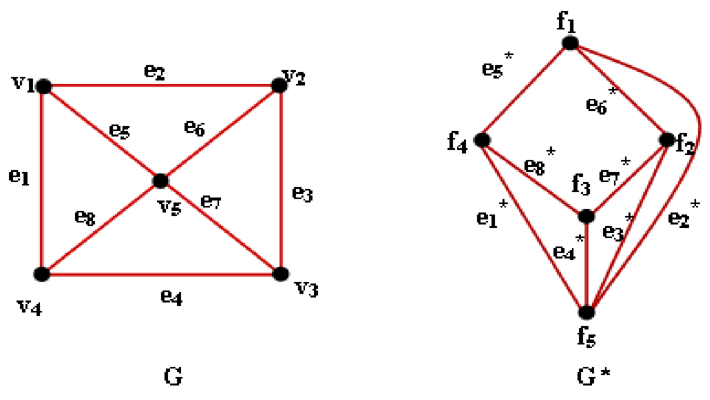


Fig. 5

Step 2 Randomly pick K edges from G say e_1, e_2, … e_k .

We randomly choose the edges e_7, e_6, e_4, e_5, e_8, e_3 from Fig. 5.

Step 3 Convert each character in S into numerical values using Table –1 to generate a sequence w_1, w_2, … w_k .

In our example the sequence w_1, w_2, …, w_6 is 19 5 3 18 5 20 (from Table – 1).

Step 4 Assign the values w_1, w_2, … w_k as weights of the e_1^*, e_2^*, …e_k^*.

Assigning the weights 19 5 3 18 5 20 to the edges e_7*, e_6*, e_4*, e_5*, e_8*, e_3*, the graph is as seen in Fig.6

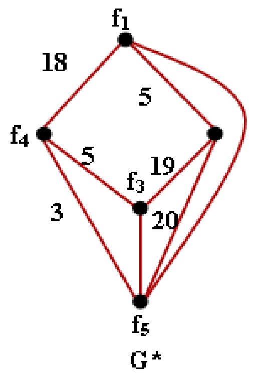


Fig. 6

Step 5 Assign arbitrary weights to the remaining edges in G^* and to edges in G.

We assign weights 2, 9 to the remaining edges e_1, e_2 in G and weights 24, 27 to the edges e_1*, e_2* in G*.

The resulting graph is as seen in Fig. 7.

Step 6 Send G, G^* to the receiver.

Finally we send the graph in Fig. 7 to the receiver (the lines with message is highlighted in blue).

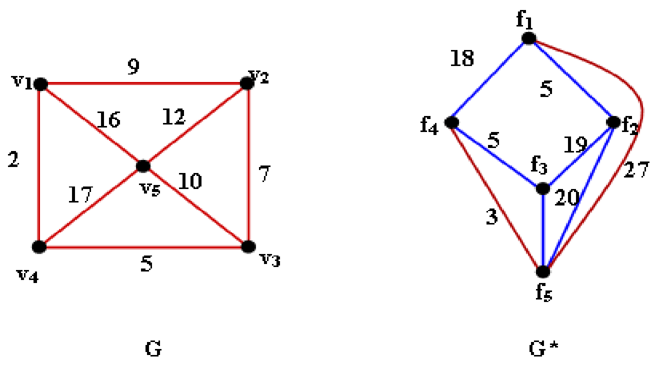


Fig. 7

For decrypting the message we reverse the procedure.

Suppose the received message is as seen in Fig. 8.

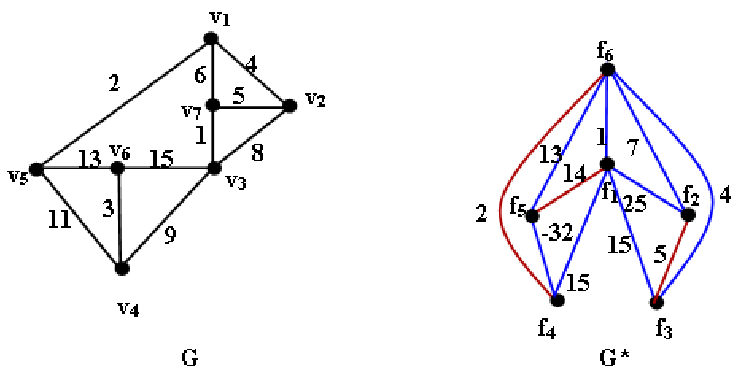
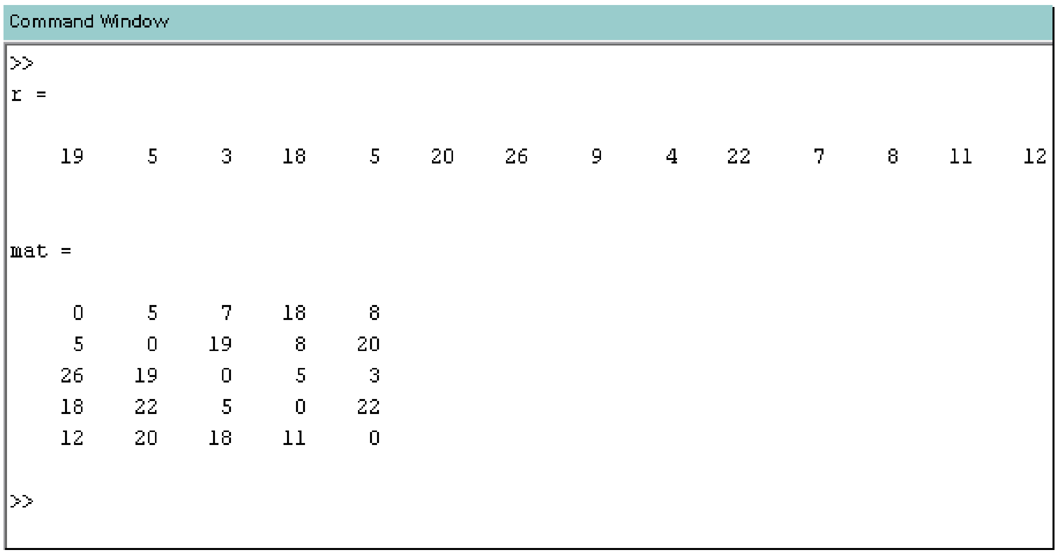


Fig. 8

Suppose the edge sequence given to us is (v_1, v_2), (v_7, v_3), (v_6, v_3), (v_5, v_6), (v_6, v_4) (v_3, v_2), (v_1, v_5), (v_1, v_7) (Note than instead of edges labeling the edges are provided using vertex labels). We pick the corresponding edge sequence from G*. The edge sequence is (f_2, f_6), (f_1, f_3), (f_1, f_4), (f_1, f_5), (f_4, f_5), (f_3, f_6), (f_1, f_6), (f_1, f_2). The corresponding weight sequence is 7 15 15 4 27 4 1 25. The message is decrypted as **GOOD DAY** from Table – 1.

IV IMPLEMENTATION OF THE ALGORITHM

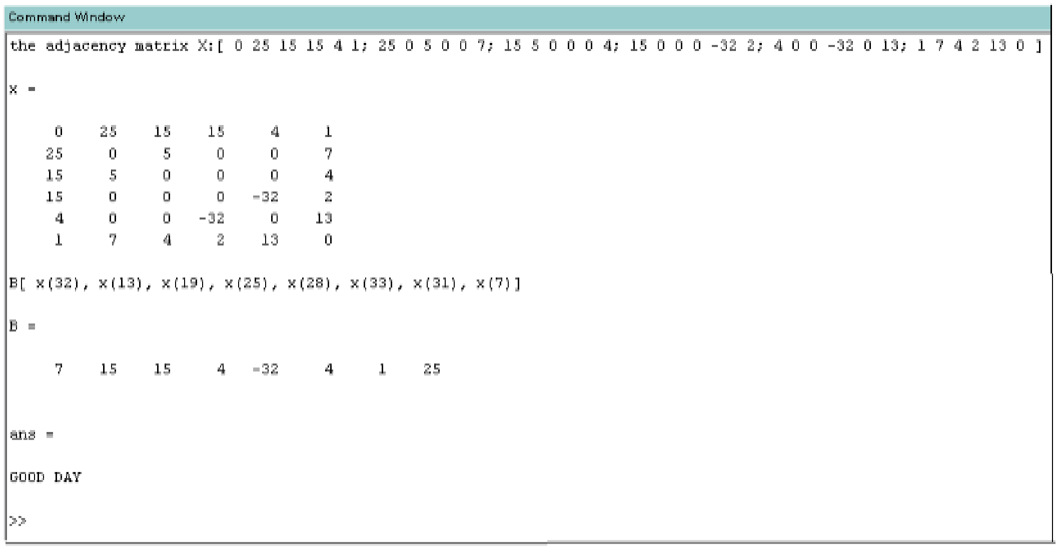
We have created a MATLAB code for encryption and decryption of the message. A graph can be represented by its adjacency matrix. In the MATLAB code we finally encrypt the graph as an a distance matrix to the receiver. Snapshot – 2 provides the output screen for the message SECRET encrypted in the example. We have used ASCII conversions for the message to be converted to a string and then into a distance matrix.



```
Command Window
>>
r =

    19     5     3    18     5    20    26     9     4    22     7     8    11    12


mat =

     0     5     7    18     8
     5     0    19     8    20
    26    19     0     5     3
    18    22     5     0    22
    12    20    18    11     0

>>
```

Snapshot – 2

It can be noted that the first six characters in the output string matches with the conversion using Table – 1 as in Sec 3. 2. The distance matrix can be send to the receiver instead of the graphs.



```
Command Window
the adjacency matrix X:[ 0 25 15 15 4 1; 25 0 5 0 0 7; 15 5 0 0 0 4; 15 0 0 0 -32 2; 4 0 0 -32 0 13; 1 7 4 2 13 0 ]

x =

     0    25    15    15     4     1
    25     0     5     0     0     7
    15     5     0     0     0     4
    15     0     0     0   -32     2
     4     0     0   -32     0    13
     1     7     4     2    13     0

B[ x(32), x(13), x(19), x(25), x(28), x(33), x(31), x(7)]

B =

     7    15    15     4   -32     4     1    25


ans =

GOOD DAY

>>
```

Snapshot – 3

Snapshot – 3 provides the output screen for the decryption example discussed in Sec 3. 2. It can be verified that the output matches with the example discussed. We have used the regular ASCII conversions for converting numbers to strings and strings to numbers. The program provides fast output for graphs with 10 vertices. It is expected that output would be generated fast with advanced models.

V CONCLUSION

- The edge sequence can be picked in any random order. If G is a graph with e – edges and the message to be encrypted is of length k, then the number of possible combinations for choosing edges $= eC_k$.
- For each of these combinations we can choose the edge sequence from G*. So the number of ways of encrypting the message is eC_k.
- Unless the edge sequence is known the message cannot be decrypted, which means that one has to try eC_k combinations to decrypt a message.

Also numerous graphs are available in public domain, that it is difficult to find the encrypted graph and a fake one. So the proposed method is safe for encryption of any message.

REFERENCES

[1] Yadhu Ravinath, Vipul Mangla, Arkajit Battacharya, Peeyush Ohri, Graph Theory Application in Selective Encryption Mechanism for Wireless Ad Hoc Network, International Journal of Emerging Trends and Technoloy in Computer Science, Vol 2, Issue 2, March - April 2013

[2] Essan suliman Mustafa Ahmed Dr.Amin Babiker A/Nabi Mustafa, The Effect of Encryption Algorithms Delay on TCP Traffic Over Data Networks, IOSR Journal of Computer Engineering, Volume 17, Isuue 1, Ver . II (Jan – Feb 2015), pp 85-91.

[3] Hussien Th Khamees, Jalal A. Kahlf and A.Al-sajee, Encryption and Decryption of Data by using Geffe Algorithm, International Journal of Modern Engineering Research, Vol 2, Issue 3, May –June 2012 pp-1354 – 1359.

[4] http://en.wikipedia.org/wiki/Graph_theory

[5] http://mathworld.wolfram.com/WeightedGraph.html

[6] http://www.maplesoft.com/support/help/Maple/view.aspx?path=examples/GraphTheory

[7] http://mathworld.wolfram.com/SimpleGraph.html

[8] http://www.sagemath.org/doc/reference/graphs/sage/graphs/graph.html

[9] http://en.wikipedia.org/wiki/Glossary_of_graph_theory

[10] http://en.wikipedia.org/wiki/Planar_graph

[11] www.mathpages.com/home/kamth266/kamth266.htm

[12] http://en.wikipedia.org/wiki/Dual_graph

[13] http://en.wikipedia.org/wiki/Degree_%28graph_theory%29

[14] http://en.wikipedia.org/wiki/Distance_matrix

[15] http://www.gitta.info/Accessibiliti/en/html/StructPropNetw_learningObject4.html