

**MINOR PROJECT- SYNOPSIS**

**TITLE: SOCIAL NETWORK ANALYSIS**

**( FOR DETECTINGMUTUAL FRIENDS AND SHORTEST PATH BETWEEN USERS)**

**Submitted By:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Sap Id: | Roll No: | Branch |
| Aadya Srivastava | 500105682 | R2142220003 | CSE AI/ML(B-5 Non-Hons) |
| Riya | 500101787 | R2142220617 | CSE AI/ML(B-5 Non-Hons) |
| Manas Arora | 500109627 | R2142221204 | CSE AI/ML(B-5 Non-Hons) |
| Anubhav Anand | 500105017 | R2142220229 | CSE AI/ML(B-5 Non-Hons) |

**Under the guidance of:**

**Dr. Mohammad Ahsan**

**Approved By**:

**Project Guide Cluster Head**

**TABLE OF CONTENTS:**

1. **ABSTRACT**
2. **INTRODUCTION**
3. **LITERATURE REVIEW**
4. **PROBLEM STATEMENT**
5. **OBJECTIVES**
6. **METHODOLOGY**
7. **APPLICATION**
8. **SYSTEM REQUIREMENTS**
9. **RESULT**
10. **CONCLUSION**
11. **REFERENCES**

**School of Computer Science**

University of Petroleum & Energy Studies, Dehradun

**ABSTRACT**

In today's interconnected world, social networks play a pivotal role in shaping relationships, influencing decisions, and fostering communities. As these networks grow increasingly complex, there is a pressing need for tools that can analyze and visualize the intricate web of connections. The "Social Network Analysis" project addresses this need by offering a platform that not only simplifies the exploration of social networks but also provides valuable insights into the underlying structures that govern them. Detecting mutual friends and identifying the shortest paths between users are critical tasks for enhancing user engagement on social media platforms, improving targeted marketing strategies, and understanding social influence patterns. By equipping users with the ability to uncover these connections and paths effortlessly, this project meets the growing demand for accessible and practical tools in the fields of data analysis, digital marketing, and social sciences.

The "Social Network Analysis" project is a Java-based application designed to provide a comprehensive platform for analyzing and visualizing social networks using graph theory algorithms. The project aims to bridge the gap between theoretical graph concepts and practical applications in social network analysis (SNA).

This project focuses on two core functionalities: detecting mutual friends and finding the shortest path between users. The "Detecting Mutual Friends" feature identifies common connections between two users, a common functionality used by social media platforms to suggest new friends or connections. Meanwhile, the "Finding the Shortest Path Between Users" function determines the shortest connection between two individuals in a network. This provides insights into how closely connected users are, even when they are not directly linked.

**INTRODUCTION**

The Social Network Analysis project is a Java-based tool designed to analyze and visualize social networks using graph algorithms like BFS. It enables users to explore connections, detect mutual friends, and identify shortest paths in a network through accessible visual interfaces. This project serves as an educational platform, bridging theoretical graph concepts with practical social network analysis without requiring extensive coding skills.

In this project, "Detecting Mutual Friends and Shortest Path Between Them," we’ll focus on basically two specific tasks:

**1.Detecting Mutual Friends:** Identifying friends that two users have in common. This is a fundamental task in SNA, often used in social media platforms to suggest new connections.

**2. Finding the Shortest Path Between Users:** Determining the shortest connection or path between two users in a network. This helps understand how closely two individuals are connected, even if they aren't directly linked.



**LITERATURE REVIEW**

Social network analysis (SNA) leverages graph theory to understand and model relationships within networks, particularly for detecting mutual friends and determining the shortest paths between users. Various studies have explored algorithms and techniques applicable to such scenarios, with a particular focus on optimizing performance, accuracy, and scalability in real-world social networks. This review outlines key research that informs the methodology and algorithms adopted in this project.

**[1] Pathfinding Algorithms in Real-World Scenarios**

Subrata et al. (2023)[1] conducted a comparative study on pathfinding algorithms using real-world Google Maps data to assess the performance of Floyd-Warshall, Johnson’s, and Dijkstra’s algorithms. The research aimed to evaluate their suitability for different graph structures, ranging from dense to sparse, and analyse their time complexities and memory efficiency.

Key findings indicate that:

* **Floyd-Warshall** is best suited for dense graphs, though its cubic time complexity (O(V^3)) makes it costly for larger graphs.
* **Johnson’s algorithm**, with a time complexity of O(VE + V^2 log V), was identified as ideal for sparse graphs, offering a faster, memory-efficient solution using adjacency lists.
* **Dijkstra’s algorithm** is efficient for sparse graphs but is limited when handling negative weights.

This study underscores the importance of selecting appropriate algorithms for different network structures. In their project, Johnson’s and Dijkstra’s algorithms hold significant potential for efficiently calculating paths in large, sparse social networks, such as those found in online social platforms.

**[2] Link Prediction in Social Networks**

Daud et al. (2020) [2]provides an extensive review of link prediction methods within social networks, focusing on approaches that forecast future connections between nodes. The study categorizes link prediction into:

* **Graph-Based Methods**, which utilize the network's structural properties,
* **Content-Based Methods**, which rely on the attributes of nodes, and
* **Hybrid Methods**, which combine both techniques.

The study reveals several applications of link prediction, such as friend recommendations and anomaly detection, which are relevant to our project's aim of detecting mutual friends. One major challenge highlighted is the dynamic nature of social networks, where network structures evolve, making accurate and scalable prediction models necessary.

**[3] Shortest Path-Based Friend Recommendation**

Tian et al. (2012)[3] propose a novel shortest path-based method for recommending potential mutual friends in online social networks (OSNs). By combining the **Floyd-Warshall algorithm** with an **Extended Longest Common Subsequence (ELCS)** algorithm, the study integrates both topological structures and user profiles to identify potential friends between indirectly connected users.

This dual approach highlights the practicality of using both graph structure and partial user profile data to find mutual friends, a technique that informs the mutual friend detection model in our project. The integration of shortest path algorithms, particularly improved versions of Floyd-Warshall, is crucial for achieving efficient and effective friend recommendations.

**[4] Survey of Shortest Path Algorithms**

Another comprehensive study explores various shortest path algorithms, such as Bellman-Ford, Dijkstra’s, Floyd-Warshall, and Johnson’s, comparing their strengths and limitations across different graph types. The paper notes that **Bellman-Ford** and **Floyd-Warshall** can handle negative weight edges, while **Dijkstra’s** is limited in this regard. For sparse graphs, **Johnson’s**, **Prim’s**, and **Kruskal’s** algorithms are emphasized for their efficiency, while Floyd-Warshall is better suited for dense graphs.

This survey reinforces the decision to utilize Dijkstra’s algorithm in our project for their scalability and efficiency in large networks, where detecting the shortest paths between users is crucial. Additionally, the research points toward the future development of hybrid algorithms that can better handle large-scale networks.

**PROBLEM STATEMENT**

This project aims to model a social network as a graph where nodes represent users, and edges represent friendships. The primary objectives are twofold: first, to develop an algorithm that can efficiently detect mutual friends between any two users, thereby providing insights into shared connections and common interests; and second, to implement a shortest path algorithm to determine a path of minimum links between two users.

This will help in identifying the most direct or influential paths between users in the network, which can be beneficial for recommendation systems, targeted advertisements, and enhancing user interaction. The overall goal is to build a robust foundation for analysing and optimizing the structure and dynamics of social networks.

**OBJECTIVES**

The primary objectives of this project are:

1. To represent a social network as a graph where nodes represent users and edges represent friendships.
2. To develop an algorithm to detect mutual friends between any two users.
3. To implement a shortest path algorithm to find the minimum connection distance between two users.



**METHODOLOGY**

* 1. **Data Collection:**
* Gather data representing a social network. This data could be collected from real social media platforms (subject to privacy policies and permissions) or generated synthetically to mimic a real social network structure.
* The dataset should consist of nodes (representing users) and edges (representing connections or friendships between users).
  1. **Data Preprocessing:**
* Clean the data by removing any duplicates, missing values, or inconsistencies.
* Convert the data into a graph structure where users are represented as nodes and friendships as edges.
* Ensure that the graph is undirected and may be weighted or unweighted, depending on the nature of the connections (e.g., frequency of interactions).
  1. **Graph Representation:**
* Use graph theory to represent the social network. The graph G(V,E)G(V, E)G(V,E) will consist of a set of vertices VVV (users) and edges EEE (connections).
* Store the graph in a suitable data structure such as an adjacency list or adjacency matrix to optimize search and traversal operations.
  1. **Mutual Friends Detection:**
* For detecting mutual friends between two users, implement an algorithm that checks for common neighbours in the graph.
* Given two nodes AAA and BBB, retrieve their adjacency lists and find the intersection to determine mutual friends.
* This can be done using set intersection operations or traversal algorithms like Breadth-First Search (BFS).
  1. **Shortest Path Algorithm:**
* Implement an algorithm to find the shortest path between two users. Depending on the requirements, this could be achieved using:
  + Dijkstra's Algorithm: For finding the shortest path in a weighted graph where edges have different costs.
  + Breadth-First Search (BFS): For finding the shortest path in an unweighted graph, where each edge has equal weight.
* The algorithm will output the minimum number of edges (or "hops") between the two users, representing the shortest path in terms of social connections.
  1. **Algorithm Optimization:**

**Large-Scale Graph Handling:** For massive networks with millions of nodes and edges, consider using graph databases like Neo4j or Apache Giraph.

Apply techniques like Graph Pruning (removing low-relevance edges), Parallelization (using tools like Apache Spark), and Approximation Algorithms (estimating rather than exact paths) to manage computational complexity.

**Memory Optimization:** Utilize memory-efficient data structures and algorithms to reduce space complexity, especially for dense graphs.

1. **Visualization and Analysis:**

* Visualize the network and the results using graph visualization tools like Gephi.
* Highlight the mutual friends and shortest path between selected users to provide a clear understanding of the social network's structure and user interconnections.

1. **Validation and Testing:**

* Test the algorithms on different datasets to validate their accuracy and efficiency.
* Compare the results with known benchmarks and assess the performance of the algorithms in terms of time complexity, space complexity, and scalability.

**APPLICATION**

This project focuses on applying Social Network Analysis (SNA) techniques to enhance user experience in digital social networks by detecting mutual friends and determining the shortest path between users. The applications of this project are multifaceted:

1. Enhanced Social Connectivity: By identifying mutual friends, the system can suggest potential connections, thereby increasing user engagement and fostering a sense of community. This feature can be used in social media platforms to suggest friend recommendations, group memberships, or event participation.

2. Optimized Network Navigation: Determining the shortest path between users helps in understanding the closeness and connectivity within a social network. This can be used to optimize searches and find the most efficient route for networking, information dissemination, or message routing within the network.

3. Community Detection and Analysis: The project can help in identifying clusters or communities within a network, which can be useful for targeted marketing, content personalization, and the detection of influential users or thought leaders in specific communities.

4. Security and Privacy Management: The analysis can be extended to detect suspicious connections or clusters that may indicate fraudulent activities or malicious users. This enhances the security measures of the platform by enabling early detection and prevention of unwanted behaviours.

5. Support for Business and Research Applications: Businesses can leverage these insights for targeted advertising, market research, and consumer behaviour analysis. Similarly, researchers can use the social network data to study the dynamics of human interactions, social influences, and information spread patterns.

Overall, the project provides a robust framework for understanding and analysing social networks, leading to a more connected, efficient, and secure social networking environment.

**SYSTEM REQUIREMENTS**

**1. Hardware Requirements:**

* **Processor:** Intel Core i5 or equivalent AMD processor with a clock speed of 2.5 GHz or higher.
* **RAM:** Minimum of 8 GB of RAM to ensure smooth execution of Java applications and handling large datasets.
* **Storage:** At least 500 GB of HDD or 256 GB SSD for storing datasets, application files, and logs.
* **Graphics:** Integrated graphics are sufficient as the project does not require GPU-intensive tasks.
* **Network:** Stable internet connection for downloading dependencies and accessing online repositories.

**2. Software Requirements:**

* **Operating System:** Windows 10 or later, macOS 10.15 or later, or a Linux distribution (e.g., Ubuntu 20.04 or later).
* **Programming Language**: Java (Version 8 or higher)
* **Java Development Kit (JDK):** JDK 17 or later for developing and running the Java application.
* **Integrated Development Environment (IDE):** IntelliJ IDEA, Eclipse, for coding, debugging, and project management.
* **Build Tools:** Apache Maven or Gradle for project management and dependency management
* **Java Libraries:**
  + **Apache Commons Collections:** For efficient handling of collections and graph data structures.
  + **JGraphT:** For graph algorithms, including the shortest path algorithms.
  + **Maven/Gradle:** For dependency management and project building.
* **Database:** MySQL or PostgreSQL for storing user data and friendship connections, with appropriate JDBC drivers.
* **Version Control:** Git for version control, hosted on platforms like GitHub or GitLab.
* **Testing Framework:** JUnit for unit testing and ensuring code reliability.

**3. Additional Tools:**

* **Documentation Tools:** Javadoc for generating API documentation.
* **Collaboration Tools:** Slack or Microsoft Teams for team communication, and Trello or Jira for task management.
* **Build Automation:** Jenkins or GitHub Actions for continuous integration and continuous deployment (CI/CD).

**4. Functional Requirements**

* The system should efficiently detect mutual friends between any two users in the network.
* It should calculate the shortest path between users, showcasing the degree of separation between them.
* The application must handle large datasets and perform complex graph operations with optimal performance.
* A user-friendly interface should be provided for displaying the network and results of analysis.

**5. Non-Functional Requirements**

* **Performance**: The system must perform graph operations like shortest path calculation in real-time for small networks and within acceptable limits for larger networks.
* **Scalability**: The application should support scalability to accommodate a growing number of users and connections.
* **Security**: Ensure data privacy and secure access to user information.

**6. Deployment Requirements**

* The application should be deployable on cross-platform environments, supporting Windows, macOS, and Linux.
* It should be possible to deploy the application on cloud platforms if needed for scalability and accessibility.

**RESULT**

The Social Network Analysis (SNA) project aims to identify mutual friends and find the shortest path between users in a given social network. Based on the proposed methodologies and algorithms, we expect the following outcomes:

1. **Identification of Mutual Friends**:  
   By analysing the connections within the network, the system will accurately detect mutual friends between any two users. The expected result is that the algorithm will efficiently handle large datasets, providing results with a low time complexity. The visualization of mutual friends for each user pair will help in understanding network clusters and shared social connections.
2. **Shortest Path Calculation**:  
   The system will utilize graph traversal algorithms, such as Breadth-First Search (BFS) or Dijkstra’s Algorithm, to determine the shortest path between two users. The expected result is that the algorithm will return the shortest path even in dense and highly connected networks. This will enhance our understanding of how information or influence can propagate between users in the network.
3. **Scalability and Efficiency**:  
   We anticipate that the designed system will be scalable and capable of handling large social networks with thousands of nodes (users) and edges (connections). Performance tests are expected to show the system’s robustness, with results produced within an acceptable time frame, even for complex queries.
4. **Insights into Network Structure**:  
   The mutual friends and shortest path functionalities will provide valuable insights into the overall structure of the network. We expect the results to highlight key users, popular clusters, and potential weak points in connectivity, which could be useful for further social network analysis studies.

In summary, the system is expected to perform efficiently and provide accurate results for detecting mutual friends and finding the shortest path between users. The analysis will also contribute to a deeper understanding of social connections within a given network.

**CONCLUSION**

The project on Social Network Analysis (SNA) for detecting mutual friends and finding the shortest path between users provides valuable insights into understanding the structural properties and dynamics of social networks. Through the application of graph theory and network analysis algorithms, we have effectively demonstrated how to identify mutual friends among users, which is crucial for enhancing social connections and recommendations in social media platforms.

Furthermore, the shortest path algorithm implementation showcases how efficiently we can compute the minimum steps or connections required for any two users to reach each other within a network. This has significant implications for optimizing communication strategies, designing better recommendation systems, and improving overall user engagement on social networking sites.

By leveraging SNA techniques, this project contributes to developing more intuitive and user-friendly social networks, fostering better community-building and enhancing the user experience. Overall, this study serves as a foundation for further exploration and development in the field of social network analytics, enabling the creation of more connected and interactive digital ecosystems.



**REFERENCES**

1. Subrata, P. N., Tandra, P. A., Owen, C., Yuwono, A., & Astriani, M. S. (2023). Comparative research on all to all pairs path finding algorithms in a real-world scenario. In E3S Web of Conferences (Vol. 426, p. 01024). EDP Sciences.

<https://www.e3s-conferences.org/articles/e3sconf/abs/2023/63/e3sconf_icobar23_01024/e3sconf_icobar23_01024.html>

1. Daud, N. N., Ab Hamid, S. H., Saadoon, M., Sahran, F., & Anuar, N. B. (2020). Applications of link prediction in social networks: A review. Journal of Network and Computer Applications, 166, 102716.<https://www.sciencedirect.com/science/article/pii/S1084804520301909?casa_token=nJN9UOS8Ux4AAAAA:wp0HpvhyEpPaJNg8ABNFsl7XUwqHKZyzmWV8MING6xm7xBPr5IxLGXLWLoRZSzPmWeaxsMZyBn5B>
2. X. Tian, Y. Song, X. Wang and X. Gong, "Shortest Path Based Potential Common Friend Recommendation in Social Networks," 2012 Second International Conference on Cloud and Green Computing, Xiangtan, China, 2012, pp. 541-548, doi: 10.1109/CGC.2012.106. keywords: {Social network services;Arrays;Collaboration;Business;Educational institutions;Algorithm design and analysis;Data models;Friend recommendation;Online social network;Shortest path;Longest Common Subsequence},

<https://ieeexplore.ieee.org/abstract/document/6382869/references#references>

1. Barkund, S. H., Sharma, A., & Bhapkar, H. R. Survey of Shortest Path Algorithms.

<https://www.researchgate.net/publication/367030421_Survey_of_Shortest_Path_Algorithms>