International Journal of Education, Modern Management, Applied Science & Social Science (IJEMMASSS)
ISSN :2581-9925, Impact Factor: 7.555, Volume 07, No. 02(I), April- June, 2025, pp. 41-50

Advancing Cryptographic Security through Graph Theory: A Comprehensive Review

Hemant Gena[1] | Binny Kakkar[2*]

[1]Research Scholar, Department of Mathematics, SKD University, Hanumangarh, Rajasthan, India.
[2]Department of Mathematics, SKD University, Hanumangarh, Rajasthan, India.

*Corresponding Author: binnykakkar29@gmail.com

ABSTRACT

Graph theory has become a pivotal area of research due to its broad applications in fields such as biochemistry (genomics), coding theory, communication networks, and cyber security. Recently, its integration into cryptography has garnered significant attention. This paper reviews the various ways in which graph theory is applied to cryptography, focusing on cryptographic algorithms that leverage general graph theory concepts, external graph theory, and expander graphs. In addition, we explore the potential of graph-theoretical models in enhancing cryptographic primitives, such as secure key exchange protocols, digital signatures, and encryption schemes. A novel perspective is presented on how graph theory could be used to construct new types of cryptographic systems with increased security and efficiency. Furthermore, we discuss emerging trends, including the use of quantum-resistant graph-based cryptographic protocols and the role of random graphs in improving the scalability of cryptosystems. This paper highlights the promising synergy between graph theory and cryptography and suggests future research directions to optimize these interdisciplinary approaches.

Keywords*: Pivotal Area, Biochemistry, Genomics, Coding Theory, Cyber Security, Cryptography.*

Introduction

In recent years, the explosive growth of computers and their network connections has highlighted the increasing need for securing digital communications, particularly in sensitive areas such as military and governmental sectors. These sectors involve the exchange of vast amounts of confidential information, making confidentiality, authentication, and access control essential. This has led to a surge in research focused on cryptography and security, aiming to protect systems and information from an ever-growing range of cyber threats. Modern cryptography relies heavily on mathematical principles, particularly from Complexity Theory, to ensure strong security. One key concept in this area is the provable hash function. A hash function is considered "provable" if finding a collision in it is equivalent to solving another well-known hard problem, such as integer factorization or discrete logarithms. This assumption forms the basis for the security of current cryptosystems, with the idea that certain computational problems cannot be solved efficiently. While traditional mathematical and algorithmic methods have been central to modern cryptography, recent years have seen a growing interest in applying graph theory concepts to enhance cryptographic security. Graph theory offers a wide range of unique tools that can be used to tackle cryptographic challenges. For example, bipartite graphs, which represent two sets of vertices with edges between them, are particularly useful in secure communication systems and key exchange protocols. Similarly, Hamiltonian graphs, which focus on cycles that visit each vertex exactly once, have applications in problems involving optimal data routing and network flow in distributed systems.

Other interesting areas in graph theory, such as Eulerian graphs, which deal with paths that visit every edge exactly once, and extremal graphs, which focus on optimizing specific graph properties, are also being explored for cryptographic purposes. Eulerian graphs can be particularly useful in routing problems within cryptographic protocols, where data needs to flow efficiently without revisiting paths. Extremal graphs help in identifying optimal graph structures that can lead to more efficient and secure

cryptographic algorithms. Moreover, expander graphs, known for their excellent connectivity properties, are being increasingly applied to cryptography. These graphs are particularly effective in scenarios that require fast and secure information dissemination, such as in block chain technologies or secure multi-party computations. In this paper, we explore the application of graph theory concepts, such as bipartite, Hamiltonian, Eulerian, extremal, and expander graphs, in the development of cryptographic methods. We will begin by reviewing the use of simple and directed graphs in algorithm design, followed by an exploration of extremal graphs, Hamiltonian, Eulerian and bipartite graph. This study aims to highlight how graph theory and cryptography together can lead to innovative solutions for securing digital communications.

Literature Review

Several strategies for exploring the applications of labelled graphs are discussed in [1]. Graph labelling refers to the assignment of real values to vertices and edges of a graph under specific conditions. These techniques are driven both by their usefulness in practical domains and their inherent mathematical appeal. The concept of graph labelling emerged during the 1960s, and since then, a wide variety of labelling methods have been developed, appearing in over a thousand research papers. The field continues to evolve due to the growing interest in its application-based models.

In [2], the notions of **inner magic** and **inner anti magic** labelling were introduced. As per this study, "an inner magic labelling" of a graph, with vertices, edges, and internal faces, assigns labels such that the weights of the internal faces follow an arithmetic sequence with a common difference. If the graph is termed as **inner magic**, and if it is referred to as **inner anti magic**. These labelling methods were utilized in [3] for secure data transmission in cryptographic applications.

The work in [4] explores the role of **super mean** and **magic labelling** in cryptography. Additionally, [5] presents fundamental ideas regarding graph-based encryption, while [6] demonstrates the use of specific graphs for cryptographic implementation.

The correlation between **randomness** and **cryptography** is examined in [8], and [9] describes algorithms for encryption and decryption grounded in graph theory principles, particularly for symmetric key cryptography. Computers may attempt to generate randomness through various methods, including algorithmic number generation or by observing unpredictable physical events—like temperature fluctuations, system interrupts, or mouse movement. Although these sources don't always produce ideal randomness, they are often adequate for practical encryption purposes. Research by Yevgeniy focuses on defining the precise conditions that determine when a source of randomness is sufficient and on developing efficient ways to utilize such sources.

A method leveraging **bipartite graphs** for encrypting messages is proposed in [10]. In [11], the use of labelled graphs in applications such as cryptography is further explored. A **bipartite graph** can be colour using two colour such that no two adjacent vertices share the same colour. This property implies that the graph's vertex set can be divided into two groups: each edge connects a vertex from one group to a vertex in the other, and no edges exist within the same group.

Graph theory has become an essential tool in cryptography, leading to innovative approaches for securing data. Several techniques leverage graph structures, directed graphs, and related concepts to design cryptographic algorithms, secret sharing schemes, and encryption systems. [12**]**Samid (2004) encryption method uses a graph as the key, where a path on the graph represents the plaintext and the edges form the cipher text. This approach directly applies graph traversal to encryption, ensuring secure communication. Quick trickle permutations, where the spacing between elements is distinct, are linked to Complete Latin squares. [13**]** Mittenthal (2007) work on directed graphs and Latin squares enhances block encryption systems by introducing structured permutations to protect data. Secret sharing schemes introduced by Shamir (1979) [14**]** enable an authority to divide a secret into multiple shares, which are distributed among a group of participants. Only a specific subset of these participants can combine their shares to reconstruct the original secret. These schemes are crucial in distributed cryptographic systems where trust is shared among multiple parties. Visual cryptography schemes (VCS), developed by Naor and Shamir(1994) [15**]**, encode images into shares, and the secret is revealed when the shares are stacked. Lu et al.(2008) [16**]** extended this concept using graphs, where both vertices and edges carry images, enabling the sharing of multiple secret. Visual cryptography using graphs allows for the sharing of multiple secrets through transparency overlays. The study of Shamir (1979) [14**]** demonstrated that the complexity of these schemes depends on the chromatic number of the graph's cube, influencing scalability and security. Steve Lu et al.(2008) extended this concept to enable sharing of multiple secret

images on graphs. Directed graphs represent quick trickle permutations, allowing for flexible encryption sequences. Mittenthal (2007) [13**]** methods enable the rearrangement of these sequences, improving the security of block ciphers.

Algorithm for Encryption

Construction of Encryption Table:- We allocate the integers 0,1,2,…, m to represent the rows, and assign the numbers m+1,m+2,m+3,…, n to denote the column. The characters from the set S, which consists of all 26 letters of the alphabet, a @, and a dot (.), are then randomly distributed across this table. The characters in S are the ones used to form the original message. A sample representation of this layout can be seen in Table 1

Table (A): Encryption Table

	7	8	9	10
0	A	H	O	V
1	B	I	P	W
2	C	J	Q	X
3	D	K	R	Y
4	E	L	S	Z
5	F	M	T	@
6	G	N	U	.

In this scheme, each character is mapped to a unique numerical value based on its group classification. The character set is divided into two distinct groups:

Group 1 includes all vowels (A, E, I, O, U), the @ character, and the dot (.).

Group 2 consists of all remaining consonants.

The method of assigning numeric values differs for each group:

- For Group 2 (Consonants): The first digit of the assigned number corresponds to the row index, and the remaining digits represent the column index.
- For Group 1 (Vowels, @, and dot): The last digit corresponds to the row index, and the preceding digits represent the column index.

Examples:

A = 70

F = 57

R = 39

X = 210

O = 90

U = 96

@ = 105

Dot = 106

Construction of Graph

In this phase, the plain text (original message) is encoded into a graph structure. The process involves the following:

- Represent each character in the message as a vertex of a complete graph K_n where n is the number of characters in the message M.
- Each character is converted into its numeric equivalent (from Table a) and assigned to a vertex.
- Edges are then labelled by computing the modulus of the difference between the numerical labels of the connected vertices.

Graph-Based Matrices Generation

Two matrices are constructed from the labeled graph:

- Matrix A: A complete graph matrix derived from the labeled graph K_n.

- Matrix B: A cycle matrix formed by removing the inner edges of the complete graph, retaining only the outer cycle.

Encoding Matrix Modification

To form the modified cycle matrix 'B*'

Replace the **diagonal elements** of matrix B with the numeric values of the original characters (from Table 2, the Alphabet Encoding Table.

Table (B): Alphabet Encoding Table

A	1	N	14
B	2	O	15
C	3	P	16
D	4	Q	17
E	5	R	18
F	6	S	19
G	7	T	20
H	8	U	21
I	9	V	22
J	10	W	23
K	11	X	24
L	12	Y	25
M	13	Z	26

Cipher Matrix Generation

Compute the intermediate matrix N as follows:

$N=A\times B^{*}$

Where A is the complete graph matrix, B* is the modified cycle matrix.

Cipher Matrix Generation

To derive the first cipher matrix C_1:

- Multiply matrix N with a key matrix K.
- The key matrix K is an upper triangular matrix of order n×n, where n is the length of the original message.

$$k=\begin{bmatrix}1 & 2 & 3 & . & . & . & n\\ 0 & 1 & 2 & 3 & . & . & n-1\\ 0 & 0 & 1 & 2 & & . & n-2\\ 0 & 0 & 0 & 1 & 2 & . & .\\ 0 & 0 & 0 & 0 & 1 & 2 & .\\ 0 & 0 & 0 & 0 & 0 & 1 & .\\ 0 & 0 & 0 & 0 & 0 & 0 & 1\end{bmatrix}$$

Figure 1: Illustrates the Structure of the Upper Triangular Key Matrix

Figure (a) Upper Triangular Key Matrix

Selection of Hamiltonian Cycle

From the complete graph K_n, a Hamiltonian Cycle is chosen. Since a complete graph with n vertices has (n−1)! Hamiltonian cycles, one specific cycle is selected using the Algorithm of Nearest-Neighbor.

Algorithm of Nearest-Neighbor

- Begin at the vertex that represents the first character of the original message.
- From the current vertex, move to the adjacent vertex connected by the smallest edge label (minimum value).
- If multiple edges have the same minimum value, select randomly among them.
- Continue this process until all vertices are visited exactly once and the cycle returns to the starting vertex, forming a Hamiltonian Cycle.

Second Key Generation

- Calculate the sum of the edge labels along the chosen Hamiltonian Cycle.
- Let this sum be denoted by S. This value acts as the second encryption key.

Final Cipher Matrix

- Apply modulus S on each entry of the first cipher matrix C_1 to produce a new matrix C_2:
- The cipher text is then derived by linearizing matrix C_2.

Algorithm for Decryption Process

To retrieve the original message, follow these steps:

- **Reshape Cipher Text:** Convert the linear cipher text into the matrix form C_2.
- **Recover C_1:** Use the known key S to reverse the modulus operation and retrieve matrix C_1.

Compute Matrix N

$N=C1\times K^{-1}$

where K^{-1} is the inverse of the key matrix K.

Recover Modified Matrix B*

$B*=A^{-1}\times N$

where $A^{-1}$is the inverse of matrix A.

Extract Original Characters: Read the diagonal entries of B* and decode the corresponding character values using the Alphabet Encoding Table (Table b).

The result is the original plain text message.

Encryption Example

Let's consider the original message: "HELP".

Since the message contains 4 characters, construct a complete graph K_4.

Character-to-Value Conversion (Using Table 1):

Character	Numeric Value
H	08
E	74
L	48
P	19

Vertex Assignment

V_1	08
V_2	74
V_3	48
V_4	19

To label the edges, compute the **modulus of the difference** between the values of connected vertices.

For example, for edge (V_1, V_2), the label is:

$e_1=|08-74|=66$

$e_2=|74 - 48| =26$

$e_3=|48-19|=29$

$e_4=|19-08|=11$

$e_5=|08-48|=40$

$e_6=|19-74|=55$

Repeat this for all edges in K_4

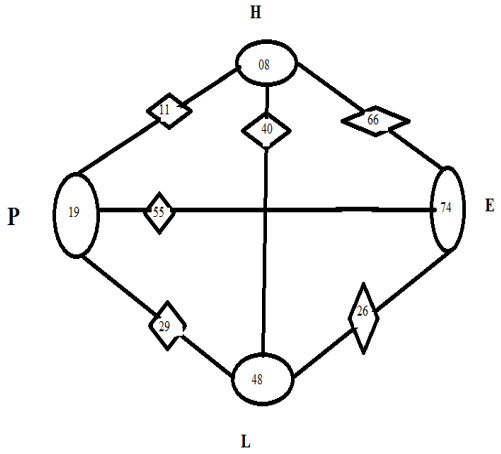


Figure (b) Complete Graph

We obtained a complete graph matrix from above graph which is denoted as “A”

$$A = \begin{bmatrix} 0 & 66 & 40 & 11 \\ 66 & 0 & 26 & 55 \\ 40 & 26 & 0 & 29 \\ 11 & 55 & 29 & 0 \end{bmatrix}$$

Figure (c) Complete Graph Matrix

Obtain a New Cycle of Length 4 from k_4

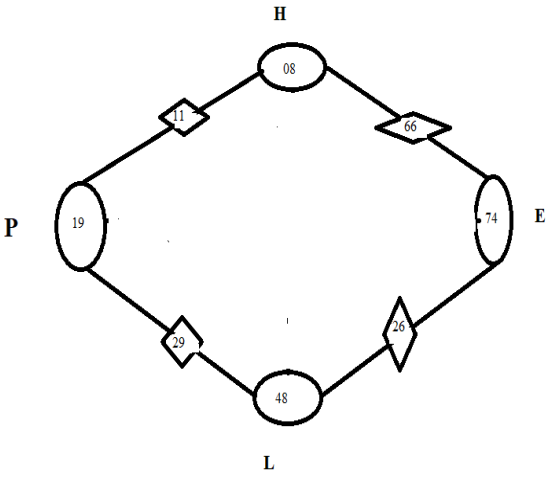


Figure (d) Cycle of Length 4

Then we obtain matrix B with the help of cycle of length 4

$$B = \begin{bmatrix} 0 & 66 & 0 & 11 \\ 66 & 0 & 26 & 0 \\ 0 & 26 & 0 & 29 \\ 11 & 0 & 29 & 0 \end{bmatrix}$$

Now matrix B* by using alphabet encoding table

$$B* = \begin{bmatrix} 8 & 66 & 0 & 11 \\ 66 & 5 & 26 & 0 \\ 0 & 26 & 12 & 29 \\ 11 & 0 & 29 & 16 \end{bmatrix}$$

$$N = A \times B * = \begin{bmatrix} 0 & 66 & 40 & 11 \\ 66 & 0 & 26 & 55 \\ 40 & 26 & 0 & 29 \\ 11 & 55 & 29 & 0 \end{bmatrix} \times \begin{bmatrix} 8 & 66 & 0 & 11 \\ 66 & 5 & 26 & 0 \\ 0 & 26 & 12 & 29 \\ 11 & 0 & 29 & 16 \end{bmatrix} = \begin{bmatrix} 4477 & 1370 & 2515 & 1336 \\ 1133 & 5032 & 1907 & 2360 \\ 2355 & 2770 & 1517 & 904 \\ 3718 & 1755 & 1778 & 962 \end{bmatrix}$$

The key matrix will be 4x4 order since original message has 4 characters

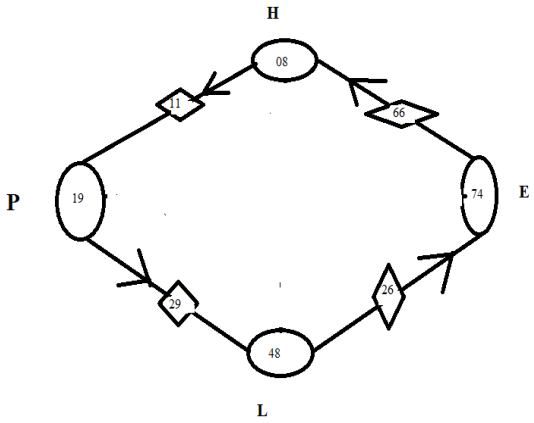
$$k = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Now C_1 is obtained by C_1=NxK

$$C_1 = N \times K = \begin{bmatrix} 4477 & 1370 & 2515 & 1336 \\ 1133 & 5032 & 1907 & 2360 \\ 2355 & 2770 & 1517 & 904 \\ 3718 & 1755 & 1778 & 962 \end{bmatrix} \times \begin{bmatrix} 1 & 2 & 3 & 4 \\ 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 4477 & 10324 & 18686 & 28384 \\ 1133 & 7298 & 15370 & 25802 \\ 2355 & 7480 & 14122 & 21668 \\ 3718 & 9191 & 16442 & 24655 \end{bmatrix}$$

Hamiltonian Cycle Construction using the Nearest-Neighbor Algorithm

To determine the Hamiltonian Cycle, we employ the **Nearest-Neighbor Algorithm**, starting with the vertex that represents the first character of the message. In this case, the message begins with the character **'H'**, so the algorithm initiates from the vertex labelled with the value for 'H'.



Second key S=11+29+26+66=132

We will apply mod 132 on matrix C_1 to get matrix C_2

C_2= C_1 mod 132

$$C_2 = \begin{bmatrix} 121 & 28 & 74 & 4 \\ 77 & 38 & 58 & 62 \\ 111 & 88 & 130 & 20 \\ 22 & 83 & 74 & 103 \end{bmatrix}$$

And the quotient matrix be

$$Q = \begin{bmatrix} 33 & 78 & 141 & 215 \\ 8 & 55 & 116 & 195 \\ 17 & 56 & 106 & 164 \\ 28 & 69 & 124 & 186 \end{bmatrix}$$

Then the receiver will get the cipher text given below

121 28 74 4 77 38 58 62 111 88 130 20 22 83 74 103

Decryption Algorithm

- **Step 1: Inputs Required for Decryption**

  To decrypt the encrypted message, the following components are needed:

  - **C_2**: Final Cipher Matrix
  - **Q**: Quotient Matrix (used for reversing the modulo operation)

- **S**: Second encryption key (calculated using Hamiltonian Cycle), given as **S = 132**
- **K**: Key Matrix (upper triangular matrix used during encryption)
- **A**: Complete Graph Matrix (constructed during encryption)

Step 2: Reconstruct the Final Cipher Matrix C_2

Given the Cipher Text as a linear sequence:

121 28 74 4 77 38 58 62 111 88 130 20 22 83 74 103

This sequence is converted into a **4×4 matrix** (since the original message had 4 characters):

$$C_2 = \begin{bmatrix} 121 & 28 & 74 & 4 \\ 77 & 38 & 58 & 62 \\ 111 & 88 & 130 & 20 \\ 22 & 83 & 74 & 103 \end{bmatrix}$$

Then we obtain the first cipher matrix C_1 with the help of C_2 and matrix Q and the key S=132

As $[Q]_{ij} \times 124 + [C2]_{ij} = [C1]_{ij}$

The we get C_1as

$$C_1 = \begin{bmatrix} 4477 & 10324 & 18686 & 28384 \\ 1133 & 7298 & 15370 & 25802 \\ 2355 & 7480 & 14122 & 21668 \\ 3718 & 9191 & 16442 & 24655 \end{bmatrix}$$

We compute the inverse of key matrix k as

$$K^{-1} = \begin{bmatrix} 1 & -2 & 1 & 0 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Now find matrix N as

N=C_1XK^{-1}

$$N = \begin{bmatrix} 4477 & 10324 & 18686 & 28384 \\ 1133 & 7298 & 15370 & 25802 \\ 2355 & 7480 & 14122 & 21668 \\ 3718 & 9191 & 16442 & 24655 \end{bmatrix} \times \begin{bmatrix} 1 & -2 & 1 & 0 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & -2 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 4477 & 1370 & 2515 & 1336 \\ 1133 & 5032 & 1907 & 2360 \\ 2355 & 2770 & 1517 & 904 \\ 3718 & 1755 & 1778 & 962 \end{bmatrix}$$

Now find inverse of matrix “A” as

$$A^{-1} = \begin{bmatrix} -5/132 & 1/132 & 0 & 1/22 \\ 1/132 & -5/429 & 1/52 & 0 \\ 0 & 1/52 & -55/1508 & 1/58 \\ 1/22 & 0 & 1/58 & -20/319 \end{bmatrix}$$

Now obtain B* as

B*=$A^{-1}XN$

$$B^{*=}A^{-1} \times N = \begin{bmatrix} -\frac{5}{132} & \frac{1}{132} & 0 & \frac{1}{22} \\ \frac{1}{132} & -\frac{5}{429} & \frac{1}{52} & 0 \\ 0 & \frac{1}{52} & -\frac{55}{1508} & \frac{1}{58} \\ \frac{1}{22} & 0 & \frac{1}{58} & -\frac{20}{319} \end{bmatrix} \times \begin{bmatrix} 4477 & 1370 & 2515 & 1336 \\ 1133 & 5032 & 1907 & 2360 \\ 2355 & 2770 & 1517 & 904 \\ 3718 & 1755 & 1778 & 962 \end{bmatrix}$$

$$= \begin{bmatrix} 8 & 66 & 0 & 11 \\ 66 & 5 & 26 & 0 \\ 0 & 26 & 12 & 29 \\ 11 & 0 & 29 & 16 \end{bmatrix}$$

Now from the diagonal entries of B* we get our message by using alphabet encoding table

8=H

5=E

12=L

16=P

Therefor the original message is “HELP”

Conclusion

This work presents a multi-layered encryption scheme designed to securely conceal plain text messages. The method utilizes a complete graph constructed according to the length of the message, incorporating both graph theory principles and a Hamiltonian cycle.

We first generate a Complete Graph Matrix and a Cycle Matrix, assigning edge weights based on character values from a predefined Encryption Table. The Cycle Matrix is then modified using values from the Alphabet Encoding Table. These two matrices—Graph Matrix (A) and Modified Cycle Matrix (B*)—are combined using a shared key matrix (K), which is an upper triangular matrix. The Hamiltonian cycle provides a secondary key S, used to perform modular operations that lead to the construction of the Final Cipher Matrix (C_2). The output is a cipher text that is highly obfuscated and resistant to direct decryption.

The decryption process reverses these steps by calculating:

- Matrix C_1 from C_2 using key S,
- Matrix N as $C_1 \times K^{-1}$,
- Matrix B* as $A^{-1} \times N$,

and finally retrieving the original message from the diagonal entries of B* using the Alphabet Encoding Table.

This technique ensures that the original message remains well-hidden through a sequence of graph-based transformations and matrix operations, offering a robust cryptographic mechanism suitable for secure data transmission.

References

1. Krishnaa A., An Example Usage of Graph Theory in Other Scientific Fields: On Graph Labeling, Possibilities and Role of Mind/C onsciousness, Chapter in the book titled Graph Theory: Advanced Algorithms and Applications, IntechOpen, London, UK (2018).
2. Krishnaa A. and Dulawat M.S., Algorithms for Inner Magic and Inner Antimagic Labelings for Some Planar Graphs, Informatica (Lithuania), 17(3) (2006) 393-406.
3. Krishnaa A., Inner magic and inner antimagic graphs in cryptography Journal of Discrete Mathematical Sciences and Cryptography, 22(6) (2019) 1057-1066.
4. I.W. Sudarsana, S.A. Suryanto, D. Lucianti and N P A P S Putri, an application of super mean and mean graphs labeling in cryptography system, J. of Physics, Conference Series, 1763, The 2nd International Seminar on Sciencce and Technology, Palu, Indonesia. Published under license by IOP Publishing Ltd (2020).
5. Ustimenko V.A., on graph based cryptography and symbolic computations, Serdica Journal of Computing I, (2007) 131-156.
6. Krishnaa A., Certain specific graphs in cryptography, Advances and Applications in Discrete Mathematics, 26(2) (2021) 157-177.
7. Shamir Adi, Random graphs in cryptography, The Weizman Institute, Israel, The Onassis Foundation Science Lecture Series, 28 (2010).
8. Perera P.A.S. and Wijesiri G.S., Encryption and decryption in symmetric key cryptography using graph theory, (2021).
9. Etaiwi W. M. A., Encryption Algorithm using Graph Theory. Journal of Scientific Research and Reports. 3(19) (2014) 2519-2527.
10. Yamuna M. and Karthika K**.,** Data transfer using bipartite Graphs. International Journal of Advance Research in Science and Engineering (IJARSE), 4(2) (2015).

11. Krishnaa A., Some Applications of Labelled Graphs, International Journal of Mathematics Trends and Technology, 37(3) (2016).
12. Samid Gideon, Denial Cryptography based on Graph Theory, US patent 6823068-2004 http://www.patentstorm.us/patents/6823068.html
13. Lothrop Mittenthal, Sequencings and Directed Graphs with Applications to Cryptography, S.W. Golomb et al. (Eds.): *Springer-Varlag LNCS* 4893, pp 70-81, 2007
14. Adi Shamir. How to share a secret. *Commun. ACM*, 22(11) pp. 612-613, 1979.
15. Moni Naor and Adi Shamir. Visual cryptography. In *Advances in Cryptology - EURO-CRYPT'94*, *LNCS*, vol 950, pp 1-12, 1994.
16. Steve Lu, Daniel Manchala and Rafail Ostrovsky, Visual Cryptography on Graphs, *COCOON 2008*: pp. 225-234, 2008.
17. Vasyl Ustimenko CRYPTIM: Graphs as tools for symmetric encryption in: Lecture Notes in Comput. Sci., vol. 2227, Springer, New York,2001.
18. Vasiliy A. Ustimenko, Graphs with Special Arcs and Cryptography,*Acta Applicandae Mathematicae*, Vol. 74, No. 2, pp. 117-153, 2002,
19. Ustimenko V.A. , On Linguistic Dynamical Systems, Graphs of Large Girth and Cryptography, *J. of Mathematical Sciences*, Springer, 140-3, pp. 412-434, 2007. APPLICATIONS OF GRAPH THEORY 217
20. V. A. Ustimenko, On Graph-Based Cryptography and Symbolic Computations, *Serdica J. Computing*, pp. 131-156, 2007
21. J.S.Kotorowicz_ and V.A.Ustimenko, On the Implementation of Cryptoalgorithms Based on Algebraic Graphs Over Some Commutative Rings, *Condensed Matter Physics* Vol. 11, No 2(54), pp. 347–360, 2008
22. Vasyl Ustimenko, On the hidden discrete logarithm for some polynomial stream ciphers, *Proceedings of the IMCSIT*. Volume 3, pp.297–301, 2008
23. Vasyl Ustimenko, On Extremal Graph Theory for Directed Graphs and its Applications to Information Security, http://www.imath.kiev.ua/~congress2009/Abstracts/Ustymenko.pdf
24. William Stallings, Cryptography and Network Security Principles and Practices, Prentice Hall India, 2006.

❖◆❖