Indian Journal of Advanced Mathematics (IJAM)
ISSN: 2582-8932 (Online), Volume-4 Issue-1, April 2024

Some Algorithms of Graph Theory in Cryptology

Auparajita Krishnaa

Abstract: The inventive use of concepts from Graph Theory plays a significant role in hiding the original plaintext, resulting in a significantly safer data transfer. In this work, tree traversal algorithms such as Inorder, Preorder, and Postorder, as well as Kruskal's algorithm for creating a minimal spanning tree, and the modified graph labelling scheme of graceful labelling, which allows for the repetition of exactly one vertex label for certain graphs, have been employed to create highly hidden Cypher Texts. Encryption and decryption algorithms for all these methods are being presented in this work.

Keywords: Modified Graceful Labelling, Tree Traversal Algorithms, Expression Tree, Modified Ordered Rooted Tree, Basic-Cypher-text, Cypher-Matrix (of different kinds), Encryption-Array.

I. INTRODUCTION

The *tree traversal* algorithms, including Inorder, Preorder, and Postorder, along with their respective notations and the corresponding trees, can be found in Kolman, Busby, **and Ross** [1]. **Katz** [2] is a survey that provides the basic definitions of binary tree encryption (BTE), as well as recent applications to forward–secure encryption, identity-based and hierarchical identity-based encryption, chosen-ciphertext security, and adaptively-secure encryption. **Krishnaa and Dulawat** [3] have proposed a modified graceful labelling method that allows repetition of exactly one repeated vertex label. **Shanmugam** [4] presents a security technique that combines the Caesar Cypher and graph traversal. **Sivakumar, Humshavarthini, Jayasree, and Eswaran** [5] present encryption and decryption using ASCII values of characters to achieve first-level encryption. The Binary Tree Traversal (BTT) is then used at the second level of encryption to achieve permutation. **Krishnaa** [6] has applied inner magic and inner antimagic labelling using both encryption and decryption algorithms based on the Wheel graph. Additionally, several specific graphs with their corresponding encryption and decryption algorithms are presented in **Krishnaa** [7]. **Krishnaa** [8] presents a cryptographic application employing lattices, utilising certain concepts from the theory of Lattices in Discrete Mathematics. The idea of Basic Cypher Text has been introduced in **Krishnaa and Gurjar.** [9] to further increase the hiding capacity of the final Cypher texts to be developed subsequently, or the option of using the Basic-Cypher-Text as the final Cypher Text itself.

Manuscript received on 27 February 2024 | Revised Manuscript received on 06 March 2024 | Manuscript Accepted on 15 April 2024 | Manuscript published on 30 April 2024.
** Correspondence Author (s)*
Auparajita Krishnaa*, Department of Mathematics and Statistics, Mohan Lal Sukhadia University, Udaipur (Rajasthan) India. Email: akrishnaa1@gmail.com, ORCID ID: 0000-0002-1222-1711

In subsection A, the tree traversal algorithms —Inorder, Preorder, and Postorder —both with and without expression trees — have been used to create the Ciphertexts. In subsection B, the *modified graceful labelling, which allows the repetition of exactly one vertex label for binary trees and double-star graphs, is presented. In subsection C, Kruskal's algorithm for creating a minimal spanning tree using the concept of a modified ordered rooted tree is presented for generating highly encrypted Cypher Texts*. For all these concepts, encryption and decryption algorithms have been presented.

II. MAIN RESULTS

A. Tree Traversal Algorithms

A **Basic Cypher Text** using the tree traversal algorithms has been created to provide added security in hiding the Plaintext. The process of visiting every vertex of a tree in a particular order is called *tree traversal*. It is of 3 kinds and mainly performed on *binary trees*. In a ***binary tree***, each vertex has at most two offspring (left and right). The three tree traversal algorithms are as follows:

Preorder:
1. Visit the root
2. Search the left subtree if it exists
3. Search the right subtree if it exists.

A Preorder traversal yields a *Prefix* notation.

Inorder:
1. Search the left subtree if it exists
2. Visit the root
3. Search the right subtree if it exists

An Inorder traversal yields an *Infix* notation.

Postorder:
1. Search the left subtree if it exists
2. Search the right subtree if it exists
3. Visit the root.

A Postorder traversal yields a *Postfix* notation.

a. Tree Traversal Algorithms Without Expression Trees

An *expression* tree represents a mathematical expression. In this section, a case of creating cypher texts is demonstratedwithout using an *expression tree*.

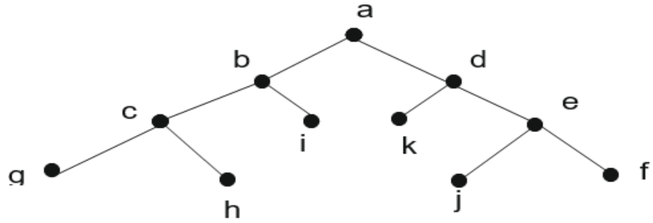

Figure 1: Tree used without a Mathematical Expression for Preorder Traversal

Encryption:

1. The Plain-text: GRAPHTHEORY is assigned to the vertices of the graph in **Figure 1** in Prefix notation (a b c g h i d k e j f) where v1, v2, v3 … are the names of the vertices, where by the **Preorder** Traversal, the **Basic-Cypher-text** obtained is: G R P E O H A T H R Y, and the assignments are as shown below:

G = a= v1
R = b = v2
P = c = v4
E = g = v8
O = h = v9
H = i = v5
A = d = v3
T = k = v6
H = e = v7
R = j = v10
Y = f = v11

2. The Cypher-text **H T T M X M D Z O B J** is calculated (from top to bottom) by adding the number "i" in "vi" to the letters of the **Basic-Cypher-text** shown below in the Cypher-Matrix (only the Cypher-text needs to be sent to the receiver; calculations of the The ciphertext is only for demonstration purpose)

Cipher-Matrix

G + 1 = H	a	v1
R + 2 = T	b	v2
P + 4 = T	c	v4
E + 8 = M	g	v8
O + 9 = X	h	v9
H + 5 = M	i	v5
A + 3 = D	d	v3
T + 6 = Z	k	v6
H + 7 = O	e	v7
R + 10 = B	j	v10
Y + 11 = J	f	v11

3. **Adjacency Matrix**: It is a matrix which can represent the graph in matrix form such that its matrix entry A_{ab} = 1 if there is an edge between vertices a and b, else A_{ab} = 0.

The Adjacency Matrix for the graph in **Figure 1** is given below:

Vertex	a	b	c	d	e	f	g	h	i	j	k
a	0	1	0	1	0	0	0	0	0	0	0
b	1	0	1	0	0	0	0	0	1	0	0
c	0	1	0	0	0	0	1	1	0	0	0
d	1	0	0	0	1	0	0	0	0	0	1
e	0	0	0	1	0	1	0	0	0	1	0
f	0	0	0	0	1	0	0	0	0	0	0
g	0	0	1	0	0	0	0	0	0	0	0
h	0	0	1	0	0	0	0	0	0	0	0
i	0	1	0	0	0	0	0	0	0	0	0
j	0	0	0	0	1	0	0	0	0	0	0
k	0	0	0	1	0	0	0	0	0	0	0

4. Send the Adjacency Matrix, Cypher-text H T T M X M D Z O B J, Cypher-Matrix to the receiver. The vertices in the Adjacency Matrix can be labelled as v1, v2, v3, … or a,b,c,….

Decryption:

1. Arrange the's in the Cypher-Matrix in ascending order.
2. The original plaintext is calculated by the reverse operation, i.e., subtraction, as follows:

H – 1 = G, T – 2 = R, D – 3 = A, T – 4 = P, M – 5 = H, Z – 6 = T, O – 7 = H, M – 8 = E, X – 9 = O,
B – 10 = R, J – 11 = Y, therefore yielding the original plaintext: GRAPHTHEORY.

Retrieval Number:100.1/ijam.A116704010424
DOI: 10.54105/ijam.A1167.04010424
Journal Website: www.ijam.latticescipub.com

Published By:
Lattice Science Publication (LSP)

b. Using Expression Trees in Inorder and Postorder traversals to make Ciphertexts:

Encryption:

1. Make the following assignments of the Plain-text: COUNTRIES to the vertices of the *expression tree* of **Figure 2** as follows:

v1 = C = x
v2 = O = -
v3 = U = a
v4 = N = b
v5 = T = +
v6 = R = c
v7 = I = /
v8 = E = d
v9 = S = e

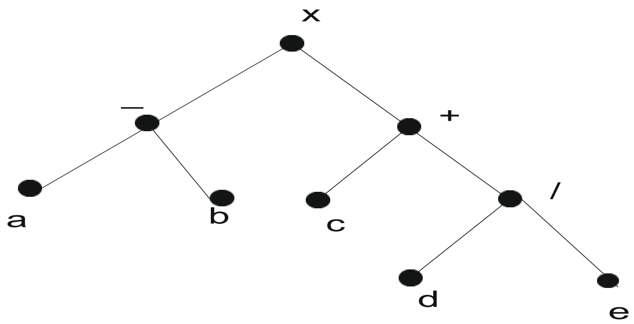

Figure 2: Expression Tree for Postorder and Inorder Traversal

2. Two Cipher-texts are obtained as shown below:

(a) **Postorder** Traversal yielding **Postfix notation**: a b – c d e / + x and obtain the Cypher-text by adding the "i" number in vertex "vi" to the letters of the Basic-Cypher-text UNORESITC obtained by the Postorder traversal.

(b) **Inorder** Traversal yielding **Infix notation**: a–b x c +d / e and obtain the
Cypher-text by adding the "i" number in vertex "vi" to the letters of the Basic-Cypher-text UONCRTEIS obtained by the Inorder traversal.

3. The two Cypher-texts are shown in the following matrices in their rightmost columns from top to bottom, namely: **XRQXMBPYD** and **XQRDXYMPB** (only the Cypher-text needs to be sent to the receiver; calculations of the Cypher-text are only for demonstration purposes)

Postfix-Cipher-Matrix

U + 3 = X	a	v3
N + 4 = R	b	v4
O + 2 = Q	-	v2
R + 6 = X	c	v6
E + 8 = M	d	v8
S + 9 = B	e	v9
I + 7 = P	/	v7
T + 5 = Y	+	v5
C + 1 = D	x	v1

Infix-Cipher-Matrix

U + 3 = X	a	v3
O + 2 = Q	-	v2
N + 4 = R	b	v4
C + 1 = D	x	v1
R + 6 = X	c	v6
T + 5 = Y	+	v5
E + 8 = M	d	v8
I + 7 = P/	v7	
S + 9 = B	e	v9

4. Send the Cypher-texts XRQXMBPYD (by Postfix notation) and XQRDXYMPB (by Infix notation), the respective Postfix-Cypher-Matrix and Infix-Cypher-Matrix and the Adjacency Matrix of the tree to the receiver. It is to be noted that the first column in each Cypher-Matrix is shown for calculation purposes only; the Cypher-text is to be sentto the receiver.

Decryption:

1. Perform a Preorder Traversal on the original tree obtained by the Adjacency Matrix, yielding the Prefix notation: x-a b + c / d e. It is to be noted that the Preorder traversal of the expression, i.e., x – a b + c / d e, is the ascending order of the vi's in both Postfix-Cypher-Matrix and Infix-Cypher-Matrix.
2. Arrange the VIs in ascending order
3. Subtract the number "i" of "vi" from the respective letter of the Cypher-text to get the original Plain-text.

(i)Postfix : D – 1 = C, Q – 2 = O, X – 3 = U, R – 4 = N, Y – 5 = T, X – 6 = R, P – 7 = I, M – 8 = E,
B–9 = S. Therefore, the original plaintext: COUNTRIES is obtained.

(ii)Infix : D – 1 =C, Q – 2 = O, X – 3 = U, R – 4 = N, Y – 5 = T, X – 6 = R, P – 7 = I, M – 8 = E, B – 9 = S. Therefore, the original plaintext: COUNTRIESis obtained.

Another option is to have the original Plaintext message in Postfix notation and the Ciphertexts in Prefix and Infix notation, as the suitability of Prefix and Postfix notations is more suitable for computers, both being free of parentheses. In contrast, Infix notation is ambiguous without parentheses and is more ideal for human use.

Retrieval Number:100.1/ijam.A116704010424
DOI: 10.54105/ijam.A1167.04010424
Journal Website: www.ijam.latticescipub.com

Published By:
Lattice Science Publication (LSP)

In case the expression is not so important, the Infix notation can also be used to stand for the original Plain-text message and have the Cypher-texts in Prefix and Postfix notations and proceed similarly.

B. Binary Tree and Doublestar with Modified Graceful Labelling Allowing Repetition of Exactly One Vertex Label:

Graph Labelling: It is an assignment of labels (numbers) to vertices or edges such that the induced edge or vertex labels follow a specific pattern.

Graceful Labelling: For a graph G(p,q) with p number of vertices and q number of edges, a *graceful labelling* is given by assigning the vertices with 0,1,2,…, q such that the *induced edge labels* are provided by $f^*(x, y) = |f(x) - f(y)|$ where f(x) and f(y) are the vertex labels of the vertices x and y and f* is injective.

Modified Graceful Labelling: Given in [3], this allows repetition of exactly one vertex label and the induced edge label of 0 is allowed.

Binary Tree: The plaintext for creating the ciphertext for thisgraph is: COUNTRY. The vertex labels are those of the *modified graceful labelling,* allowing repetition of exactly one vertex label.

Encryption (Binary Tree):

1. The vertices in the Binary Tree of **Figure 3** are given the following assignments exhibiting the *modified graceful labelling* as shown in [3] (right-most column provides the vertex with labels of the *modified graceful labelling*):

v1 = C = 0
v2 = O = 5
v3 = U = 0
v4 = N = 1
v5 = T = 4
v6 = R = 3
v7 = Y = 2

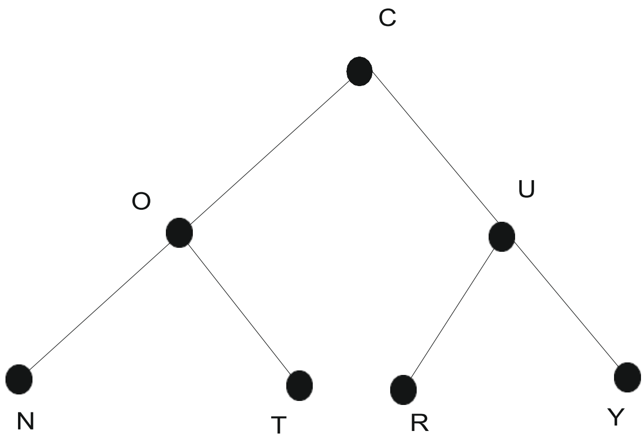

Figure 3: Binary Tree.

2. The ciphertext **CTUPXUA** is calculated by adding the respective vertex labels, namely 0, 5, 0, 1, 4, 3, 2 of the vertices v1, v2, v3, …, v7, to the letters of the Plain-text COUNTRY as follows:
C + 0 = C, O + 5 = T, U + 0 = U, N + 1 = P, T + 4 = X, R + 3 = U, Y + 2 = A thus, yielding the Cipher-text **CTUPXUA**
3. Make the Cypher-Matrix-Binary Tree as follows:

Cipher-Matrix-Binary Tree

2	v7	A
4	v5	X
3	v6	U
5	v2	T
1	v4	P
0	v1	C
0	v3	U

4. Send the Cypher-text, Cypher-Matrix-Binary-Tree to the receiver.

Decryption (Binary Tree):

1. In the order of v1, v2, v3,…v7 or the vertex labels of the *modified graceful labelling* 0 5 0 1 4 3 2 of the Binary Tree, associate letters of the Cypher-text with the vertex labels as follows: C = 0, T = 5, U = 0, P = 1, X = 4, U = 3, A = 2
2. Arrange the vertex labels 0 5 0 1 4 3 2 in the ascending order v1, v2, v3,…v7 (these vertices and the respective vertex labels are of the *modified graceful labelling* of the Binary Tree)
3. Subtract the vertex labels from the letters of the ciphertext to obtain the plaintext as shown below:
C – 0 = C, T – 5 = O, U – 0 = U, P – 1 = N, X – 4 = T, U – 3 R, A – 2 = Y, therefore yielding the original plaintext, which is the country.

Doublestar: This graph has been mentioned and explored in [3] for studying the effect of repeating a vertex label in the various graph labelling schemes. It is a kind of *tree*. In this case, an additional binary tree has been used to generate the initial Basic-Cypher-text, providing an added layer of secrecy in hiding the original Plain-text message through Inorder traversal. Doublestar also exhibits a *modified graceful labelling* as per [3].

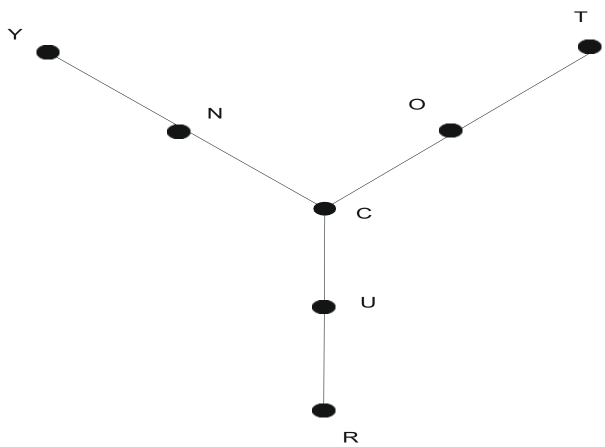

Figure 4: Doublestar

Plain-text is COUNTRY

Retrieval Number:100.1/ijam.A116704010424
DOI: 10.54105/ijam.A1167.04010424
Journal Website: www.ijam.latticescipub.com

Published By:
Lattice Science Publication (LSP)

Encryption (Doublestar):

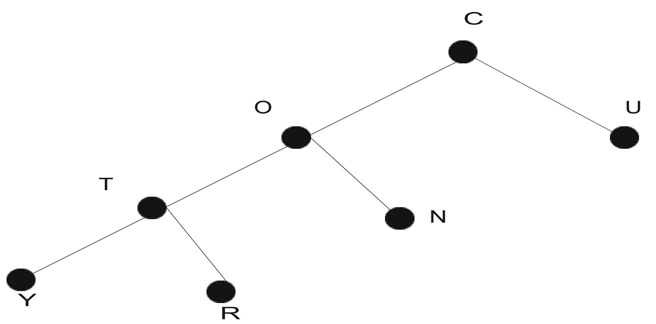

Figure 5: Inorder traversal to get Basic-Cypher-text: Y T R O N C U for the Doublestar

1. The vertices in the Doublestar of **Figure 4** are given the following assignments (right-most column gives the vertex labels of the *modified graceful labelling*):
   v1 = C = 0
   v2 = O = 2
   v3 = N = 4
   v4 = U = 0
   v5 = T = 3
   v6 = Y = 1
   v7 = R = 5

2. Use the adjacency matrices of Doublestar and binary tree to generate the **Basic-Cypher-text** by Inorder traversal: YTRONCU

3. Make the ciphertext, which is calculated by adding the vertex labels of the modified graceful labelling to the letters of the plaintext **in reverse order** (using the **Basic-Cypher-text)** as follows:

Y + 2 = A, T + 3 = W, R + 4 = V, O + 1 = P, N + 5 = T, C + 0 = C, U + 0 = U, thus yielding the Cypher-text: **A W V P T C U** (only the Cypher-text needs to be sent to the receiver; Calculations of the Cypher-text are only for demonstration purposes)

4. Make the Cypher-Matrix-Doublestar as follows:

Cipher-Matrix-Doublestar

Y	2	x1	A = Y + 2
T	3	x3	W = T + 3
R	4	x2	V = R + 4
O	1	x6	P = O + 1
N	5	x4	T = N + 5
C	0	x7	C = C + 0
U	0	x5	U = U + 0

5. Send the Cypher-text, Cypher-Matrix-Doublestar tothe receiver.

Decryption (Doublestar):

1. Arrange xi's in ascending order of subscripts: x1, x2, x3, x4, x5, x6, x7 to get AVWTUPC
2. Arrange the above in reverse order to get CPUTWVA
3. Perform subtraction to obtain the original plaintext as follows:

C – 0 = C, P – 1 = O, U – 0 = U, T – 5 = N, W – 3= T, V – 4 = R, A - 2 = Y, therefore yielding the original Plain-text: COUNTRY

C. Kruskal's Algorithm

The definitions of the concepts used are as follows:

Weighted Graph: A *weight* is a non-negative integer assigned to an edge of a graph, which could mean an entity, for instance, the distance between the two end-vertices of that edge. Such a graph with weights assigned to its edges is called a *weighted graph.*

Spanning Tree: A *spanning tree* of a graph G is its subgraph which contains *all* the vertices of G and is a *tree* (connected and acyclic) in itself.

Minimal Spanning Tree: The minimally weighted spanning tree of a weighted graph is called a minimal spanning tree. Kruskal's algorithm gives a method to find *minimal spanning tree* of a graph.

Rooted Tree: A *rooted tree* has a fixed vertex called *the root* from which all other edges go downwards.

Modified Ordered Rooted Tree: An *ordered rooted tree* numbers edges in a *rooted tree* from left to right in ascending order at each *level* of the *rooted tree*. In this work, the pre-assigned edge labels are read from left to right at each level of the newly defined, modified rooted tree, starting from the top level and proceeding to the bottom level. The labels are not necessarily in ascending order.

The letters of the **Plain-text: CRYPTOLOGICS** are assigned to the edges labelled with 1, 2, 3,4, 5, 6, 7, 8, 9, 10, 11, 12 with C, R, Y, P, T, O, L, O, G, I, C, S respectively in **Figure 6:**

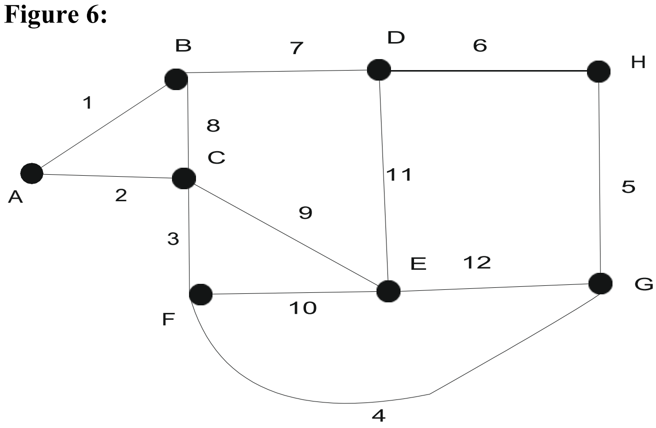

Figure 6: Original Graph

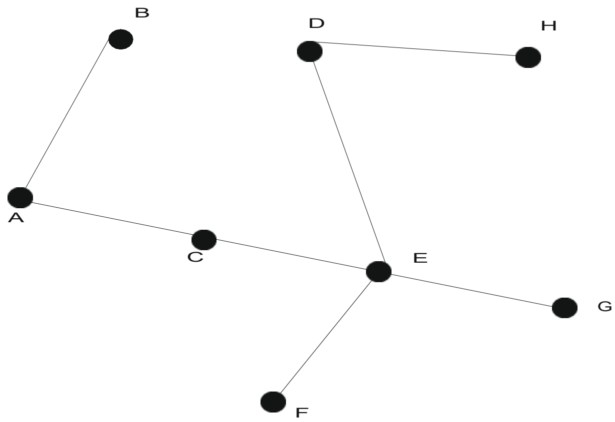

Figure 7: Minimal Spanning Tree of the Original Graph by Kruskal's Algorithm

Retrieval Number:100.1/ijam.A116704010424
DOI: 10.54105/ijam.A1167.04010424
Journal Website: www.ijam.latticescipub.com

Published By:
Lattice Science Publication (LSP)

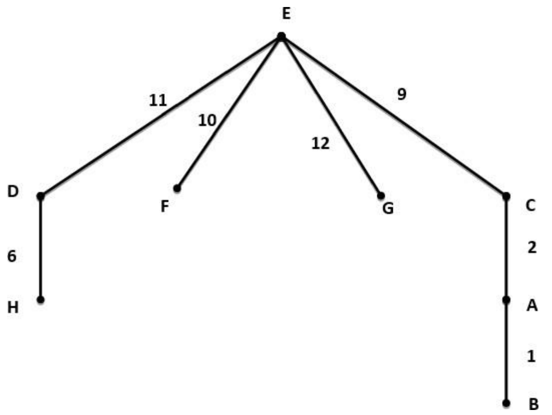

Figure 8: Modified Ordered Rooted Tree of the Minimal Spanning Tree

The newly developed Cypher-Matrix of the remaining graph (the original graph without the Minimal Spanning Tree) is shown below, where the matrix entry comprises the pair (edge label, letter of Plaintext*)*.

A, B, C, D, E, F, G, H are the vertex labels of the Original Graph. For example, in this matrix, the entry (8, O) corresponds to the edge B-C, with end-vertices labelled as B and C, as per the Original Graph.

	A	B	C	D	E	F	G	H
A	-	-	-	-	-	-	-	-
B	-	-	(8,O)	(7,L)	-	-	-	-
C	-	(8,O)	-	-	-	(3,Y)	-	-
D	-	(7,L)	-	-	-	-	-	-
E	-	-	-	-	-	-	-	-
F	-	-	(3,Y)	-	-	-	(4,P)	-
G	-	-	-	-	-	(4,P)	-	(5,T)
H	-	-	-	-	-	-	(5,T)	-

Cypher-Matrix of the *remaining graph* (original graph without the minimum spanning Tree)

Encryption:

1. Associate the numbers in the order of appearance of plain text as follows to make the cypher-text:

C + 1 = D
R + 2 = T
Y + 3 = B
P + 4 = T
T + 5 = Y
O + 6 = U
L + 7 = S
O + 8 = W
G+ 9 = P
I + 10 = S
C + 11 =N
S + 12 = E

Therefore, the ciphertext is: **D T B T Y U S W P S N E**

2. Make the Minimal Spanning Tree

3. Make a Modified Ordered Rooted Tree of the Minimal Spanning Tree (weights/numbers/ edge labels are read from left to right at each level, from top level to bottomlevel)

4. Encryption- Array: (N,11), (S,10), (E,12), (P,9), (U,6), (T,2), (D,1) in the order of left to right in the Modified Ordered Rooted Tree. For example, the pair (N, 11) has been constructed from C + 11 = N.

5. Send the Cypher-text, Encryption-Array and Cypher-Matrix of the *remaining graph* (original graph without the Minimal Spanning Tree) to the receiver.

Decryption:

1. From the *remaining graph* (B,3), (T,4), (Y,5), (S,7), (W,8) [pairs constructed as in step 4 of **Encryption** for the *remaining graph* from Y + 3 = B, P + 4 = T, T + 5 = Y, L + 7 = S and O + 8 = W respectively.], construct letters of the plaintext: Y, P, T, L, O (this can be seen in The Cypher-Matrix form of the remaining graph is also. For example, from Y + 3 = B we get the character Y of the original Plain-text as Y = B – 3 and similarly, P = T – 4, T = Y – 5, L = S – 7, O = W - 8 from the Cipher-Matrix i.e., the letters Y, P, T, L, O.

2. Similarly, from the Encryption Array, also construct the letters of the Plain-text: C, I, S, G, O, R, C. For example, C = N – 11, I = S – 10, S = E – 12 from C + 11 = N, I + 10 + S, S + 12 = E respectively.

3. Regarding the 2nd element of each ordered pair from Encryption- Array and *remaining graph*, arrange the 2nd elements [from step 4. of **Encryption** and step 1. of **Decryption** taken together]:– the numbers in ascending order and the corresponding letters of Plain-text also will be simultaneously arranged in ascending order giving the original Plain-text.

(D,1), (T,2), (B,3), (T,4), (Y,5), (U,6), (S,7), (W,8), (P,9), (S,10), (N,11), (E,12)

The plaintext CRYPTOLOGICS is obtained as follows:

D – 1 = C
T – 2 = R
B – 3 = Y
T – 4 = P
Y – 5 = T
U – 6 = O
S – 7 = L
W – 8 = O
P – 9 = G
S – 10 = I
N – 11 = C
E – 12 = S

ANY numbers can be assigned to the letters of the plaintext, such as 5, 10, 16…, and we can proceed similarly with an infinite number of ciphertexts.

Retrieval Number:100.1/ijam.A116704010424
DOI: 10.54105/ijam.A1167.04010424
Journal Website: www.ijam.latticescipub.com

Published By:
Lattice Science Publication (LSP)

A variety of operations such as addition, subtraction, *addition modulo 7* (7 is the number of vertices) or any other combination of *multiplication modulo 7* and *addition modulo 7* can be taken to hide the original message still further and in decryption, the reverse operation will be done thus yielding an infinite number of Cipher-texts impossible or too difficult to crack. The approach of *modified operations, such as modulo arithmetic, can also be applied to the other methods discussed in subsections A and B to further conceal the plaintext in encryption. In decryption, the reverse operation can be performed*. For demonstration purposes, addition and subtraction are illustrated.

III. CONCLUSION

In this work, *tree traversal algorithms such as Inorder, Preorder, and Postorder, which yield Infix, Prefix, and Postfix notations, respectively, have been demonstrated for generating cyphertexts* with and without *expression trees.* Kruskal's algorithm for constructing a minimal spanning tree, creating a conceptually new modified ordered rooted tree of the minimal spanning tree, and utilising the newly developed Cypher-Matrix of the remaining graph (the original graph without the minimal spanning tree) has been presented. Additionally, the modified graceful labelling, which allows the repetition of exactly one vertex label for certain graphs such as the binary tree and double star, has been employed to create highly hidden Ciphertexts. The idea of Basic–Cypher-text has also been used to conceal the Plain Text even more and has been demonstrated in tree traversal algorithms and for the double-star graph. All these concepts and methods yield highly cryptic ciphertexts that are impossible or too difficult to crack. Encryption and decryption algorithms for these concepts have been presented in this work for their practical applications in Cryptology, enhancing data security and safety during transfer.

DECLARATION STATEMENT

Funding	No, we did not receive.
Conflicts of Interest	No conflicts of interest to the best of our knowledge.
Ethical Approval and Consent to Participate	No, the article does not require ethical approval and consent to participate with evidence.
Availability of Data and Materials	Not relevant.
Authors Contributions	I am only the sole author of the article.

REFERENCES

1. Kolman B., Busby R.C. and Ross S.C. (2002) Discrete Mathematical Structures, Pearson Education India, Delhi, India.
2. Katz J, (2003), "Binary Tree Encryption: Constructions and Applications", International Conference on Information Security and Cryptology – ICISC 2003, Part of the Lecture Notes in Computer Science book series, 1-11, Springer Link. https://doi.org/10.1007/978-3-540-24691-6_1
3. Auparajita Krishnaa and M.S. Dulawat (2011), "Study of the Effect of the Repeated Vertex Labels", *Ultra Scientist of Physical Sciences*, 23 (1), 176-180.
4. Shoba M Shanmugam (2017), "Cryptographic Techniques using Binary Tree and Tree Traversal", *International Journal of Scientific Research in Science and Technology, Vol 3, Issue* 1, 175-180.
5. Sivakumar T., Humshavarthini K, Jayasree M and Eswaran M (2017), "Data Encryption Using Binary Tree Traversal (DEBTT*)", International Journal of Advanced Technology in Engineering and Science, vol.* 5, issue no. 04, 353-362.
6. Aupaarajita Krishnaa (2019) "Inner magic and inner antimagic graphs in cryptography", *Journal of Discrete Mathematical Sciences and Cryptography*, 22 (4), 1057-1066, https://doi.org/10.1080/09720529.2019.1675298
7. Auparajita Krishnaa (2021), "Certain Specific Graphs in Cryptography", *Advances and Applications in Discrete Mathematics*, 26(2), 157-177. https://doi.org/10.17654/DM026020157
8. Auparajita Krishnaa (2022), "Lattices in Cryptology", *Aryabhatta Journal of Mathematics and Informatics*, 14(1), 111-124. https://doi.org/10.5958/2394-9309.2022.00063.4
9. Auparajita Krishnaa and Dharmendra Kumar Gurjar (2023), "The Concept of Basic Cypher Text for Enhancing Security in the Algorithms of Cryptography Applications Using Labelled Graphs", *International Journal of Scientific Research in Mathematical and Statistical Sciences,*10(3), 09-13.

AUTHOR'S PROFILE

Auparajita Krishnaa, the author, has academic qualifications and experience in teaching and research in Computer Science and Discrete Mathematics. The author has several original research publications with national and international publishers of repute, which cover areas like Logic, Graph Labelling, Statistical Analysis of Graph Labellings, Consciousness, Cryptography, Cryptology, Computational Complexity, etc. The work done in these publications have found applications and aroused interest in various related as well as unrelated fields as well such as Chemistry, Physics, Biology, Medicine, Neural Networks, Artificial Intelligence, Engineering, Magic (Programming), Computer Science, Algebra, Analysis, Number Theory and fields like Psychology and Philosophy etc. The author intends to focus on problem-solving for various aspects of existence using features of Discrete Mathematics.