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. 11-18

Advanced Encryption Algorithms for Secure Data Transmission Using Graph-Based Structures

Surbhi Sonia[1] | Binny Kakkar[2*]

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

*Corresponding Author: binnykakkar29@gmail.com

ABSTRACT

In today's interconnected world, safeguarding sensitive information through secure communication is more important than ever. Traditional cryptographic techniques often face challenges in defending against modern, sophisticated attacks. This paper proposes innovative encryption algorithms that combine the power of corona graphs, bipartite graphs, and complete graphs, along with algebraic properties, to enhance data security. The corona graph, formed by attaching copies of one graph to the vertices of another, adds a layer of complexity, while bipartite graphs offer a structured division of vertices into two sets, making the system more resistant to unauthorized access. The use of complete graphs further strengthens the system by ensuring that every pair of vertices is connected, contributing to an intricate and secure encryption framework. By integrating these graph structures, the proposed encryption methods provide a robust, multi-layered approach that offers enhanced protection against both classical and contemporary cryptographic attacks. This novel combination of graph-theoretical concepts paves the way for more secure communication, ensuring the confidentiality and integrity of transmitted data.

***Keywords**: Cryptographic Techniques, Sophisticated Attacks, Encryption Algorithms, Corona Graph, Data Security.*

Introduction

In the digital age, ensuring secure communication is more important than ever, as sensitive data is constantly transmitted over networks. Cryptography, the science of securing communication, plays a crucial role in protecting the integrity and confidentiality of this information. Over the years, various encryption techniques have been developed, ranging from classical methods like the Caesar cipher to modern symmetric and asymmetric encryption systems. However, as technology advances, so do the methods employed by cyber threats to compromise data security. Therefore, there is a continuous need for exploring new and innovative cryptographic techniques that can provide stronger protection against these evolving threats.

This paper introduces a novel approach to encryption and decryption that combines concepts from graph theory and matrix operations. The proposed technique uses a complete graph structure to represent characters of a message as vertices, where each character is assigned a unique numeric value using XOR operations. The edges of the graph are labeled based on the absolute differences between these numeric values, creating a complex network of relationships. By applying matrix transformations and calculating the inverse of key matrices, the method transforms the original message into a cipher text that is resistant to unauthorized decryption. The decryption process involves reversing these matrix operations and utilizing the inverse transformations to recover the original message. The system utilizes a combination of XOR-based encryption, matrix multiplication, and graph-theoretic concepts to ensure the secure encoding and decoding of data. This approach highlights the power of graph theory in creating secure communication protocols, offering a robust alternative to traditional encryption techniques.

Through this method, we aim to demonstrate how abstract mathematical concepts such as graphs and matrices can be leveraged to create practical cryptographic solutions. The paper also discusses the security benefits of this approach, showcasing its potential for applications in fields where data integrity and privacy are paramount. By exploring this novel encryption method, we contribute to the ongoing development of more secure, efficient cryptographic systems for the modern digital landscape.

Literature Review

Graph theory has become an essential tool in the realm of cryptography, offering unique solutions to complex security challenges. A considerable body of research explores various graph-based approaches; particularly in the areas of encryption, decryption, and the construction of secure cryptographic systems. Krishnaa A. (2016, 2018, 2021) has extensively investigated the use of labeled graphs in cryptography. In her work, she addresses the application of inner magic and inner antimagic labeling for certain classes of graphs. These studies provide foundational insights into how graph labeling can enhance the security features of cryptographic algorithms (Krishnaa, 2016, 2021). Krishnaa's exploration of the magic labeling concept extends to cryptographic **systems**, offering novel approaches for constructing encryption keys based on the properties of specific graphs, including planar graphs (Krishnaa & Dulawat, 2006). This research emphasizes the potential for graph labeling methods to bolster cryptographic security by providing unique identifiers and key structures.

A significant contribution by Krishnaa (2019) in the field of cryptography explores the inner magic and antimagic graphs and their role in ensuring secure encryption. These labeled graphs, with their inherent properties, contribute to stronger cryptographic systems by utilizing graph-theoretic concepts such as Hamiltonian cycles and complete graph**s**, which play critical roles in the construction of secure cryptographic primitives.

Parallel to Krishnaa's works, other researchers have contributed significantly to the development of graph-based cryptographic methods. Sudarsana et al. (2020) presented an application of mean and super mean graph labeling in cryptographic systems, highlighting their use in constructing robust encryption schemes. Their research contributes to the exploration of how different types of graph labeling can offer various security guarantees, particularly in symmetric key cryptography.

Shamir (2010) further strengthens the connection between random graphs and cryptographic systems, suggesting that the inherent randomness in graph structures can be leveraged to create more secure cryptographic protocols. This work complements the studies on graph labeling by introducing the role of random graphs in enhancing the unpredictability of cryptographic systems.

In terms of graph-based encryption techniques, Etaiwi (2014) discusses how graph theory can be applied to encryption algorithms, offering insights into the mathematical underpinnings of cryptographic systems. Perera and Wijesiri (2021) expand on this concept, specifically focusing on symmetric key cryptography**,** where graph theory plays a pivotal role in both encryption and decryption processes. These studies align with Krishnaa's contributions by demonstrating the practicality of graph labeling in securing cryptographic operations.

The role of complete graphs **and** Hamiltonian cycles in encryption was explored by Gurjar and Krishna (2021), who showed how these graph-theoretic constructs can be incorporated into cryptographic algorithms to enhance security. Their work highlights the potential of Hamiltonian cycles in the encryption-decryption process, offering a new perspective on how classical graph theory concepts can be applied to modern cryptographic challenges. Lastly, Ustimenko (2007) explored the broader implications of graph-based cryptography and symbolic computations, discussing the symbolic nature of graphs and how they can be used for more efficient encryption and decryption algorithms. Ustimenko's work is foundational, offering a theoretical framework that has inspired much of the modern research in the field.

Enhancing Encryption with Case Sensitivity, Special Characters, and Dynamic Tables

In the world of data security, encryption is an essential tool that converts readable information into an unreadable format. It ensures that sensitive data remains confidential, even if intercepted. Traditional encryption

Techniques, while useful, can often be vulnerable to attacks if the system lacks complexity. In this article, we will explore a more advanced approach to encryption by integrating case sensitivity, special characters, and dynamic tables into the encryption process. Additionally, we will explain how numeric values are assigned to these characters and provide a method for increasing the complexity of encryption through the use of XOR operations.

- **Introduction to Case Sensitivity in Encryption**

In many traditional encryption schemes, uppercase and lowercase letters are treated as identical. However, by incorporating **case sensitivity** into the encryption process, we can increase the level of complexity. This means that uppercase letters (A-Z) will be treated differently from lowercase letters (a-z). This addition helps create a larger set of possible encryption outcomes, making it more difficult for attackers to break the encryption.

Example: A (uppercase) will have a different numeric value than a (lowercase).Similarly, B and b, C and c, etc., will all have distinct values.

- **Including Special Characters in the Encryption System**

Special characters such as spaces, dots, and symbols (e.g., @, #, !, etc.) are often excluded from basic encryption tables. However, to make the encryption system more flexible and secure, we will include these special characters. This allows the system to encrypt messages that contain a wider range of symbols, making the encryption more dynamic.

- **Dynamic Table Structure**

A dynamic table adjusts its structure based on the characters being encrypted. While static tables have fixed values, dynamic tables change based on the input. This method allows us to encrypt a wide variety of characters, including special symbols, and adapt the encryption system for different types of messages.

- **Encryption Table with Case Sensitivity, Special Characters, and Dynamic Structure**

Numeric Value Assignment

Character	Num. Value	Character	Num. Value	Character	Num. Value	Character	Num. Value
A	1	T	20	m	39	5	58
B	2	U	21	n	40	6	59
C	3	V	22	o	41	7	60
D	4	W	23	p	42	8	61
E	5	X	24	q	43	9	62
F	6	Y	25	r	44	space	63
G	7	Z	26	s	45	DOT	64
H	8	A	27	t	46	@	65
I	9	B	28	u	47	#	66
J	10	C	29	v	48	$	67
K	11	D	30	w	49	%	68
L	12	E	31	x	50	&	69
M	13	F	32	y	51	*	70
N	14	G	33	z	52	(	71
O	15	H	34	0	53	)	72
P	16	I	35	1	54	!	73
Q	17	J	36	2	55	?	74
R	18	K	37	3	56	;	75
S	19	L	38	4	57	:	76

Explanation of the Table

Now, we will assign numeric values to each character in the table. The numeric values are assigned as follows:

- **Uppercase letters** (A-Z): These will be assigned numeric values from **0 to 25**.
- **Lowercase letters** (a-z): These will be assigned values from **26 to 51**.
- **Numbers** (0-9): Numbers will be assigned values from **52 to 61**.
- **Special Characters**: These will have values starting from **62** onward

- **Encrypting a Message Using XOR**

Let's apply XOR encryption to a sample message using the assigned numeric values and a **key**. This increases the security of the encryption.

Example Message: "Hello"

Convert the message to numeric values: H = 7, e = 30, l= 37, l = 37, o = 40

Apply XOR operation with a key (let's use 5 for this example):

H (7) XOR 5 = 2

e (30) XOR 5 = 25

l (37) XOR 5 = 32

l (37) XOR 5 = 32

o (40) XOR 5 = 45 Encrypted message: **[2, 25, 32, 32, 45]**

- **Encryption Algorithm**
  - **STEP 1:** We need to determine the numeric values of each character based on a specific lookup table, and then perform the XOR operation using these values.
  - **STEP 2:** Graph Creation and Labeling: We represent the original message as vertices in a complete graph K_n where n is the number of characters. Each character is converted into a number (using Table 1), and the edges are labeled based on the modulus of the difference between connected vertices.
  - **STEP 3:** Constructing Matrices A and B: Next, we create the Complete Graph Matrix A from the labeled graph. A cycle matrix B is also formed by removing internal edges from the graph.
  - **STEP 4:** Modifying the Cycle Matrix B^*: The diagonal of matrix B is updated with the numerical values of the original message characters from the Alphabet Encoding Table (Table 2), resulting in a new matrix B^*.
  - **STEP 5:** Multiply matrix A with matrix B* to obtain a new matrix 'N'.
  - **STEP 6:** Apply the modulo 73 (we take 72 values in table 1) to each element of the initial Cipher Matrix N in order to generate a new final Cipher Matrix 'N_1'. The Cipher Text is represented by the Matrix 'N_1' in a linearized form.

Decryption Algorithm

- **Matrix Form (N_1)**: Convert the message into a matrix form, N_1(with help of table1).
- **Apply Key Matrix (S)**: Use the key matrix S=73 to transform N_1 into a new matrix, N.
- **Compute B^***: Use the inverse of matrix A^{-1} to compute B^* by multiplying it with N.
- **Extract Diagonal**: Take the diagonal entries from matrix B^*then apply XOR inverse key and decode them using an alphabet table.
- **Reconstruct Plaintext**: The decoded values give you the original message.

In essence, it's all about transforming and reversing matrix operations to reveal the original text.

Illustration

- **Encryption Using the Complete Graph**

Assume the initial message is 'WateY@#'. Since it contains 7 characters, we will construct a K_7 Complete Graph and associate these characters with the graph's vertices, as illustrated in Figure 2.

- **Step 1: Assign Numeric Values to Each Character**

First, we need to find the numeric values for each character using a predefined table and using XOR opration.we get W=18,a=22,t=41,e=26,Y=20,@=60,#=61 using XOR opration key (k=5). Now, we assign these numeric values to vertices: V_1=18,V_2=22,V_3=41,V_4=26,V_5=20,V_6=60,V_7=61.

- **Step 2: Calculate Edge Labels**

Now that we have the vertices labeled, we can find the edge labels by calculating the absolute difference between the numeric values of the vertices. Here are the edge calculations

e_1 = |V1 - V2| = |18 - 22| = 04

e_2 = |V1 - V3| = |18 - 41| = 23

e_3 = |V1 - V4| = |18 - 26| = 08

e_4 = |V1 - V5| = |18 - 20| = 02

e_5 = |V1 - V6| = |18 - 60| = 42

e_6 = |V1 - V7| = |18 - 61| = 43

e_7 = |V2 - V3| = |22 - 41| = 19

e_8 = |V2 - V4| = |22 - 26| = 04

e_9 = |V2 - V5| = |22 - 20| = 02

e_{10} = |V2 - V6| = |22 - 60| = 38

e_{11} = |V2 - V7| = |22 - 61| = 39

e_{12} = |V3 - V4| = |41 - 26| = 15

e_{13} = |V3 - V5| = |41 - 20| = 21

e_{14} = |V3 - V6| = |41 - 60| = 19

e_{15} = |V3 - V7| = |41 - 61| = 20

e_{16} = |V4 - V5| = |26 - 20| = 06

e_{17} = |V4 - V6| = |26 - 60| = 34

e_{18} = |V4 - V7| = |26 - 61| = 35

e_{19} = |V5 - V6| = |20 - 60| = 40

e_{20} = |V5 - V7| = |20 - 61| = 41

e_{21} = |V6 - V7| = |60 - 61| = 01

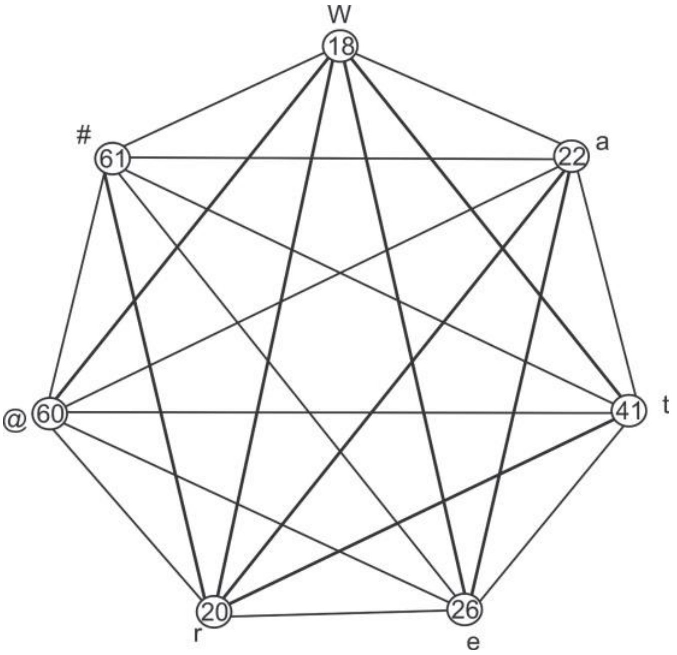

Figure 2: complete graph K_7

Generate a new labeled adjacency matrix for the complete graph shown in Figure 2, and denote it by 'A'.

$$A = \begin{bmatrix} 0 & 4 & 23 & 8 & 2 & 42 & 43 \\ 4 & 0 & 19 & 4 & 2 & 38 & 39 \\ 23 & 19 & 0 & 15 & 21 & 19 & 20 \\ 8 & 4 & 15 & 0 & 6 & 34 & 35 \\ 2 & 2 & 21 & 6 & 0 & 40 & 41 \\ 42 & 38 & 19 & 34 & 40 & 0 & 1 \\ 43 & 39 & 20 & 35 & 41 & 1 & 0 \end{bmatrix}$$

Figure 3: Complete Graph Marix

Generate a cycle of length 7 using the complete graph **K7** shown in Figure 2

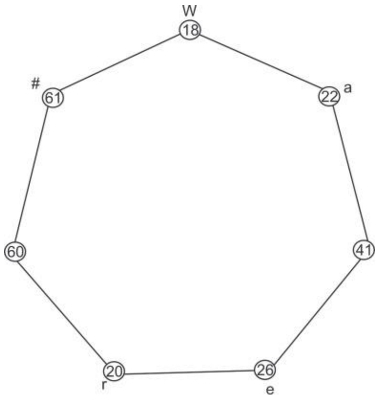

Figure 4: Cycle of Length 7

Create a matrix 'B' based on the cycle. The cycle matrix is constructed in the same way as the complete graph matrix, as shown in Figure 4.

$$B = \begin{bmatrix} 0 & 4 & 0 & 0 & 0 & 0 & 43 \\ 4 & 0 & 19 & 0 & 0 & 0 & 0 \\ 0 & 19 & 0 & 15 & 0 & 0 & 0 \\ 0 & 0 & 15 & 0 & 6 & 0 & 0 \\ 0 & 0 & 0 & 6 & 0 & 40 & 0 \\ 0 & 0 & 0 & 0 & 40 & 0 & 1 \\ 43 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Figure 5: cycle matrix

Modify the diagonal entries in matrix 'B' by replacing the 0's with the newly assigned numeric values corresponding to the characters in the original message, as per the Alphabet Encoding Table (Table 2). The alphabet positions for the characters are: W=23,a=27,t=46,e=31,Y=25,@=65,#=66. Once these values are applied to the diagonal of matrix 'B', we get an updated matrix 'B*' (the revised cycle matrix) as shown below:

$$B^* = \begin{bmatrix} 18 & 4 & 0 & 0 & 0 & 0 & 43 \\ 4 & 22 & 19 & 0 & 0 & 0 & 0 \\ 0 & 19 & 41 & 15 & 0 & 0 & 0 \\ 0 & 0 & 15 & 26 & 6 & 0 & 0 \\ 0 & 0 & 0 & 6 & 20 & 40 & 0 \\ 0 & 0 & 0 & 0 & 40 & 60 & 1 \\ 43 & 0 & 0 & 0 & 0 & 1 & 61 \end{bmatrix}$$

Obtain a new matrix 'N' as follows:

$$N = A \times B^* = \begin{bmatrix} 0 & 4 & 23 & 8 & 2 & 42 & 43 \\ 4 & 0 & 19 & 4 & 2 & 38 & 39 \\ 23 & 19 & 0 & 15 & 21 & 19 & 20 \\ 8 & 4 & 15 & 0 & 6 & 34 & 35 \\ 2 & 2 & 21 & 6 & 0 & 40 & 41 \\ 42 & 38 & 19 & 34 & 40 & 0 & 1 \\ 43 & 39 & 20 & 35 & 41 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 18 & 4 & 0 & 0 & 0 & 0 & 43 \\ 4 & 22 & 19 & 0 & 0 & 0 & 0 \\ 0 & 19 & 41 & 15 & 0 & 0 & 0 \\ 0 & 0 & 15 & 26 & 6 & 0 & 0 \\ 0 & 0 & 0 & 6 & 20 & 40 & 0 \\ 0 & 0 & 0 & 0 & 40 & 60 & 1 \\ 43 & 0 & 0 & 0 & 0 & 1 & 61 \end{bmatrix}$$

$$\mathrm{N} = \begin{bmatrix} 1865 & 525 & 1139 & 565 & 1768 & 2643 & 2665 \\ 1749 & 377 & 839 & 401 & 1584 & 2399 & 2589 \\ 1350 & 510 & 586 & 516 & 1270 & 2000 & 2228 \\ 1665 & 405 & 691 & 261 & 1480 & 2315 & 2513 \\ 1807 & 451 & 989 & 471 & 1636 & 2441 & 2627 \\ 951 & 1365 & 2011 & 1409 & 1004 & 1601 & 1867 \\ 930 & 1410 & 2086 & 1456 & 1070 & 1700 & 1850 \end{bmatrix}$$

Now apply mod 73 0n every element of N to get new Ciper Matrix.

$$N_1 = \begin{bmatrix} 40 & 14 & 44 & 54 & 16 & 15 & 37 \\ 70 & 12 & 36 & 36 & 51 & 63 & 34 \\ 36 & 72 & 2 & 5 & 29 & 29 & 38 \\ 59 & 40 & 34 & 42 & 20 & 52 & 31 \\ 55 & 13 & 40 & 33 & 30 & 32 & 72 \\ 2 & 51 & 40 & 22 & 55 & 68 & 42 \\ 54 & 23 & 42 & 69 & 48 & 21 & 25 \end{bmatrix}$$

The Divison matrix holds the results obtained by dividing each value in N by 73, reflecting the quotients of these divisions.

$$D = \begin{bmatrix} 25 & 7 & 15 & 7 & 24 & 36 & 36 \\ 23 & 5 & 11 & 5 & 21 & 32 & 35 \\ 18 & 6 & 8 & 7 & 17 & 27 & 30 \\ 22 & 5 & 9 & 3 & 20 & 31 & 34 \\ 24 & 6 & 13 & 6 & 22 & 33 & 35 \\ 13 & 18 & 27 & 19 & 13 & 21 & 25 \\ 12 & 19 & 28 & 19 & 14 & 23 & 25 \end{bmatrix}$$

Given the input text 'WateY@#' the encrypted output will be the elements of N_1, arranged sequentially from left to right.The Ciper text will be:

nNr1POk*Ljjy hj) BEccl6nhpTze2Mngdf) BynV2%p1Wp&vUY Decryption Algorithm:

The input for decryption are as follows: N_1(Final Cipher Text), D(divison Matrix), K=5(XOR key), S(second key) = 73, A(Complete graph Matrix)

Write the Cipher text:

nNr1POk*Ljjy hj) BEccl6nhpTze2Mngdf) BynV2%p1Wp&vUY in the final Cipher Matrix N_1form as:

$$N_1 = \begin{bmatrix} 40 & 14 & 44 & 54 & 16 & 15 & 37 \\ 70 & 12 & 36 & 36 & 51 & 63 & 34 \\ 36 & 72 & 2 & 5 & 29 & 29 & 38 \\ 59 & 40 & 34 & 42 & 20 & 52 & 31 \\ 55 & 13 & 40 & 33 & 30 & 32 & 72 \\ 2 & 51 & 40 & 22 & 55 & 68 & 42 \\ 54 & 23 & 42 & 69 & 48 & 21 & 25 \end{bmatrix}$$

Obtain the First Cipher Matrix N with the use of second Key 'S'=73 as follows:

$[D]_{ij}\times 73+[N_1]_{ij}=[N]_{ij}$ where $[D]_{ij}$, $[N_1]_{ij}$ and $[N]_{ij}$ are entries of matrices D, N_1 and N respectively at i^{th} row and j^{th} column. For instance, 25 × 73+ 40 = 1865

$$\mathrm{N} = \begin{bmatrix} 1865 & 525 & 1139 & 565 & 1768 & 2643 & 2665 \\ 1749 & 377 & 839 & 401 & 1584 & 2399 & 2589 \\ 1350 & 510 & 586 & 516 & 1270 & 2000 & 2228 \\ 1665 & 405 & 691 & 261 & 1480 & 2315 & 2513 \\ 1807 & 451 & 989 & 471 & 1636 & 2441 & 2627 \\ 951 & 1365 & 2011 & 1409 & 1004 & 1601 & 1867 \\ 930 & 1410 & 2086 & 1456 & 1070 & 1700 & 1850 \end{bmatrix}$$

Now Compute inverse matrix of A and find B^* with help of $B^*=A^{-1}\times N$

$$B^* = \begin{bmatrix} 18 & 4 & 0 & 0 & 0 & 0 & 43 \\ 4 & 22 & 19 & 0 & 0 & 0 & 0 \\ 0 & 19 & 41 & 15 & 0 & 0 & 0 \\ 0 & 0 & 15 & 26 & 6 & 0 & 0 \\ 0 & 0 & 0 & 6 & 20 & 40 & 0 \\ 0 & 0 & 0 & 0 & 40 & 60 & 1 \\ 43 & 0 & 0 & 0 & 0 & 1 & 61 \end{bmatrix}$$

Now we got diagonal entries of matrix B* as 18,22,41,26,20,60,61 and aply XOR inverse key we get the vlues as: 23,27,46,31,25,65,66 which are when decoded with the help of Table 2(Alphabet Encoding Table), we get 23 = W, 27= a, 46 = t, 31 = e, 25=Y, 65=@, 66=#. Hence the original text is: WateY@#.

Conclusion

This paper introduces a novel encryption and decryption scheme using a complete graph structure and matrix operations. By converting the message into graph vertices, applying XOR operations, and calculating edge labels, the method encrypts the message into a complex cipher text. Decryption involves reversing the process through matrix transformations and using the inverse key to retrieve the original message. The approach highlights the power of graph theory and matrix algebra in cryptography, offering a secure and efficient method for data protection.

References

1. Etaiwi W. M. A., Encryption Algorithm using Graph Theory. Journal of Scientific Research and Reports. 3(19) (2014) 2519-2527.
2. GURJAR D. and Krishna A., Complete Graph and Hamiltonian Cycle in Encryption and Decryption, vol. 67, issue 12 (2021) 62-71.
3. 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 Science and Technology, Palu, Indonesia. Published under license by IOP Publishing Ltd (2020).
4. Krishnaa A., An Example Usage of Graph Theory in Other Scientific Fields: On Graph Labeling, Possibilities and Role of Mind/Consciousness, Chapter in the book titled Graph Theory: Advanced Algorithms and Applications, IntechOpen, London, UK (2018).
5. Krishnaa A., Certain specific graphs in cryptography, Advances and Applications in Discrete Mathematics, 26(2) (2021) 157-177.
6. Krishnaa A., Inner magic and inner antimagic graphs in cryptography Journal of Discrete Mathematical Sciences and Cryptography, 22(6) (2019) 1057-1066.
7. Krishnaa A., Some Applications of Labelled Graphs, International Journal of Mathematics Trends and Technology, 37(3) (2016).
8. 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.
9. Perera P.A.S. and Wijesiri G.S., Encryption and decryption in symmetric key cryptography using graph theory, (2021).
10. Shamir Adi, Random graphs in cryptography, The Weizman Institute, Israel, The Onassis Foundation Science Lecture Series, 28 (2010).
11. Ustimenko V.A., on graph based cryptography and symbolic computations, Serdica Journal of Computing I, (2007) 131-156.
12. Yamuna M. and Karthika K., Data transfer using bipartite Graphs. International Journal of Advance Research in Science and Engineering (IJARSE), 4(2) (2015).