A MAJOR PROJECT II REPORT
ON

Application of Graph Theory in Cryptography

Submitted in Partial Fulfillment of the Requirements
for the Degree of

MASTER OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

Mr. SHUBHAM MERAVI
(Roll No: 2K22/CSE/23)

Under the Supervision of

Dr. Manoj Kumar
(Professor, Department of Computer Science and Engineering)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)

Shahbad Daulatpur, Main Bawana Road, Delhi-110042

MAY, 2024

ACKNOWLEDGEMENT

I would like to express my deep appreciation to **Dr. Manoj Kumar**, Professor at the Department of Computer Science and Engineering, Delhi Technological University, for his invaluable guidance and unwavering encouragement throughout this research. His vast knowledge, motivation, expertise, and insightful feedback have been instrumental in every aspect of preparing this research plan.

I am also grateful to **Dr. Vinod Kumar**, Head of the Department of Computer Science and E Engineering, for his valuable insights, suggestions, and meticulous evaluation of my research work. His expertise and scholarly guidance have significantly enhanced the quality of this major project II.

My heartfelt thanks go out to the esteemed faculty members of the Department of Computer Science & Engineering at Delhi Technological University. I extend my gratitude to my colleagues and friends for their unwavering support and encouragement during this challenging journey. Their intellectual exchanges, constructive critiques, and camaraderie have enriched my research experience and made it truly fulfilling.

While it is impossible to name everyone individually, I want to acknowledge the collective efforts and contributions of all those who have been part of this journey. Their constant love, encouragement, and support have been indispensable in completing this MTech major project II.

Shubham Meravi

(2K22/CSE/03)

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

CANDIDATE DECLARATION

I Shubham Meravi, Roll No: 2K22/CSE/23, student of Master of Technology (Department of Computer Science and Engineering) hereby certify that the work being presented in the major project II entitled “**Application of Graph Theory in Cryptography”** in partial fulfillment of the requirements for the award of the Degree of Master of Technology submitted in the Department of Computer Science and Engineering, Delhi Technological University in an authentic record of my work carried out during the period from August 2022 to May 2024 under the supervision of Dr. Manoj Kumar.

The matter presented in the major project II has not been submitted by me for the award of any other degree of this or any other Institute.

Shubham Meravi

This is to certify that the student has incorporated all the corrections suggested by the examiner in the major project II and that the statement made by the candidate is correct to the best of our knowledge.

Signature of Supervisor Signature of External Examiner

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

CERTIFICATE BY THE SUPERVISOR

Certified that Shubham Meravi, (Roll No: 2K22/CSE/23), Department of Computer Science and Engineering has carried out his research work presented in this major project II entitled "**Application of Graph Theory in Cryptography"** for the award of **Master of Technology** from the Department of Computer Science and Engineering, Delhi Technological University, Delhi under my supervision. The major project II embodies results of original work, and studies are carried out by the student himself and the contents of the major project II do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Dr. Manoj Kumar
(Supervisor)
Professor,
Department of CSE,
DTU-Delhi, India

ABSTRACT

Now, encryption has become very crucial in this world, especially when important information is exchanged, because data is the new oil of this era. This means that we require a new, non-standard, well-secured encryption algorithm to be made. Using the theory of graphs, the encryption concept will comprise meaningful safety features, thereby eliminating standard sniff data. Proposed Algorithm: This paper introduces a new encryption algorithm that uses the characteristics of graph theory in a secure fashion to encrypt and decrypt data. Cryptography presents the most fundamental pillar of graph theory, loaded with a rich set of tools and concepts directed at supporting the security system of communication. The result is that complex relationships within cryptographic systems may be modeled well with graphs because of data stream representation, network topology, and key exchange mechanisms. In such cryptographic uses of graph theory, one finds applications in protocols for scrambling an algorithm and obtaining new algorithms for encryption. The development of encryption schemes is among the most popular applications to which graph theory has been put for the purpose of increasing security through cryptography. Graph-theoretic structures and relationships provide the cryptographic algorithms with a means to encrypt plaintext securely, effectively, and efficiently into ciphertext. These additional properties of the graphs, such as the connectivity of the vertices and edge-disjoint paths, are used to assure better data confidentiality and integrity of the data under encryption through graph-based encryption techniques. Additionally, graph theory provides one of the important tools that can be applied in the key management and distribution process of the cryptographic system. Key exchange protocols, like the Diffie-Hellman key exchange, use a graph-based approach for secure communication channel establishment with the help of secure exchange of cryptographic keys between two parties. Graph theory also forms the basis of vulnerability analysis in networks and the design of secure communication networks by modeling network configurations and identifying possible security risks. In other words, the applications of graph theory to cryptography are driven by the need to fortify the security of digital communication through strong solutions that secure sensitive information and reduce cyber threats in a more connected world.

Keywords: Cryptography; symmetric key; graph algorithms; network security; public key.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 A Brief Overview

Cryptography, or cryptology, the practice and study of techniques for secure communication and third parties called adversaries. In actuality, it works on designing and analyzing protocols which thwart three parties or people from listening in or modifying the data being transferred between two people employing diverse aspects of information security and security services most probably including confidentiality, authenticity, and integrity. An instance is the one called "secure communication," where both parties send certain data or a message, and for this, no information must come out that is being supplied to the adversary.

In cryptography, an adversary is a malicious party that wants to extract sensitive data or information by breaking the neighbor principle of information security. More importantly, these four basic building blocks of today's cryptographies non-repudiation, authenticity, integrity of data, and confidentiality.

• Confidentiality: These are collections of standards and guidelines typically embraced by privacy agreements ensuring the report is available only to a minimal group of participants in person or at certain locations.

• Data Integrity: Processes to maintain the data and assure that it stays accurate and consistent at all levels of its life.

• Authentication is confirming whether the data is of the owner or not.

• Non-repudiation: which is the means of providing proof (to a third party) that someone involved in a transaction or communication cannot deny their identity, signature on a document or message having been sent.

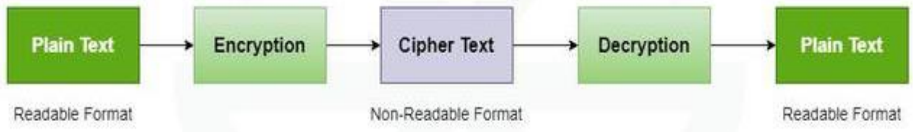


Figure 1.1: Basic Flowchart

1.2 Types of Cryptography System

1.2.1 Symmetric Key Cryptography

Symmetric key uses an encryption system of sending key and receiving key, which is also known as the shared secret key. This would mean that the message must have been sent by one key and also be decrypted by that same single key. Although symmetric key cryptography is easy to use and faster, both the receiving and sending ends need to accept a single secret key in a manner such that it never leaks or should not be known. The most widely used symmetric key cryptography systems are DES and AES.

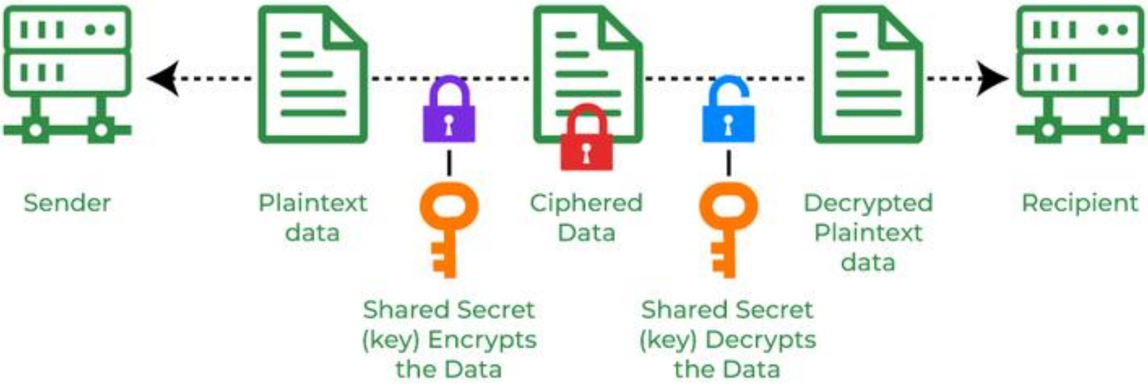

Sender
Plaintext data
Ciphered Data
Decrypted Plaintext data
Recipient
Shared Secret (key) Encrypts the Data
Shared Secret (key) Decrypts the Data

Figure 1.2: Symmetric Key Cryptography

1.2.2 Hash Functions

This rule set does not use any codes. It is hard to get the words from the secret message because a set-size code is made from the words. Codes are broadly used in computer systems to hide codes.

1.2.3 Asymmetric Key Cryptography

Asymmetric key cryptography involves a pair of keys to encrypt and decrypt information. The public key of the recipient is used for encryption and the private key of the receiver is used for decryption. Private and public keys are not the same. Only the intended recipient knows his private key therefore even if the public key is known to all he will be the only one who will be able to decode it. The most used asymmetric key encryption algorithm is the RSA algorithm.

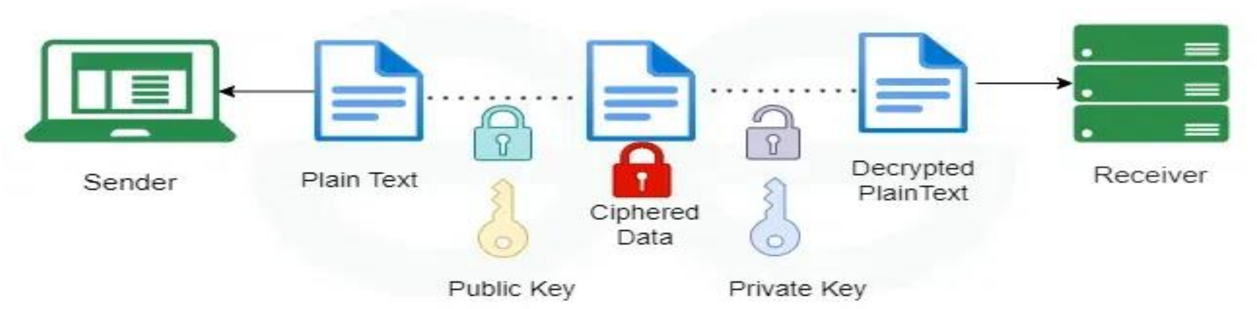


Figure 1.3: Asymmetric Key Cryptography

1.2.4 Applications of Cryptography

- **Passwords on computers:**

  Normally, passwords on computers are established and kept through cryptography for security measures. When a user checks in, his or her password is hashed and compared to previously saved hashes. Before getting kept, passwords are encrypted and hashed. In that manner, the technique applied has the effect that passwords are encrypted so well that a hacker cannot read them, even if they somehow get access to the password database.

- **Secure web browsing:**

  Sniffing and man-in-the-middle attacks are protected by cryptography. SSL and TLS execute public-key cryptography in the course of browsing to encrypt data in transit between a web server and a user, thus creating a secure line of communication.

- **E-signatures:**

  E-signatures are the digital equivalents of handwritten signatures in the digital world. Public key cryptography can be used in order to verify electronic signatures. Already in most countries, electronic signatures have been legally recognized under the information technology acts. Very rapidly, their use is growing..

- **Cryptocurrencies:**

  A huge amount of cryptography is used in the operations of both cryptocurrencies: Bitcoin and Ethereum. The cryptography secures, authenticates, and, in a manner of speaking, keeps intact the nature of transactions in a way that even the most powerful algorithms and cryptographic keys must fail to tamper with the transactions.

- **End-to-end Internet Encryption:**

  It provides a two-way communication channel in use for emails, instant messages, and video chats. Specifically, it provides mechanisms to make sure that the message will be read by no one else except the desired recipients, even with encryption applied. In general, end-to-end encryption is used with strong communication apps, such as WhatsApp and Signal.

1.2.5 Advantages of Cryptography

- **Access Control:**

  Cryptography-based access control can ensure that no person accesses a resource other than personnel who are authorized to do so. At the most primitive level, the idea is to ensure that the person who wants to access the resource is himself the one having the appropriate key to decrypt it.

- **Secure Communication:**

  This technology is very important in the secure communication of the Internet because it gives very secure methods to send over the Internet sensitive information such as account numbers, bank account numbers, and their passwords.

- **Attack protection:**

  Cryptography provides functionalities that can prevent several types of attacks, including replays and man-in-the-middle assaults. It advises on how such threats can be identified and mitigated.

- **Legal compliance:**

  It is in this respect that firms can apply cryptography to ensure compliance with various laws considering privacy, and even data security.

1.3 Graph

Vertices, or nodes, and edges, or lines or arcs joining any two nodes, make up a graph, a non linear data structure. A graph is formalized as the set of edges (E) and vertices (V), represented as G(V, E).

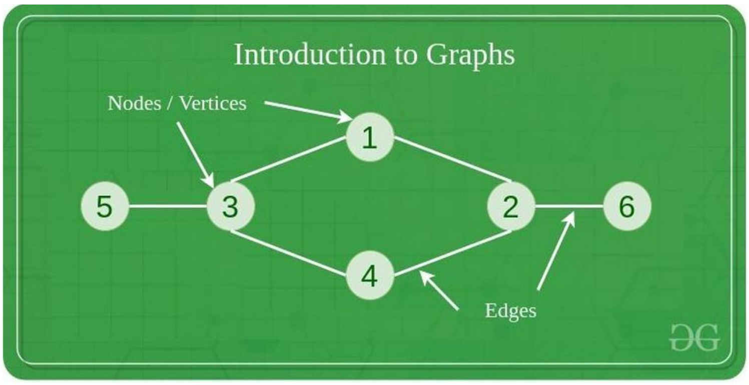


Figure: 1.4: Introduction to Graphs

Components of a Graph:

- Vertices serve as the foundational units of a graph and are alternatively referred to as vertex or nodes. Each node/vertex can possess labels or remain unlabelled.
- Edges, on the other hand, establish connections between two nodes within the graph. In a directed graph, edges are represented as ordered pairs of nodes. They can connect nodes in various ways without specific constraints. Edges are occasionally termed arcs, and, like vertices, each edge may have labels or remain unlabelled.

1.3.1 Minimum Spanning Tree

A spanning tree of a connected, undirected graph is a tree-shaped subgraph; that is, it is a tree that involves all the nodes of the graph. Minimum spanning tree inherits the properties of a spanning tree and has to ensure its weights are least among all of its possible spanning trees. Feasible minimum spanning trees exist more than one can be formed in a graph and generally look like spanning trees.

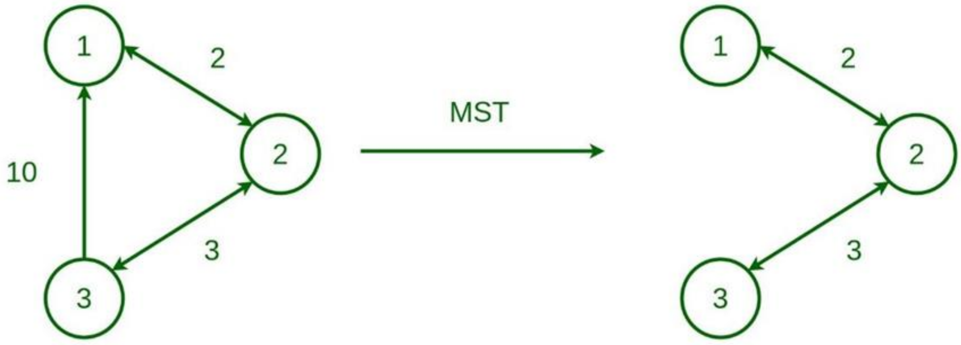


Figure 1.5: Minimum Spanning Tree for Directed Graph

Properties of a Spanning Tree:

The following principles characterize a spanning tree:

- Both spanning tree and graph matrices have the same total vertices (V).
- The spanning tree has a fixed number of edges, one less than there are in the total count, of vertices: E = V - 1.
- There will be no cycle in the spanning tree. It must be connected since it will be put to use to deliver a single source of components without any portion being unconnected.
- The total cost (or weight) of that tree is the sum of the weights of all the edges of the spanning tree.

1.3.2 Prim’s Algorithm

Prim's Algorithm is a greedy technique of finding a minimum spanning tree in a graph. It would find a subset of the edges in the graph where every vertex is contained, such that the total cost of all these edges is minimized.

The algorithm of Prim works by observing all the neighboring nodes and connections to them starting from one node and one iteration at a time. It chooses the least weighted edges so that there is no cycle in the graph.

The Prim's algorithm is going to be greedy in nature. It starts with one vertex and incrementally adds the single attached edges if they have the least weight until the objective is achieved. The steps to follow in the Prim's algorithm are now written below:

- Select an arbitrary initial vertex to be used as the base of operations for the Minimum Spanning Tree (MST).
- Repeat steps 3 to 5 for such vertices until there exists no vertices left not inserted into the MST. Those are termed as the fringe vertices.
- Find all edges connecting any vertex in the current tree to the fringe vertices.
- Determine the edge of minimum weight present in the set of candidate edges.
- Include the chosen edge in the MST if this process will not create a cycle.
- Return Minimum Spanning Tree and return.

Prim's algorithm finds applications in various scenarios:

- Network Design: Prim's algorithm is applicable in the design of networks.
- Network Cycle Formation: It can be utilized for creating cycles within a network.
- Electrical Wiring Layout: The algorithm is employed in the layout and planning of electrical wiring cables.

Illustration of Prim's Algorithm:

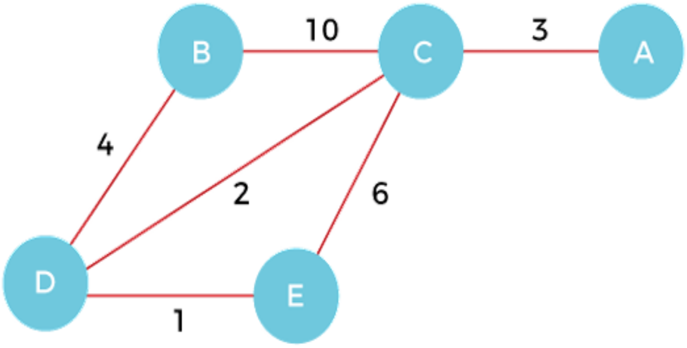


Figure 1.6: Directed Graph

STEP 1: To commence, we opt for a starting vertex from the provided graph; in this instance, let's select vertex B.

Figure 1.7: Step 1

STEP 2: Next, we find and include the shortest edge that is adjacent to vertex B. Examining all other edges, BC of length 10 and BD of length 4, We pick edge BD because it is of low weight so that we then add it to Minimum Spanning Tree (MST).

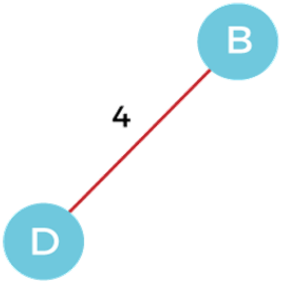


Figure 1.8: Step 2

STEP 3: The edges with the lowest weights among the remaining choices are then iteratively chosen. In this instance, we add edges DE and CD to the MST and investigate C's neighboring vertices, A and E. As a result, we add edge DE to the MST after selecting it.

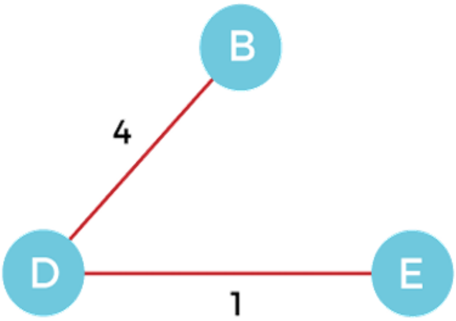


Figure 1.9: Step 3

STEP 4: Following this method, we select edge CD and incorporate it into the MST. The next option is edge CA because choosing edge CE would result in a cycle in the graph. As a result, we augment the MST with edge CA.

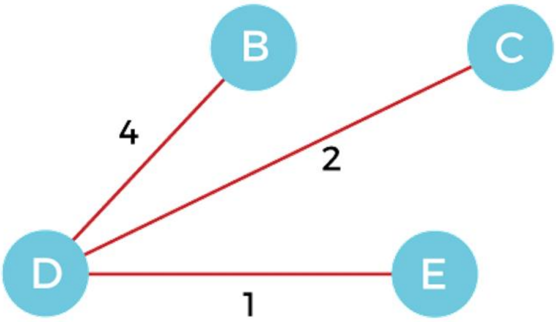
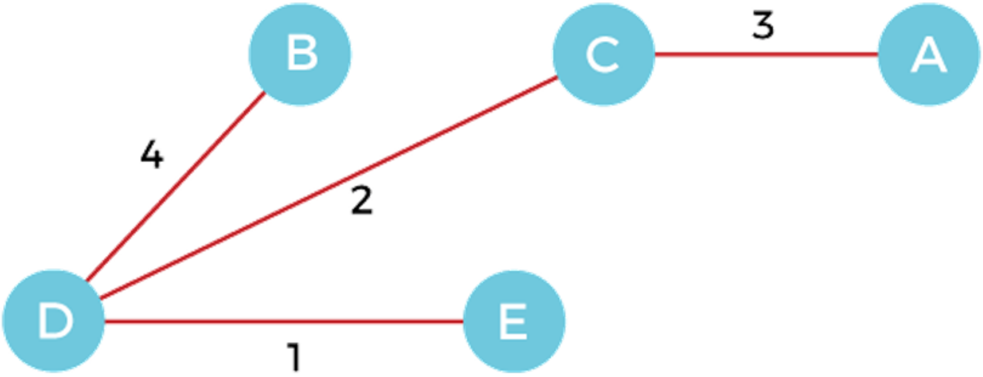


Figure 1.10: Step 4

STEP 5: After doing this all the series of operations, we get a graph in which the Minimum Spanning Tree is very clearly visible. The sum of the weights of the edges contained in an MST is often called the cost of that MST.

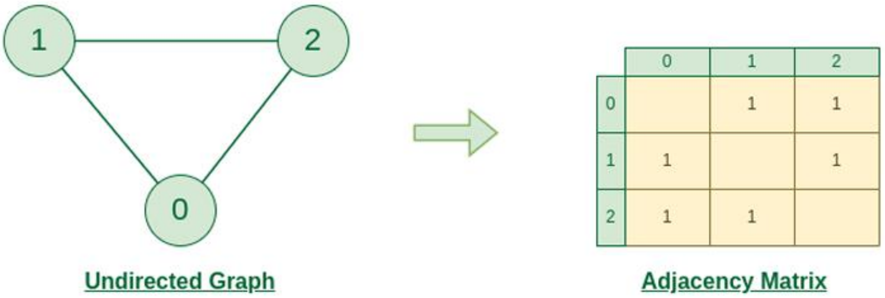


Step 1.11: Step 5

MST is equal to 4 + 2 + 1 + 3 units.

1.3.3 Adjacency Matrix

An adjacency matrix is explicitly defined as an N by N square matrix, where N is the number of nodes contained in a graph. It indicates what links and relationships exist in a graph's edges.



	0	1	2
0		1	1
1	1		1
2	1	1	

Figure 1.12: Graph Representation of Undirected graph to adjacency matrix

The following characteristics of an adjacency matrix :

1. The size of the matrix is related to the number of vertices in the graph.
2. The matrix has dimensions determined by the number of nodes.
3. One can count the number of edges in a graph easily.
4. If the graph has few edges, then the derived matrix is poorly populated.

1.4 Project Objective

The aim of the project is to Investigate the utilization of graph theory concepts such as connectivity, paths, cycles, and flows in designing secure encryption algorithms, Developing graph-based encryption schemes that can efficiently protect sensitive data and ensure confidentiality and integrity during transmission, Exploring graph-based key management protocols for secure key exchange and distribution in cryptographic systems, Analyzing the use of graph theory in identifying vulnerabilities in communication networks and designing secure communication protocols, Evaluating the performance and effectiveness of graph based cryptographic techniques in comparison to traditional cryptographic methods, Demonstrating the practical implementation of graph-based cryptographic algorithms in real world scenarios to showcase their applicability and benefits, and By achieving these objectives, the project aims to contribute to the advancement of cryptography by harnessing the power of graph theory to strengthen data security, enable secure communication, and mitigate cybersecurity threats. The work will further explore the applicability of graph-based cryptography to practical communication network security, and graph-based as well as some other possible solutions to boost the overall resiliency of cryptographic systems. Main project: in general, it is the whole number of types of research and experimentations to evaluate performance, scalability, and effectiveness of cryptographic graph-based techniques in real-world scenarios, extracting their potentiality to provide reasonable data protection and security communication channels within the digital environment. More concretely, the major project II leads to a contribution within the general area of cryptography in which advantages gained by using graph theory as an auxiliary tool ensure data confidentiality, integrity, and authenticity in cryptographic systems.

CHAPTER 2
LITERATURE SURVEY

2.1 Overview

The literature review explains the borderline approach of graph theory in regard to cryptographic systems. It is clearly meant to try to explore the diverse applications and implications a graph-based approach might just have in the field of improving data security and communication privacy. This study intends to cover a wide-ranging study and consideration of potential ways to introduce the principles of graph theory into the field of cryptographic science. Key areas of interest for the literature review include the development of graph-based encryption algorithms, key distribution protocols, and network security mechanisms required to fortify cryptographic systems against cyber threats. Consequently, the research will open up strengths, weaknesses, and potential challenges associated with the innovative approach by analyzing the theoretical bases in use and practical implementations of graph-based cryptography. The literature survey will also span the efficacy with which graph-based techniques enforce data privacy, ensure integrity of messages, and manage to preserve secure communication channels. It then further examines the systemic way that graph theory gives in the identification of vulnerabilities in the communication networks, optimization key exchange processes, and enhancement resilience of the overall system. Therefore, a full-scale literature review on graph-based cryptography must help provide insights and highlight the emerging trends for the development of the new directions of study both theoretically and in terms of practice in this dynamic field. This survey would actually help illustrate new openings for further research work and innovations emanating from the insights derived from the same. Finally, the literature review could contribute to advancements within cryptographic practices by revealing advantages and implications arising from embedding graph theory principles into modern security solutions.

2.2 Related Work

Graph Theory Matrix Approach in Cryptography and Network Security *by Geetha N K and Ragavi V,2022, Algorithms, Computing, and Mathematics Conference (ACM).*

The paper discusses how the Graph Theory Matrix Approach (GTMA) has been integrated into computer design, information technology, and cryptography. It is shown how the approach is adopted for data structures, communication networks, security, and, more recently, relating to representation of cyber events and data for secure communication related works. This paper will begin by detailing the importance of cryptography in secure communication, expatiating on how TLS encryption can be used to attain a high level of strong authentication and confidentiality in the transition from HTTP to HTTPS. It is an explanation of what a digital signature is as a means of electronic identification and authentication employed in many online transactions and government processes.

Furthermore, the paper delves into end-to-end encryption with an indication of how PGP (Pretty Good Privacy) could work as a model to secure data and store it while in social networks and messaging applications. The authors go on to argue that data integrity and data confidentiality could come through if an encryption procedure is adhered to; for instance, storage encryption, both at rest and in transit. The paper is written with the detailed encryption process via GTMA, which shows how a message is turned into a graph that is then encrypted using a public key. It gives a procedure for encoding step by step such that ,making the spanning tree and then encrypting the message. The decryption process has also been provided showing the way through which the actual message can be obtained via the inverse image of the common key. Conclusion: The paper concludes that good cryptographic systems ought to have the property of yielding significantly different cipher texts for even small changes in plain text or keys. It gives reference of few research works on graph based cryptography, and also refer to the use of matrix operations for the algorithms to be efficient.

An Approach of Graph Theory on Cryptography *by Indhu K and Rekha S, published in the International Journal for Research in Applied Science & Engineering Technology (IJRASET) in November 2022.*

The authors start off by presenting the general perspective of the notion and term cryptography. In fact, it concerns the practice of securing information from being understood in a readable format and, on the contrary, transforming it into an unreadable format, called ciphertext; the other way around goes under the name of plain text, and all of that is made by the practices of encryption and decryption. They propose a way of encoding plaintext into the form of nodes of a graph, where one character is to be transformed into one vertex, and adjacent characters into adjacent vertices. They generate a cycle graph containing all pairs of characters and label the edges by an encoding table, which specifies exactly the respective numerical values given to each letter. Step-by-Step View of Encoding

Process The process begins by creating a weighted graph where the labels of each edge represent the distance between the encoding values of the two connected characters. Edges are included one by one until a fully connected graph is constructed. Lastly, one special character is added to the matrix in addition to the maximum weight in the encoding table, which represents the starting message. A fully constructed graph is shown in matrix form. The encryption process forms a minimal spanning tree from the graph and stores the arrangement in the diagonal on the matrix character. The resulting product matrix then multiplies with another matrix, and the generated encrypted data sends to the recipient. This will reach decryption for the receiver by multiplying the received cipher text with the inverse of the shared key matrix. It will then extract the original graph from the original matrix and solve the plain text message through an encoding table. The finalization of this paper includes acknowledgments for financial support and a schema of references. The authors encrypt messages with the help of the approach mixed with graph theory and linear algebra within the text to make communication secure.

Cryptography – A Graph Theory Approach *by Uma Dixit, 2017, International Journal of Advance Research in Science and Engineering.*

The Author describes graph theory in crypto, showing its uses in network security, coding theory, and communication networks. The paper discusses the categorization of cryptographic protocols into private-key (symmetric) and public-key (asymmetric) crypto, including how keys are used during encryption and decryption. It also establishes the principle of Kerckhoff as one of the basic rules supporting any modern cryptographic algorithm and defines the implication of the secrecy of the key for the secrecy of the algorithm. That is, connections of modern cryptology with discrete mathematics, primarily with graph theory, are discussed in great length to show how the study of graph theory has been incorporated in the making of newer and stronger cryptographic algorithms. This paper presents an application that demonstrates how cryptographic methods intertwine with graph theory.Basically, in this application, text or data is said to be converted into a graph, wherein each character is treated as a vertex and the adjacent characters of the underlying text are connected in this graph as its adjacent vertices. The process of selective encryption with a message-dependent key and the concept of spanning trees in graph theory are nicely covered. In this work, now the intended method of encryption is a graph to be transformed to weighted graph, to generate a complete graph and also the minimal spanning tree. The sample depicted here step by step tells about the processes of encryption, modification of distance matrix, encryption using public key and decryption operation through inverse of shared key. The bibliography included at the end of this paper also comprises of leading works in cryptography and graph theory. Overall, the paper is a good effort oriented toward the in-depth analysis of the relationship between graph theory and cryptography at large, along with applications in developing methods of encryption. The application described concisely illustrates how concepts of graph theory can be applied in combination with cryptographic methods and opens new areas for the application of graph theory in increasing security and efficiency in cryptographic algorithms.

A Graph Theory Approach in Cryptography *by Nandhini R, Maheshwari V and Balaji V, Journal of Computation Mathematica, 2018*

It starts with definitions of weighted graphs and cycle graphs, further what a spanning tree is, and then its importance in graph theory. Then they proceed to how such ideas would actually be used in encryption: for example setting text as a graph, making a complete general graph, and then presenting it in matrix form. A minimal spanning tree can be constructed in a weighted graph, but the simple process of how a minimal spanning tree can be constructed has been described below. This leads to the process of encryption using a public key and generation of cipher text. A decrypting process is also shown and one can see that the source text is being retrieved from the decrypted graph. In addition, as part of the message-encrypted text example, "HATE" is recognized for financial support given and accompanied with references for further reading on a variety of related topics. This paper presents an introduction to graph theory and its application, particularly in the field of cryptography for encryption and decryption. It shows how one can convert the plaintext into a graph by defining every one character to be one vertex and making linked up characters adjacent according to a cycle graph. Finally, it explains this encoding table, along with how to label the graph according to the distance, according to the table. It also details how the weighted graph is drawn and filled with weights sequentially increased. After which it goes on into the extensive detail of the process to create a minimal span tree from the weighted graph, after which is the procedure for encryption with matrix operations and a public key, and thereafter reads to establish how this multiplication can be decrypted using the inverse of the shared key. Now, with this document, a decrypted text example and financial support are concluded, and a boast of references where another interested in schemes of cryptography based on graphs can look for gold. In other words, the document gives elaborate research over the application of graph theory in cryptography and the full description for encryption and decryption. It provides step-by-step guidelines on how to transform the text into a graph and develop a minimal spanning tree, and how to use matrix operations for encryption and decryption. Further enhanced with a practical example and an extensive list of references for follow-up by any interested party in graph theory and cryptography.

CHAPTER 3
METHODOLOGY

This section will contain detailed descriptions, flowcharts, and illustrative examples of the algorithm that explains how the proposed cryptographic method works in this innovative approach.

3.1 PROPOSED ALGORITHM

The characters in the message are shown as vertices in a graph once it has been retrieved from the user. A cyclic graph is created by adding each vertex to the graph in turn. After that, each edge in the cycle that is produced by applying a specific scheme—which will be covered in a moment—is given an edge weight. After that, weights are assigned to each of the remaining edges, or the cycle's diagonals. We add vertex and edge with a special character (let's say "A") that points to the message's initial character. We now have a weighted undirected graph that is complete. The generated complete graph's Minimum Spanning Tree (MST) computation comes next. An adjacency matrix represents an MST. The character order of the original message can be obtained from the diagonal of the MST. Now, the adjacency matrix of complete graph will multiply with the adjacency matrix of the MST. Now multiply the resultant matrix again with the key matrix but in matrix form. At last the result which seems in the final resultant matrix will be converted into the cipher text that will be sent to the receiver.

Scheme for assigning weights to the edges:

- Start with the character that has been used to point to the 1st character of the input string (‘A’ in our case). Calculate the difference between the ASCII values of the 1st character of the input string and this pointer character and assign this weight to the edge between these two vertices by adding one.
- If the edge is a side of the polygon(cycle) or between the special character and the starting character, then for computing the edge weights, ASCII values of the characters are considered and the edge weight of any such edge is computed by calculating the difference between the ASCII values of the characters and then adding one to it. Calculate this difference for each edge cyclically. (See the example part for more info).
- The reason for adding one in the above method is to avoid the edge weight of two same characters from becoming zero. And if the edge weight becomes zero, then while

computing the MST the program will consider that no edge lies between these vertices which is a wrong assumption.

- If the edge is a diagonal of the polygon, then edge weights are sequentially assigned starting from 256 and increasing the weight by 1 each time.

3.2 Cipher Algorithm:

- Replace the initial character with a special character; designate this special character as A.
- Meanwhile create a vertex for every character in the plain text.
- Join the vertices, connecting them with an edge between each two consecutive characters in the plaintext so that we have a cycle graph.
- At this stage the graph would take the form of a cycle along with an edge sticking out.
- Assign weights to each edge according to the above scheme.
- Add additional diagonals until a complete graph M2 is constructed, where every newly inserted diagonal is given a consecutive weight starting from 256.
- Next, the Minimum Spanning Tree is found to be: M2.
- Later, save the vertices in a proper order of sequence in the diagonal places of the M2 matrix
- Now multiply the matrices M1 by M2 to receive M3
- Then we compute the product of M3 with prespecified Shared-Key K to obtain Cipher.
- The message to be encoded after encryption is performed is the Cipher matrix and the M1 matrix.

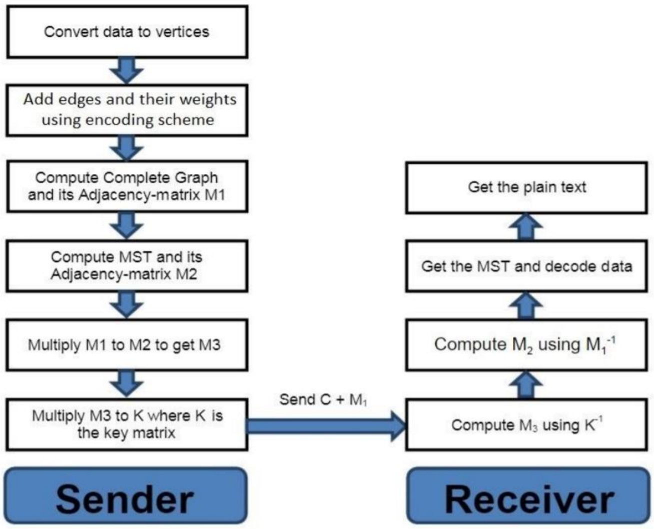

Convert data to vertices
Add edges and their weights using encoding scheme
Compute Complete Graph and its Adjacency-matrix M1
Compute MST and its Adjacency-matrix M2
Multiply M1 to M2 to get M3
Multiply M3 to K where K is the key matrix
Send C + M_1
Get the plain text
Get the MST and decode data
Compute M_2 using M_1^{-1}
Compute M_3 using K^{-1}
Sender
Receiver

Figure 3.1: Flowchart

3.3 Decipher Algorithm:

• The recipient receives the cipher and M1 matrix one by one from the sender. Receiver Side

• The receiver derives M3 as the matrix multiplication of C with the inverse form of the Shared-Secret: –1K.

• Now Calculate M2 by product of the inverse form of M1; that is, by M1-1 and the matrix M3.

• Now, the receiver end computed MST matrix, M2.

• Then find the original text by decoding M2 with the help of the encoding scheme.

• The original text will then be shown on the receiver side.

3.4 Example

Let’s encrypt the message “WAEL”.

INPUT STRING = “WAEL”

ENCODING:

Step 1: Turn the message into a graph by turning each letter into a vertex.

Convert the message into a graph by converting each letter into a vertex. (Fig 1)

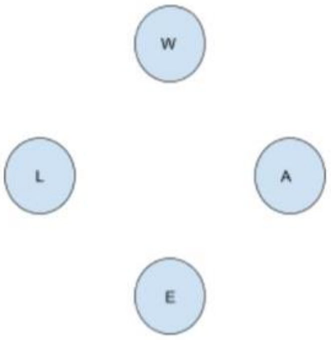


Figure 3.2: Step 1

Step 2: Adding edges to form a complete cycle of the input.

Now connect each pair of characters with a cycle graph. This will give the graph the shape of a cycle polygon. (See Fig 2)

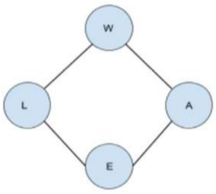


Figure 3.3: Step 2

Step 3: Add weights to each of the edges of the cycle.

Add weights for each of the edges using the weighting scheme discussed in the algorithm section (Using the difference of ASCII values of adjacent vertices- See Fig 3)

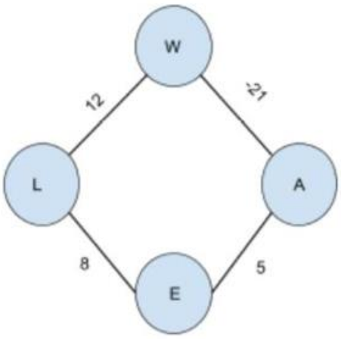


Figure 3.4: Step 3

Step 4: Add weights to the diagonals to make the complete graph.

After which add edges some more to make it a complete graph. All the diagonals can be connected. Now there remains the diagonals that we had added recently to connect which will have a consecutive weight starting from a maximum weight of 256. The weights of the diagonals now will be 256, 257, 258…(See Fig 4)

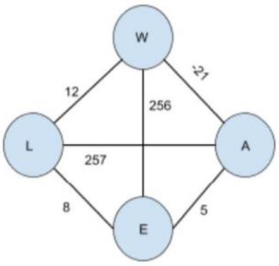


Figure 3.5: Step 4

Step 5: Add a special character for the starting symbol,.

Now, add an extra vertex label “A” to the first character of the message and then compute the edge weight of this newly created edge using the encoding scheme. (See Fig 5) The complete adjacency matrix, M1, is shown in Fig 7.

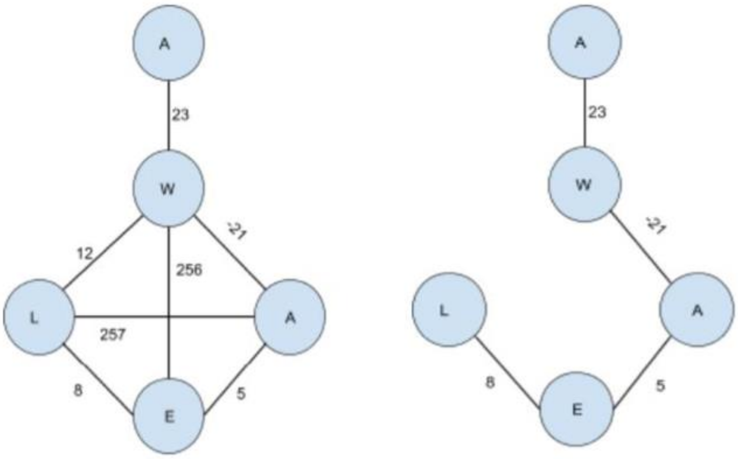
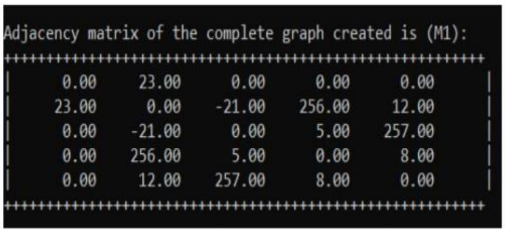


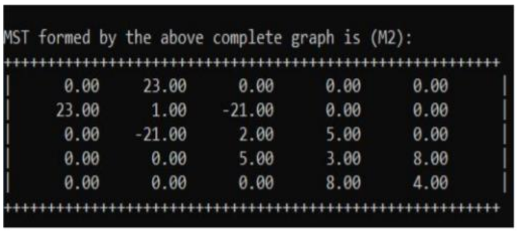
Figure 3.6:Step 5, 6

Step 6: Determine the MST of the graph and the associated adjacency matrix M2

Now that the complete graph is constructed, find the MST by means of Prim's Algorithm. The adjacency matrix of this MST, M2. is as follows in Fig 8. The adjacency matrix of the complete graph constructed is (M1).



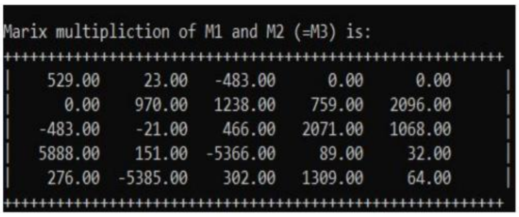
```
Adjacency matrix of the complete graph created is (M1):
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
|      0.00     23.00      0.00      0.00      0.00      |
|     23.00      0.00    -21.00    256.00     12.00      |
|      0.00    -21.00      0.00      5.00    257.00      |
|      0.00    256.00      5.00      0.00      8.00      |
|      0.00     12.00    257.00      8.00      0.00      |
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```



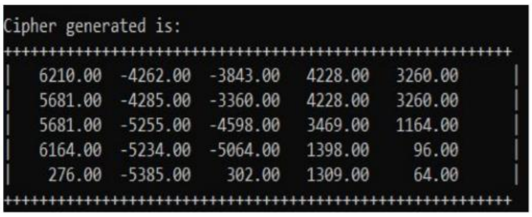
```
MST formed by the above complete graph is (M2):
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
|      0.00     23.00      0.00      0.00      0.00      |
|     23.00      1.00    -21.00      0.00      0.00      |
|      0.00    -21.00      2.00      5.00      0.00      |
|      0.00      0.00      5.00      3.00      8.00      |
|      0.00      0.00      0.00      8.00      4.00      |
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Figure 3.7: Matrix M1

Figure 3.8 : Matrix M2



```
Marix multipliction of M1 and M2 (=M3) is:
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
|    529.00     23.00   -483.00      0.00      0.00      |
|      0.00    970.00   1238.00    759.00   2096.00      |
|   -483.00    -21.00    466.00   2071.00   1068.00      |
|   5888.00    151.00  -5366.00     89.00     32.00      |
|    276.00  -5385.00    302.00   1309.00     64.00      |
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```



```
Cipher generated is:
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
|   6210.00  -4262.00  -3843.00   4228.00   3260.00      |
|   5681.00  -4285.00  -3360.00   4228.00   3260.00      |
|   5681.00  -5255.00  -4598.00   3469.00   1164.00      |
|   6164.00  -5234.00  -5064.00   1398.00     96.00      |
|    276.00  -5385.00    302.00   1309.00     64.00      |
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Figure 3.9: Matrix M3

Figure 3.10: Cipher Matrix

Step 7: Computing Matrix M3

We calculate matrix M3 by matrix multiplication of M1 and M2 . The matrix M3 for “WAEL” is shown in Fig.9 above.

M3 = M1 * M2

Step 8: Computing Cipher Matrix

This is the final step in encrypting the input message. Here, the Cipher matrix is generated by matrix multiplication of the pre-defined Key and M3. Now the Cipher and M1 are ready to be sent to the receiver. Fig.10 above shows the Cipher for our input message.

Cipher = Key * M3

Send (Cipher + M1) to Receiver.

DECODING

Step 1: Receive the data, compute the inverse of the Key

The receiver gets the encoded message and separates the message to retrieve the Cipher matrix (Fig 10) and the M1 matrix (Fig 7). The receiver node then calculates the inverse of the pre-shared Key. The Key is shown in Fig 11 and the Key -1 (Key- inverse) is shown in Fig 12.

Step 2: Compute M3 using Key -1 and Cipher

Now, the matrix M3 is reconstructed using the key-inverse (Key -1) matrix. M3 is calculated by the matrix multiplication of Key -1 and Cipher matrices.

M3 = Key -1 * Cipher

Step 3: Compute M2 using M1 -1 and M3

Next, Matrix M2 is reconstructed using the inverse of Matrix M1. M2 is calculated by the matrix multiplication of M1 -1 and M3 matrices.

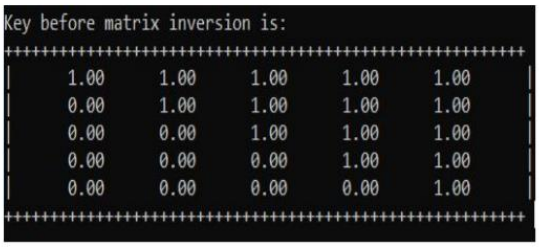
M2 = M1 -1 * M3

After the calculation of Matrix M2, each value in the double-data type M2 matrix has to be converted to the nearest integer by rounding off. Matrix M1 -1 is shown in Fig 13 and Matrix M2 after rounding off to the nearest integer is shown in Fig 14.

Step 4: Get the MST and decode the data

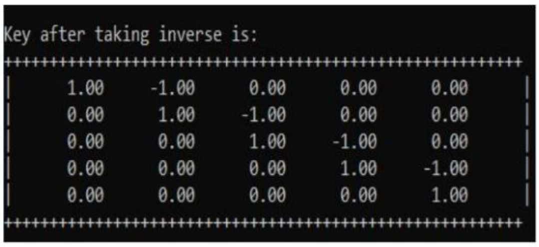
The matrix M2 (Fig 14) itself represents the MST. After the MST has been extracted, the original message is decoded from it. To get the original message back, first we traverse through the right part of the MST (Fig 6) starting from the top and keep adding the weights every time and find the ASCII character corresponding to the added weight at each edge. Thus a new character is found by adding the weight at each node and this character is concatenated to the end of the reconstructed string.

However, this method of decoding the data from MST (Step 4 in decoding) will not work for all messages. This method worked for the message “WAEL” because the MST of this graph is cyclic (excluding the special character- See Fig 6). However, in some messages, the computed MST may not be entirely cyclic. The MST can also be branched. In that case, we need to change our Step 4 a little bit. In example 2 Step 4 of decoding, has been extended further to cover all cases.



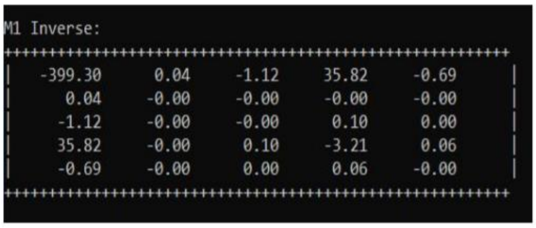
```
Key before matrix inversion is:
+++++++++++++++++++++++++++++++++++++++++++++++++++++
|       1.00       1.00       1.00       1.00       1.00      |
|       0.00       1.00       1.00       1.00       1.00      |
|       0.00       0.00       1.00       1.00       1.00      |
|       0.00       0.00       0.00       1.00       1.00      |
|       0.00       0.00       0.00       0.00       1.00      |
+++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Figure 3.11: Shared Key



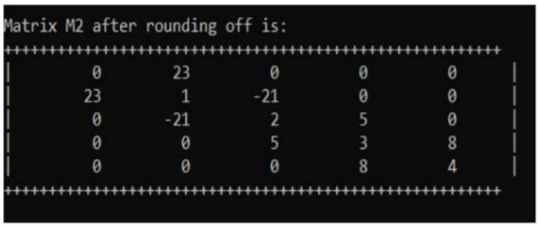
```
Key after taking inverse is:
+++++++++++++++++++++++++++++++++++++++++++++++++++++
|       1.00      -1.00       0.00       0.00       0.00      |
|       0.00       1.00      -1.00       0.00       0.00      |
|       0.00       0.00       1.00      -1.00       0.00      |
|       0.00       0.00       0.00       1.00      -1.00      |
|       0.00       0.00       0.00       0.00       1.00      |
+++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Figure 3.12: Key Inverse



```
M1 Inverse:
+++++++++++++++++++++++++++++++++++++++++++++++++++++
|   -399.30       0.04      -1.12      35.82      -0.69      |
|      0.04      -0.00      -0.00      -0.00      -0.00      |
|     -1.12      -0.00      -0.00       0.10       0.00      |
|     35.82      -0.00       0.10      -3.21       0.06      |
|     -0.69      -0.00       0.00       0.06      -0.00      |
+++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Figure 3.13 : M1 Inverse



```
Matrix M2 after rounding off is:
+++++++++++++++++++++++++++++++++++++++++++++++++++++
|          0         23          0          0          0      |
|         23          1        -21          0          0      |
|          0        -21          2          5          0      |
|          0          0          5          3          8      |
|          0          0          0          8          4      |
+++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Figure 3.14: M2 after rounding off

EXAMPLE 2

In the previous example, the MST corresponding to the constructed graph of the word “WAEL” is cyclic (Fig 6). So, Step 4 of decoding (Get the MST and decode data) was quite simple. But in many messages, the generated MST can be branched. This can be visualized by using the algorithm for the word “HELLO”. Fig 15- 20 show the various steps involved in extracting the MST corresponding to the message “HELLO”. We can see that the generated MST (Fig 20) is actually branched (neglecting the 1st special character). So, the final decoding process has to be extended to cover all such cases. In this case, first, traverse through the right branch of the MST and follow the normal process of adding the weights at each node and getting the corresponding. Then if an edge is absent (denoted by 0 in the adjacency matrix), stop the traversing. Come back to the first character (H in this case) and this time traverse through the left branch and every time subtract the weight from the ASCII value of the 1st character (H). The character corresponding to this new ASCII value has to be concatenated to a new string. This is done till we reach the end of the left branch. Finally, reverse this string and concatenate this string with the string generated during traversal of the right branch. This will give the final decoded message.

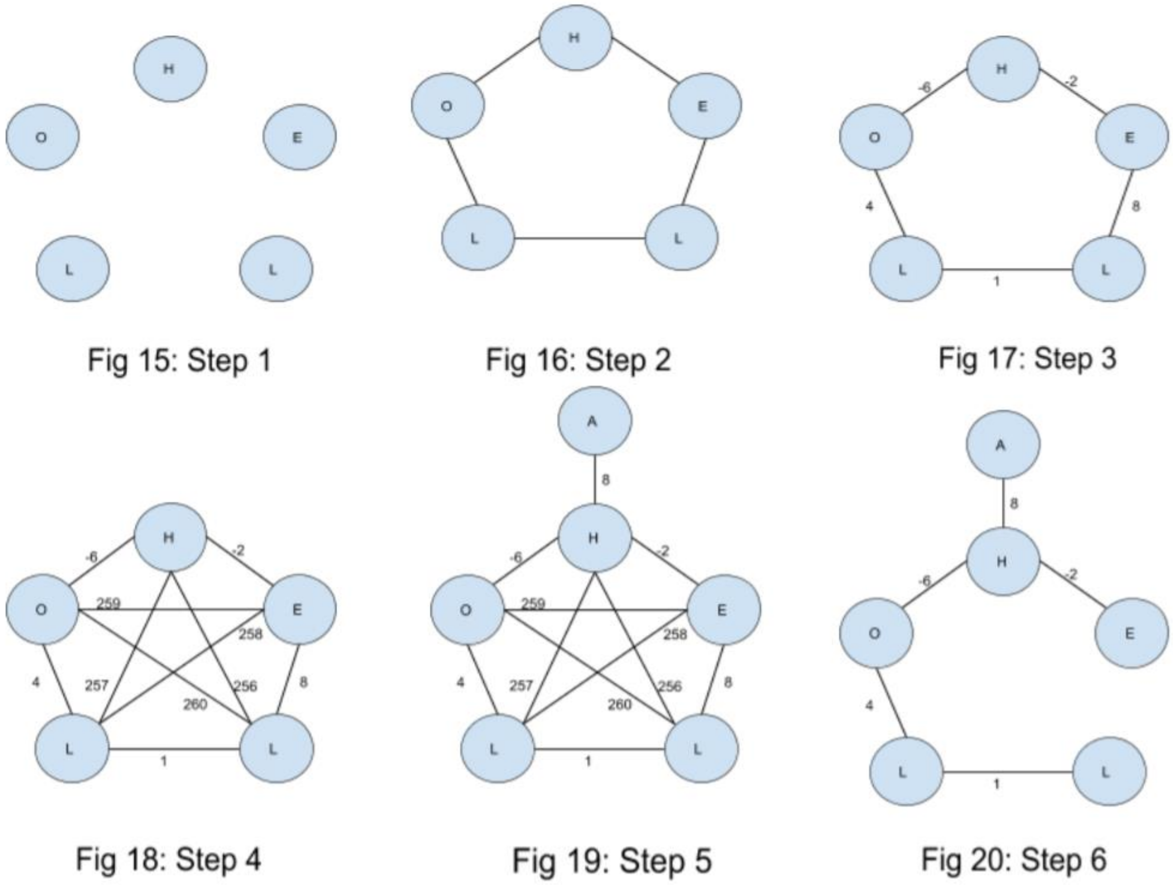


Fig 15: Step 1

Fig 16: Step 2

Fig 17: Step 3

Fig 18: Step 4

Fig 19: Step 5

Fig 20: Step 6

Figure 3.15: "HELLO" diagram

Message Decoded in trip 1: String 1 = "HE"

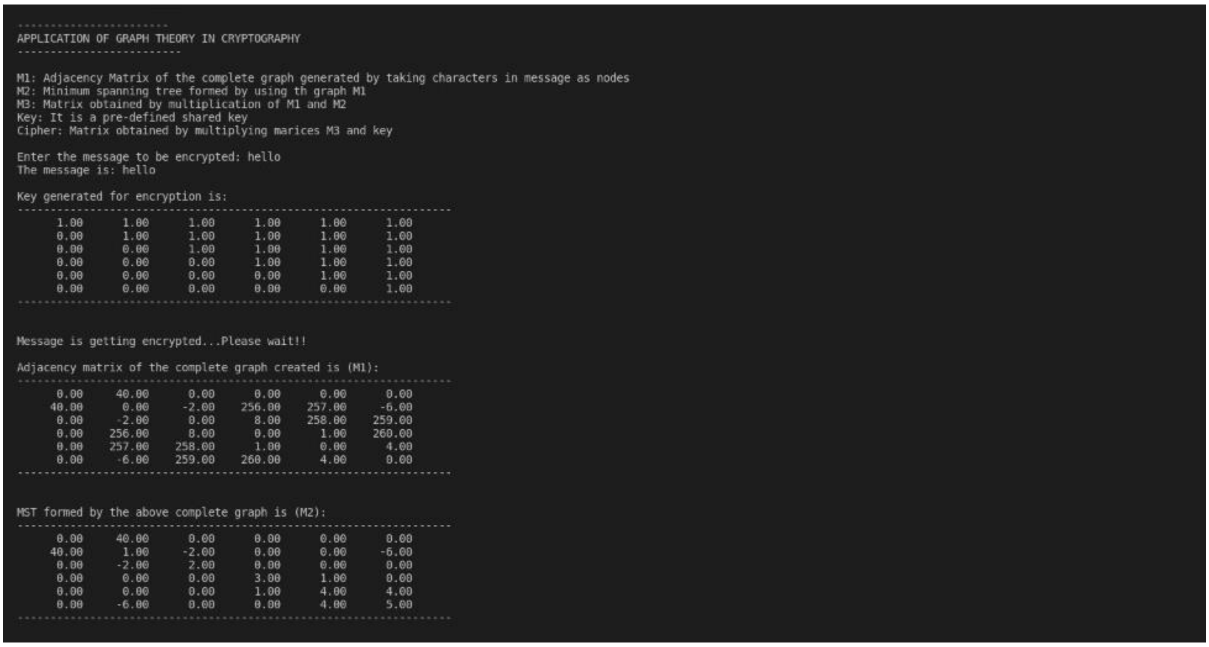
Message Decoded in trip 2: String 2 = "OLL"

Final Decode message: String 1 + reverse(String 2) HELLO

CHAPTER 4

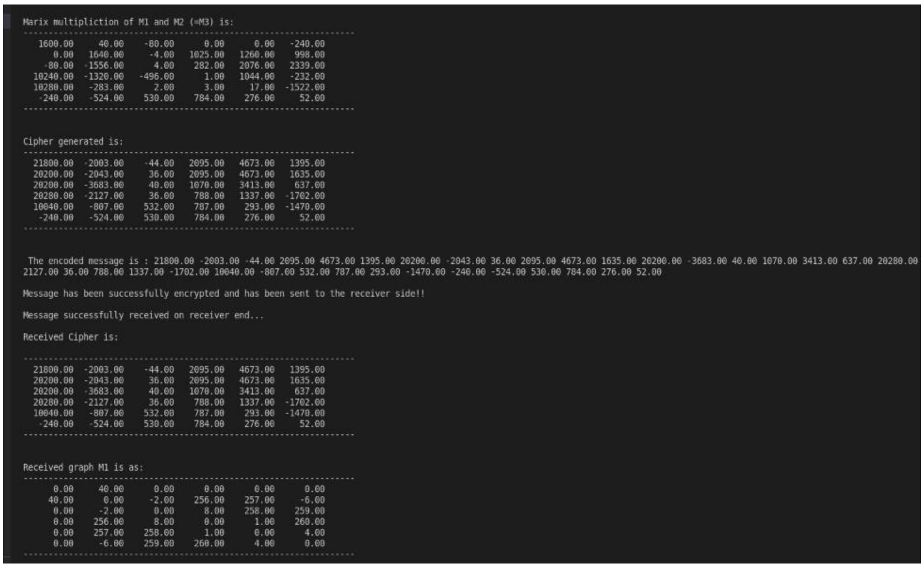
RESULTS

This proposed algorithm was implemented by us using C++ programming language. Our implementation takes the message as input. First the program encodes the input message using the encoding algorithm. Then this encoded message is passed on to the decoding module of our program. The decoding module decodes the message and finally displays the decoded message. The snapshots when the word “HELLO” was given as input message to our implementation are shown below:



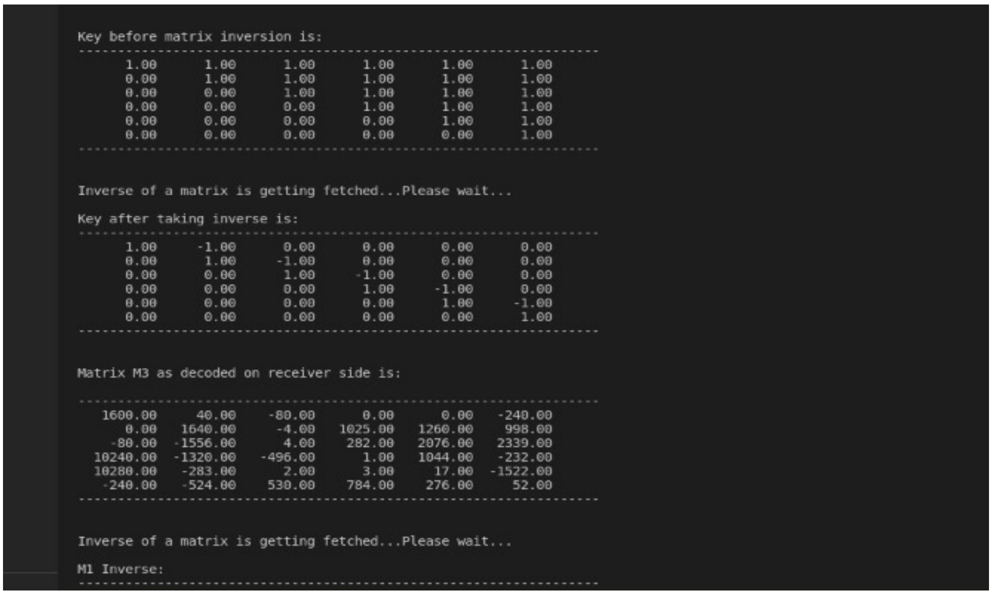
```
-----------------------
APPLICATION OF GRAPH THEORY IN CRYPTOGRAPHY
------------------------

M1: Adjacency Matrix of the complete graph generated by taking characters in message as nodes
M2: Minimum spanning tree formed by using th graph M1
M3: Matrix obtained by multiplication of M1 and M2
Key: It is a pre-defined shared key
Cipher: Matrix obtained by multiplying marices M3 and key

Enter the message to be encrypted: hello
The message is: hello

Key generated for encryption is:
-------------------------------------------------------------------
      1.00      1.00      1.00      1.00      1.00      1.00
      0.00      1.00      1.00      1.00      1.00      1.00
      0.00      0.00      1.00      1.00      1.00      1.00
      0.00      0.00      0.00      1.00      1.00      1.00
      0.00      0.00      0.00      0.00      1.00      1.00
      0.00      0.00      0.00      0.00      0.00      1.00
-------------------------------------------------------------------


Message is getting encrypted...Please wait!!

Adjacency matrix of the complete graph created is (M1):
-------------------------------------------------------------------
      0.00     40.00      0.00      0.00      0.00      0.00
     40.00      0.00     -2.00    256.00    257.00     -6.00
      0.00     -2.00      0.00      8.00    258.00    259.00
      0.00    256.00      8.00      0.00      1.00    260.00
      0.00    257.00    258.00      1.00      0.00      4.00
      0.00     -6.00    259.00    260.00      4.00      0.00
-------------------------------------------------------------------


MST formed by the above complete graph is (M2):
-------------------------------------------------------------------
      0.00     40.00      0.00      0.00      0.00      0.00
     40.00      1.00     -2.00      0.00      0.00     -6.00
      0.00     -2.00      2.00      0.00      0.00      0.00
      0.00      0.00      0.00      3.00      1.00      0.00
      0.00      0.00      0.00      1.00      4.00      4.00
      0.00     -6.00      0.00      0.00      4.00      5.00
-------------------------------------------------------------------
```

Figure 4.1: Output SS 1



```
Marix multipliction of M1 and M2 (=M3) is:
-------------------------------------------------------------------
   1600.00     40.00    -80.00      0.00      0.00   -240.00
      0.00   1640.00     -4.00   1025.00   1260.00    998.00
    -80.00  -1556.00      4.00    282.00   2076.00   2339.00
  10240.00  -1320.00   -496.00      1.00   1044.00   -232.00
  10280.00   -283.00      2.00      3.00     17.00  -1522.00
   -240.00   -524.00    530.00    784.00    276.00     52.00
-------------------------------------------------------------------


Cipher generated is:
-------------------------------------------------------------------
  21800.00  -2003.00    -44.00   2095.00   4673.00   1395.00
  20200.00  -2043.00     36.00   2095.00   4673.00   1635.00
  20200.00  -3683.00     40.00   1070.00   3413.00    637.00
  20280.00  -2127.00     36.00    788.00   1337.00  -1702.00
  10040.00   -807.00    532.00    787.00    293.00  -1470.00
   -240.00   -524.00    530.00    784.00    276.00     52.00
-------------------------------------------------------------------


 The encoded message is : 21800.00 -2003.00 -44.00 2095.00 4673.00 1395.00 20200.00 -2043.00 36.00 2095.00 4673.00 1635.00 20200.00 -3683.00 40.00 1070.00 3413.00 637.00 20280.00
2127.00 36.00 788.00 1337.00 -1702.00 10040.00 -807.00 532.00 787.00 293.00 -1470.00 -240.00 -524.00 530.00 784.00 276.00 52.00

Message has been successfully encrypted and has been sent to the receiver side!!

Message successfully received on receiver end...

Received Cipher is:

-------------------------------------------------------------------
  21800.00  -2003.00    -44.00   2095.00   4673.00   1395.00
  20200.00  -2043.00     36.00   2095.00   4673.00   1635.00
  20200.00  -3683.00     40.00   1070.00   3413.00    637.00
  20280.00  -2127.00     36.00    788.00   1337.00  -1702.00
  10040.00   -807.00    532.00    787.00    293.00  -1470.00
   -240.00   -524.00    530.00    784.00    276.00     52.00
-------------------------------------------------------------------


Received graph M1 is as:
-------------------------------------------------------------------
      0.00     40.00      0.00      0.00      0.00      0.00
     40.00      0.00     -2.00    256.00    257.00     -6.00
      0.00     -2.00      0.00      8.00    258.00    259.00
      0.00    256.00      8.00      0.00      1.00    260.00
      0.00    257.00    258.00      1.00      0.00      4.00
      0.00     -6.00    259.00    260.00      4.00      0.00
-------------------------------------------------------------------
```

Figure 4.2 : Output SS 2



```
Key before matrix inversion is:
------------------------------------------------------------------
      1.00      1.00      1.00      1.00      1.00      1.00
      0.00      1.00      1.00      1.00      1.00      1.00
      0.00      0.00      1.00      1.00      1.00      1.00
      0.00      0.00      0.00      1.00      1.00      1.00
      0.00      0.00      0.00      0.00      1.00      1.00
      0.00      0.00      0.00      0.00      0.00      1.00
------------------------------------------------------------------


Inverse of a matrix is getting fetched...Please wait...

Key after taking inverse is:
------------------------------------------------------------------
      1.00     -1.00      0.00      0.00      0.00      0.00
      0.00      1.00     -1.00      0.00      0.00      0.00
      0.00      0.00      1.00     -1.00      0.00      0.00
      0.00      0.00      0.00      1.00     -1.00      0.00
      0.00      0.00      0.00      0.00      1.00     -1.00
      0.00      0.00      0.00      0.00      0.00      1.00
------------------------------------------------------------------


Matrix M3 as decoded on receiver side is:

------------------------------------------------------------------
   1600.00     40.00    -80.00      0.00      0.00   -240.00
      0.00   1640.00     -4.00   1025.00   1260.00    998.00
    -80.00  -1556.00      4.00    282.00   2076.00   2339.00
  10240.00  -1320.00   -496.00      1.00   1044.00   -232.00
  10280.00   -283.00      2.00      3.00     17.00  -1522.00
   -240.00   -524.00    530.00    784.00    276.00     52.00
------------------------------------------------------------------


Inverse of a matrix is getting fetched...Please wait...

M1 Inverse:
------------------------------------------------------------------
```

Figure 4.2: Output SS 3

CHAPTER 5

CONCLUSION

In this project, I have proposed a technique to encrypt a message using concepts of Graph theory like Minimum Spanning Trees (MST), Prim's algorithm, adjacency matrix etc. Cryptography is necessary in the modern era to send and receive data safely and securely and our proposed algorithm is able to encrypt the messages with high security.

I have used a secure pre-defined key which is shared between the sender and the recipient only. The cipher is then generated and passed on to the receiver along with the graph. Once the recipient receives the necessary data, the MST is re-constructed by the receiver and the message is regenerated by re-tracing the encryption algorithm in the reverse order. The order of the characters in the message is determined by traversing the values in the correct order in the diagonal of the computed MST by the receiver. This algorithm works for almost all types of characters in the range of ASCII and can be safely used to encrypt passwords and secure data of short lengths.

CHAPTER 6

FUTURE WORK

- For messages of greater lengths, the computational time is comparatively higher as time is consumed in calculating the inverse of a matrix which takes O(n 3) time complexity.
- In such cases, the data of greater lengths can be divided into smaller sub-parts and then encrypted. The smaller sub-parts decrypted on the receiver end can then be concatenated to form the original message. This will increase the efficiency of the algorithm to a large extent.
- Further, we can try to optimize our algorithm by reducing the complexity of various modules of our algorithm.

REFERENCES

[1] C. E. Shannon, “Communication theory of secrecy systems∗,” Bell System Technical Journal, vol. 28, no. 4, pp. 656–715, 1949.

[2] K. H. Rosen, Elementary Number theory and its Applications, Addison-Wesley, Boston, MA, USA, 5th edition, 2005.

[3] D. R. Stinson, Cryptography: Theory and Practice, Chapman and Hall/CRC, Boca Raton, FL, USA, 4th edition, 2018.

[4] D. B. West, Introduction to Graph Theory, Pearson, London, UK, 2nd edition, 2001.

[5] R. Frucht and F. Harary, “On the corona of two graphs,” Aequationes Math, vol. 4, pp. 322 325, 1970.

[6] R. L. Rivest, A. Shamir, and L. Adleman, “A method for obtaining digital signatures and public-key cryptosystems,” Communications of the ACM,vol. 21,no. 2, pp.120–126, 1978.

[7] V. A. Ustimenko, “On graph-based cryptography and symbolic computations,” Serdica Journal of Computing, vol. 1, pp. 131–156, 2007.

[8] D. X. Charles, K. E. Lauter, and E. Z. Goren, “Cryptographic hash functions from expander graphs,” Journal of Cryptology, vol. 22, no. 1, pp. 93–113, 2009.

[9] R. Selvakumar and N. Gupta, “Fundamental circuits and cut sets used in cryptography,” Journal of Discrete Mathematical Sciences and Cryptography, vol.15, no. 4-5, pp. 287–301, 2012.

[10]P. Kedia and S. Agrawal, “Encryption using Venn-diagrams and graph,” International Journal of Advanced Computer Technology, vol. 4, no. 01, pp. 94–99, 2015.

[11]M. Yamuna and A. Elakkiya, “Data transfer using fundamental circuits,” International Journal of Computer and Modern Technology, vol. 2, no. 01, 2015.

[12]M. Yamuna and K. Karthika, “Data transfer using bipartite graphs,” International Journal of Advance Research in Science and Engineering, vol. 4, no. 02, pp. 128–131, 2015.

[13]W. Mahmoud and A. Etaiwi, “Encryption algorithm using graph theory,” Journal of Scientific Research and Reports, vol. 3, no. 19, pp. 2519–2527, 2014.

[14]B. R. Arunkumar, “Applications of Bipartite Graph in diverse fields including cloud computing,” International Journal of Modern Engineering Research, vol. 5, no. 7, p. 7, 2015.

[15]D. Sinha and A. Sethi, “Encryption using network and matrices through signed

graphs," International Journal of Computer Applications (0975-8887), vol. 138, no. 4, pp. 6–13, 2016.

[16]Hu, J. Liang, and S. Dong, "A bipartite graph propagation approach for mobile advertising fraud detection," Mobile Information Systems, vol. 2017, p.12, Article ID 6412521, 2017.

[17]Razaq, M. Awais Yousaf, U. Shuaib, N. Siddiqui, A. Ullah, and A. Waheed, "A novel construction of substitution box involving coset diagram and a bijective map," Security and Communication Networks, vol. 2017, p.16, Article ID 5101934, 2017.

[18]Razaq, H. Alolaiyan, M. Ahmad, et al., "A novel method for generation of strong substitution-boxes based on coset graphs and symmetric groups," IEEE Access, vol. 8, pp. 75473–75490, 2020.

[19]G. A. Selim, "How to encrypt a graph," International Journal of Parallel, Emergent and Distributed Systems, vol. 35, no. 6, pp. 668–681, 2020.

[20]110 Nandhini R, Maheswari V and Balaji V (2018) A Graph Theory Approach on Cryptography, Journal of Computational Mathematica, Volume 2, Issue 1, 2018:97-104, DOI:http://doi.org/10.26524/cm32.

[21]Corman TH, Leiserson CE, Rivest RL, Stein C. Introduction to algorithms 2nd edition, McGraw-Hill. K. Elissa, "Title of paper if known," unpublished.

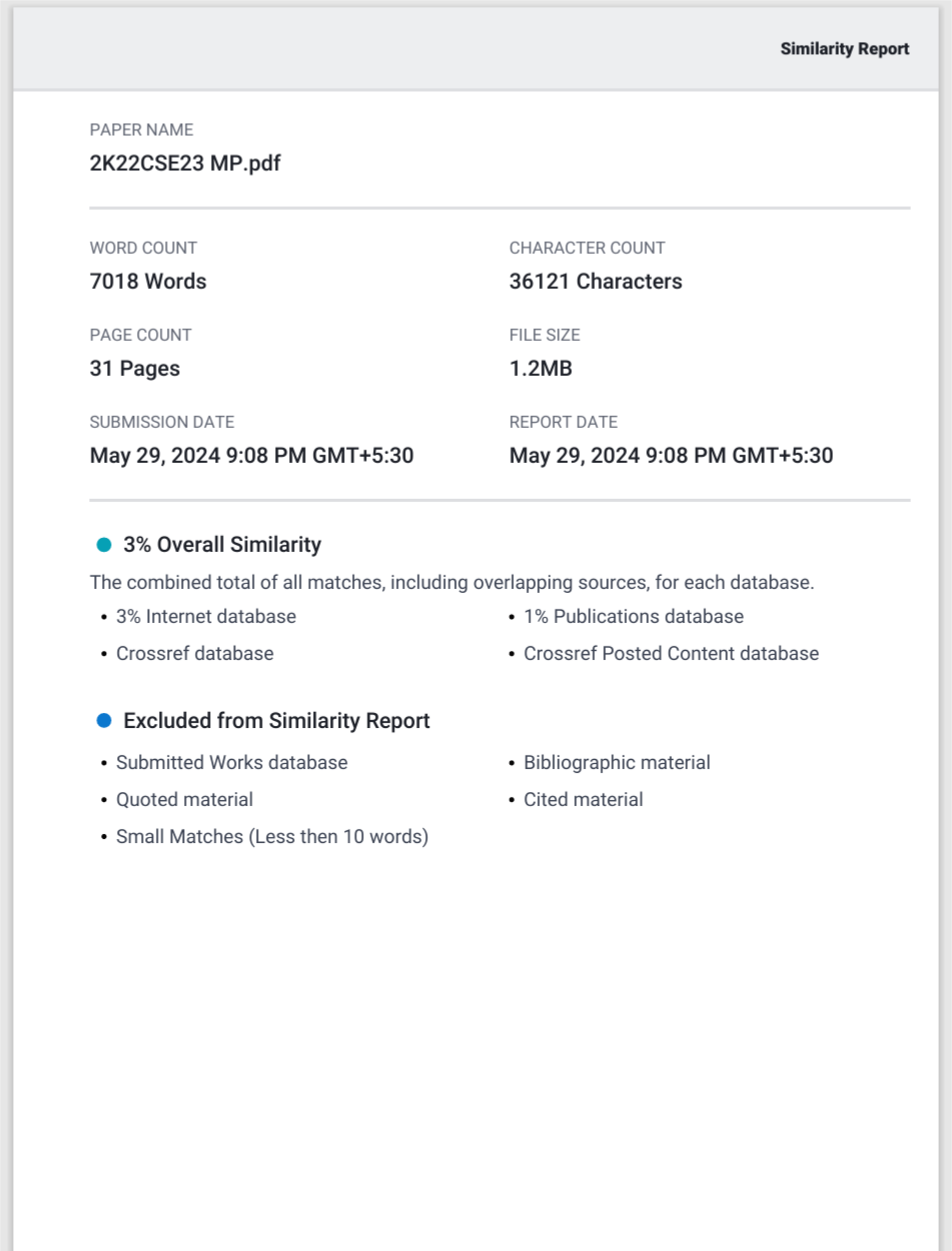
[22]Yamuna M, Meenal Gogia, Ashish Sikka, Md. Jazib Hayat Khan. Encryption using graph theory and linear algebra. International Journal of Computer Application. ISSN:2250- 1797; 2012.

[23]Ustimenko VA. On graph-based cryptography and symbolic computations, Serdica. Journal of Computing. 2007;131-156.

[24]Paszkiewicz A, et al. Proposals of graph-based ciphers, theory, and implementations. Research Gate; 2001.

[25]Steve Lu, Rafail Ostrovsky. Daniel Manchala. Visual Cryptography on Graphs, CiteSeerx, COCOON. 2008;225-234.

[26]P. L. K. Priyadarsini, "A survey on some applications of graph theory in cryptography," Journal of Discrete Mathematical Sciences and Cryptography, vol. 18, no. 3, pp. 209–217, 2015.



Similarity Report

PAPER NAME

2K22CSE23 MP.pdf

WORD COUNT

7018 Words

CHARACTER COUNT

36121 Characters

PAGE COUNT

31 Pages

FILE SIZE

1.2MB

SUBMISSION DATE

May 29, 2024 9:08 PM GMT+5:30

REPORT DATE

May 29, 2024 9:08 PM GMT+5:30

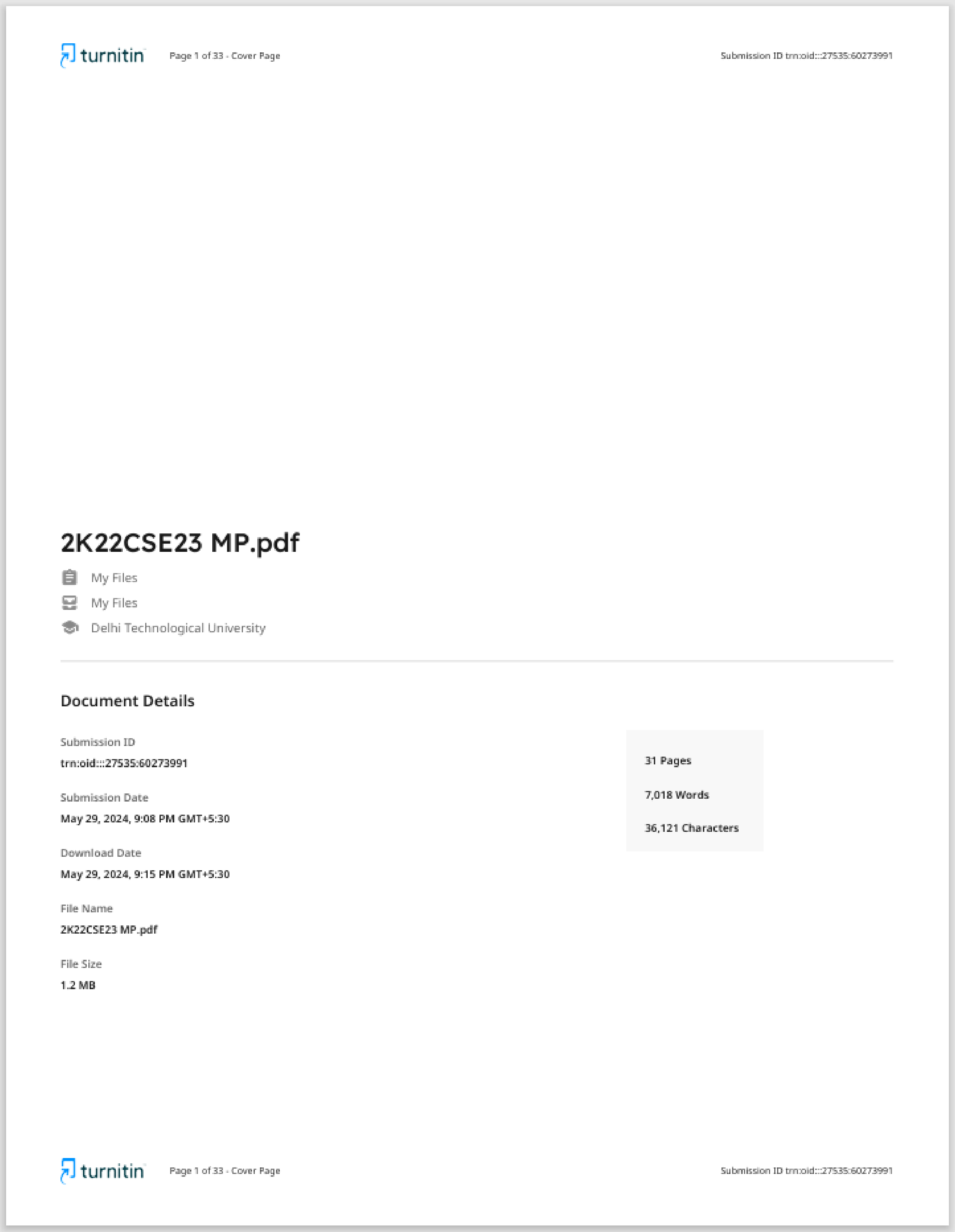
● **3% Overall Similarity**

The combined total of all matches, including overlapping sources, for each database.

- 3% Internet database
- 1% Publications database
- Crossref database
- Crossref Posted Content database

● **Excluded from Similarity Report**

- Submitted Works database
- Bibliographic material
- Quoted material
- Cited material
- Small Matches (Less then 10 words)



turnitin Page 1 of 33 - Cover Page Submission ID trn:oid:::27535:60273991

2K22CSE23 MP.pdf

My Files

My Files

Delhi Technological University

Document Details

Submission ID

trn:oid:::27535:60273991

Submission Date

May 29, 2024, 9:08 PM GMT+5:30

Download Date

May 29, 2024, 9:15 PM GMT+5:30

File Name

2K22CSE23 MP.pdf

File Size

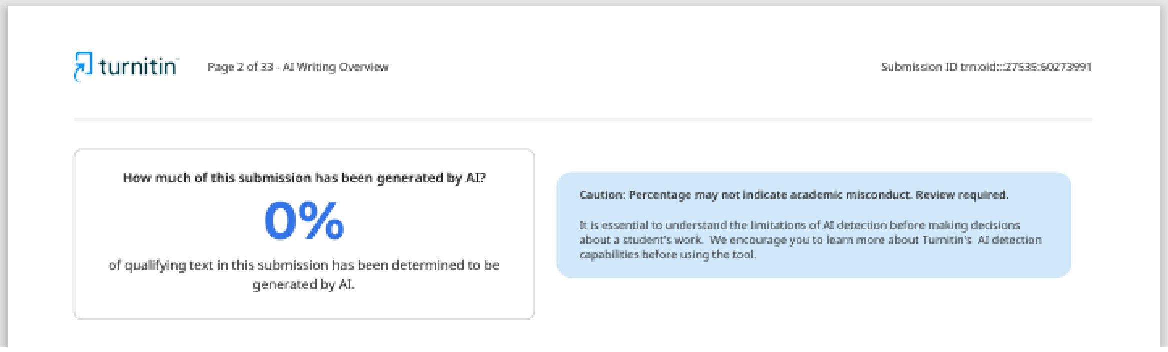
1.2 MB

31 Pages

7,018 Words

36,121 Characters

turnitin Page 1 of 33 - Cover Page Submission ID trn:oid:::27535:60273991



turnitin Page 2 of 33 - AI Writing Overview Submission ID trn:oid:::27535:60273991

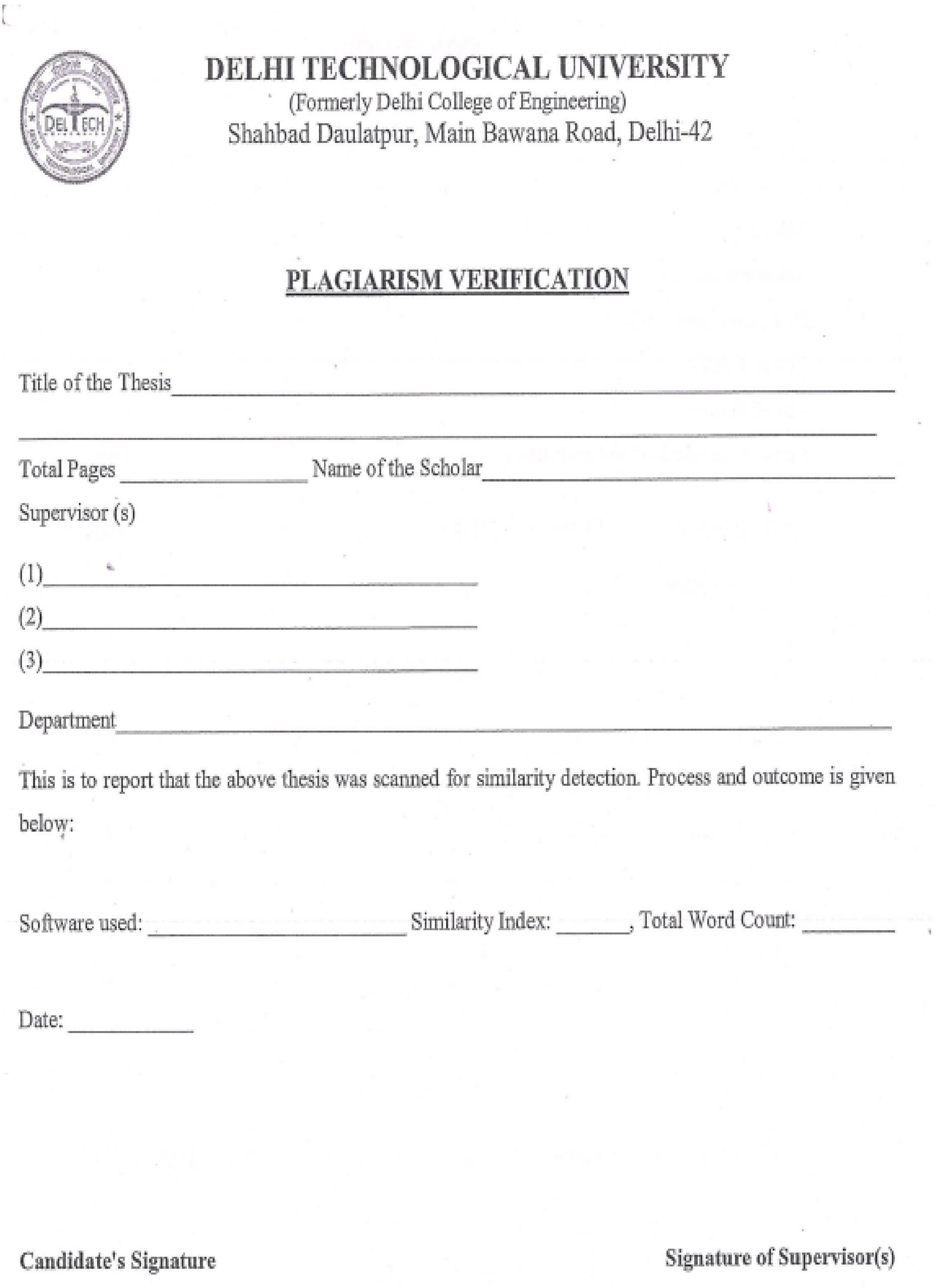
How much of this submission has been generated by AI?

0%

of qualifying text in this submission has been determined to be generated by AI.

Caution: Percentage may not indicate academic misconduct. Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.



DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-42

PLAGIARISM VERIFICATION

Title of the Thesis ____

____

Total Pages ______________ Name of the Scholar ______________________

Supervisor (s)

(1)______________________________

(2)______________________________

(3)______________________________

Department ____

This is to report that the above thesis was scanned for similarity detection. Process and outcome is given below:

Software used: ________________ Similarity Index: ______, Total Word Count: ________

Date: __________

Candidate's Signature

Signature of Supervisor(s)