ISSN: 2582 - 6379
IJISEA Publications
International Journal for Interdisciplinary Sciences and Engineering Applications
IJISEA - An International Peer- Reviewed, Open Access Journal
2025, Volume 6 Issue 3
www.ijisea.org

Identifying Link Failures Using A Graph-Based Network Method

Potru Durga Bhavani*[1] , Anumuladasu Nagendra Ramamani[2]
[1,2]Department Master of Computer Application(MCA), SVKP & Dr KS Raju Arts & Science College(A), Penugonda, W.G.Dt., A.P India
durgabhavanipotru@gmail.com[1], ramasrinivasu2@gmail.com[2]

ABSTRACT

In modern computer networks, even a single link failure can cause major disruptions by changing routing paths unexpectedly. Manually identifying the exact point of failure is time-consuming, error-prone, and often depends on guesswork. To overcome this, this project proposes a graph-based method to detect link failures, restored links, and unchanged links automatically. The network is modeled as a graph, with systems or routers as nodes and links as edges. By comparing two snapshots of the network one before and one after a failure the Find Change algorithm labels each link as unchanged, failed, or restored. The system uses simple input files, processes the data using Java, and clearly shows which connections have failed. This method provides a practical, accurate, and automated way to detect faults and improves overall network reliability.

Keywords: Link Failure Detection, Graph-Based Network Analysis, Network Fault Identification, FindChange Algorithm, Network Topology, Graph Theory, Network Reliability, Routing Path Analysis, Automated Fault Detection, Network Connectivity Monitoring.

I. INTRODUCTION

In today's digital era, continuous and reliable network connectivity is essential for organizations and Internet Service Providers (ISPs). A single link failure can lead to multiple path changes, resulting in data loss, service downtime, or degraded performance[Ref2]. Existing methods for detecting the cause of such failures often rely on manual checks and the experience of network administrators, making them inefficient and prone to errors[Ref4,Ref5].To address this challenge, this project introduces a formal graph-based approach for identifying link failures. The proposed system models the network as a graph, captures two snapshots at different times, and applies the Find Change algorithm to detect which links failed, stayed the same, or were restored.[Ref3(Paper),Ref3(Web)]. Although inspired by the behavior of the Border Gateway Protocol (BGP), the project does not implement the BGP protocol directly; instead, it uses Graph Theory concepts to simulate routing changes and pinpoint the exact cause of disruptions.[Ref2(Paper),Ref1(Paper)]This approach provides an automated and structured solution for link fault detection, forming a strong foundation for developing more advanced network diagnostic tools in the future.[Ref6].

II. LITERATURE SURVEY

ISSN: 2582 - 6379
IJISEA Publications
International Journal for Interdisciplinary Sciences and Engineering Applications
IJISEA - An International Peer- Reviewed, Open Access Journal
2025, Volume 6 Issue 3
www.ijisea.org

The problem of link failure identification in networks has been extensively studied, with various traditional and modern approaches evolving over time. Below is a review of key contributions from the literature, categorized by author, title, and contribution.

1. Author: Papadimitriou et al.

Title: Network Fault Localization: A Path-Based Approach

Description: Papadimitriou et al. (2004) proposed a path-based method using network tomography to detect failures by analyzing the paths taken by probe packets. Their method infers failure points based on discrepancies between expected and actual path behaviors. This technique proved useful for end-to-end monitoring in large networks but was limited in scalability for dynamic topologies.

2. Author: Medhi, Murray, and Doverspike
Title: IP-layer Restoration in Mesh Networks Using Network Sharing
Description: This study focused on fault restoration at the IP layer, particularly in mesh networks. The authors discussed dynamic rerouting strategies and shared backup paths to recover from failures quickly. Their approach was efficient but relied heavily on manual configuration and predefined routes, which reduced adaptability in real-time environments.

3. Author: Jiang, Yang, and Luo
Title: An Intelligent Deep Learning Model for Network Fault Prediction in SDN
Description: Jiang et al. (2019) introduced a deep learning-based framework for predicting faults in Software-Defined Networks (SDN). Utilizing Recurrent Neural Networks (RNNs), the system could forecast failures by learning from historical traffic behavior. Their approach significantly reduced false positives but required large datasets and computational resources.

III. PROPOSED SYSTEM

The proposed system for identifying link failures is designed to provide real-time, automated monitoring of network connections. It continuously tracks the health and performance of network links and instantly detects failures or abnormal behavior, such as high latency or packet loss. Upon detecting an issue, the system triggers alerts to network administrators and logs the event for analysis. This proactive approach reduces downtime, minimizes manual intervention, and ensures faster recovery. Additionally, the system can include a visual dashboard for better monitoring and optional AI-based prediction to foresee potential failures before they occur.

System Architecture

ISSN: 2582 - 6379
IJISEA Publications
International Journal for Interdisciplinary Sciences and Engineering Applications
IJISEA - An International Peer- Reviewed, Open Access Journal
2025, Volume 6 Issue 3
www.ijisea.org

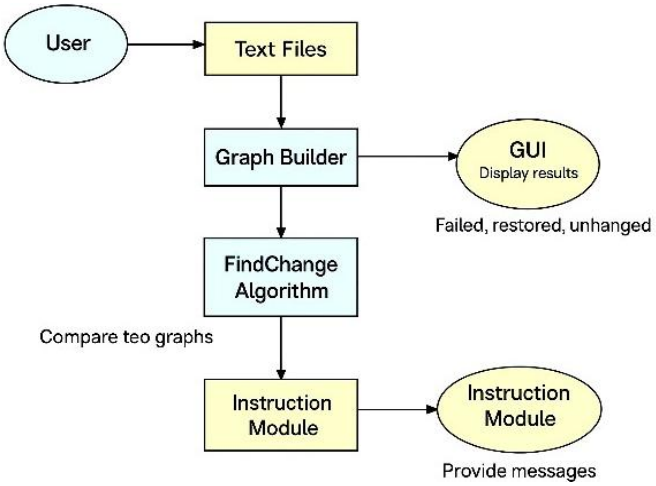


Figure 1: Shows the Architecture of Proposed Methodology

System Modules

1. User

- The end-user interacts with the system by providing input in the form of text files.
- These files typically contain network topology information (graph data).

2. Text Files

- Serve as the input source for the system.
- Store graph structure details such as nodes, edges, and connections.
- Provide the raw data required for graph construction and comparison.

3. Graph Builder

- Reads and interprets the input text files.
- Constructs graph structures from raw data.
- Converts the input into a usable graph representation.
- Forwards processed results to the GUI for visualization.

4. GUI (Graphical User Interface)

- Displays the analysis results to the user.

 Visually highlights:

o Failed links

o Restored links o Unchanged network paths

- Offers an interactive, user-friendly interface to understand system outputs.

5. Find Change Algorithm

- Acts as the core logic engine of the system.
- Compares two graphs (before and after network changes).
- Detects and classifies differences such as:
  - o Link failures
  - o Restorations
  - o Unchanged paths

6. Instruction Module (Processing)

- Processes the results from the Find Change Algorithm.

ISSN: 2582 - 6379
IJISEA Publications
International Journal for Interdisciplinary Sciences and Engineering Applications
IJISEA - An International Peer- Reviewed, Open Access Journal
2025, Volume 6 Issue 3
www.ijisea.org

- Structures the detected changes into clear, actionable instructions.
- Ensures results are in a well-organized format for user interpretation and decision-making.

IV. METHODOLOGY

In this project, a structured and formal method is adopted to identify link failures in a network using a graph-based approach. The methodology revolves around analyzing the changes in network topology over time by comparing snapshots of the network before and after a failure occurs. By treating the network as a graph consisting of nodes and edges (links), the system systematically identifies which connections have failed, which have been restored or added, and which remain unchanged.

To accomplish this, two input files representing the initial and final network graphs are processed. Using the Find Change algorithm, the system compares these graphs to detect any connectivity changes. The methodology ensures accurate and automated detection of network disturbances without requiring full knowledge of the underlying topology. The result is a practical solution for network administrators to diagnose and respond to link failures efficiently.

1.Network Initialization: The simulation environment initializes the number of nodes, transmission range, mobility model (e.g. Random Waypoint),and routing protocol parameters.

2.Link Monitoring: Each node continuously monitors the status of its neighboring nodes through periodic HELLO messages or beacon signals. A link is considered failed if there is no response from a neighbor after a predefined timeout.

3.Failure Detection: The system implements a failure detection mechanism that triggers when message acknowledgments are missed repeatedly, or HELLO messages are not received. The cause of the failure is analyzed—whether due to node mobility, signal interference, or battery depletion.

4.Logging and Reporting: Once a failure is detected, it is logged with details including time of failure, nodes involved, and possible cause. This information is then used for analysis and can be displayed in a GUI or stored in log files.

5.Simulation and Testing: Multiple test scenarios with varying node densities, mobility rates, and failure conditions are simulated to evaluate the accuracy and responsiveness of the failure detection mechanism.
6.Performance Evaluation: The effectiveness of the link failure identification mechanism is evaluated based on metrics such as detection time, false positives, and the impact on routing performance.

By following this methodology, the project aims to develop a robust system capable of efficiently detecting link failures in a dynamic MANET environment, thereby improving the reliability and performance of wireless communication network

ISSN: 2582 - 6379
IJISEA Publications
International Journal for Interdisciplinary Sciences and Engineering Applications
IJISEA - An International Peer- Reviewed, Open Access Journal
2025, Volume 6 Issue 3
www.ijisea.org

Algorithm: Find Change

Input:

- T0=(V0,E0)T_0 = (V_0, E_0): The PT from Mat time t0t_0
- T1=(V1,E1)T_1 = (V_1, E_1): The PT from Mat time t1t_1

Output:

- Marked edges: unchanged, vanished, failed, added

Procedure

1. Initialize:
    - V=V0∪V1V = V_0 \cup V_1
    - E=E0∪E1E = E_0 \cup E_1
    - E add= E fail=EE_{add} = E_{fail} = E
    - T fail= T add=PT(V,E)T_{fail} = T_{add} = PT(V, E)
2. For each edge e∈Ee \in E:
    - **Case 1:** If e∈(E0∩E1)e \in (E_0 \cap E_1)
        - Mark e=unchangede = \textbf{unchanged}
    - **Case 2:** Else if e∈E0e \in E_0
        - Mark e=vanishede = \textbf{vanished}
        - If e∈Tfaile \in T_{fail}:
        - Mark e=failede = \textbf{failed}
        - Update:
        - Efail=Efail−{e}E_{fail} = E_{fail} - \{e\}
        - Tfail=PT(V,Efail)T_{fail} = PT(V, E_{fail})
    - **Case 3:** Else if e∈E1e \in E_1
        - Mark e=appearede = \textbf{appeared}
        - If e∈Tadde \in T_{add}:
        - Mark e=addede = \textbf{added}
        - Update:
        - Eadd=Eadd−{e}E_{add} = E_{add} - \{e\}
        - Tadd=PT(V,Eadd)T_{add} = PT(V, E_{add})

Input Graph

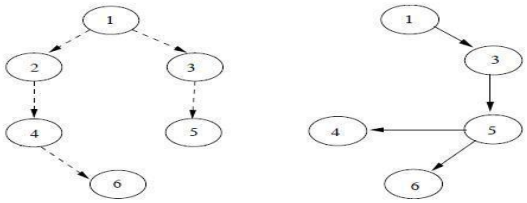


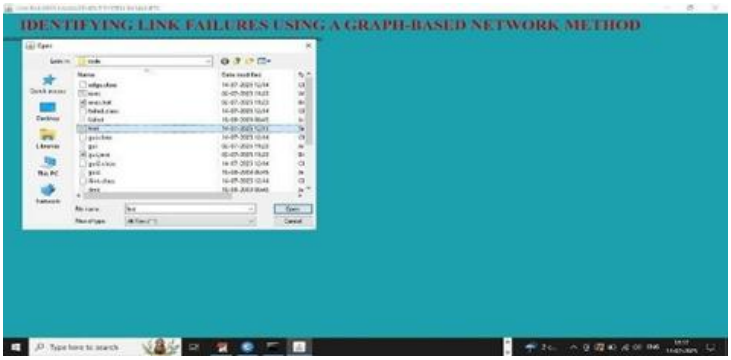
Figure 2: Shows the Input Graph for the Proposed Methodology

ISSN: 2582 - 6379
IJISEA Publications
International Journal for Interdisciplinary Sciences and Engineering Applications
IJISEA - An International Peer- Reviewed, Open Access Journal
2025, Volume 6 Issue 3
www.ijisea.org

V. RESULTS AND DISCUSSION



This is the main home screen of the project. It shows buttons to enter initial/final graphs, get links and detect changes using a graph-based



First file selection screen to load the initial graph file. The user must select both initial and final files to run the comparison.

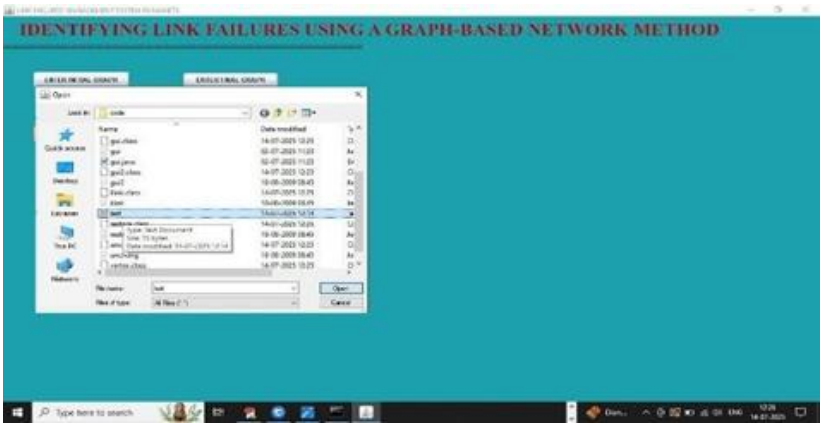


This is a sample input graph file opened in Notepad. It lists vertex and edge pairs that define the initial network topology.

ISSN: 2582 - 6379
IJISEA Publications
International Journal for Interdisciplinary Sciences and Engineering Applications
IJISEA - An International Peer- Reviewed, Open Access Journal
2025, Volume 6 Issue 3
www.ijisea.org



It displays vertices and their edges to show current network connections. Another file selection screen to load the initial graph file. The user must select both initial and final files to run the comparison.



This is a sample input graph file opened in Notepad. It lists vertex and edge pairs that define the initial network topology.

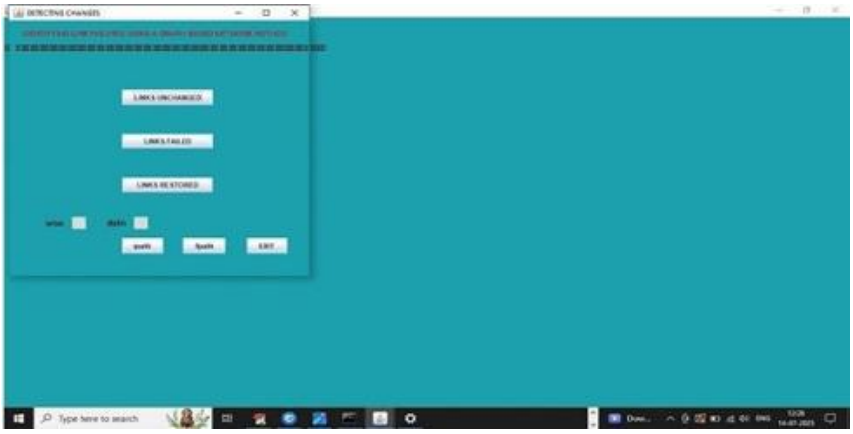


It displays vertices and their edges to show current network connections.

ISSN: 2582 - 6379
IJISEA Publications
International Journal for Interdisciplinary Sciences and Engineering Applications
IJISEA - An International Peer- Reviewed, Open Access Journal
2025, Volume 6 Issue 3
www.ijisea.org



This is the main interface of the project showing buttons to check link status—unchanged, failed, and restored. The user can input source and destination nodes to check routing using internal or final path detection.

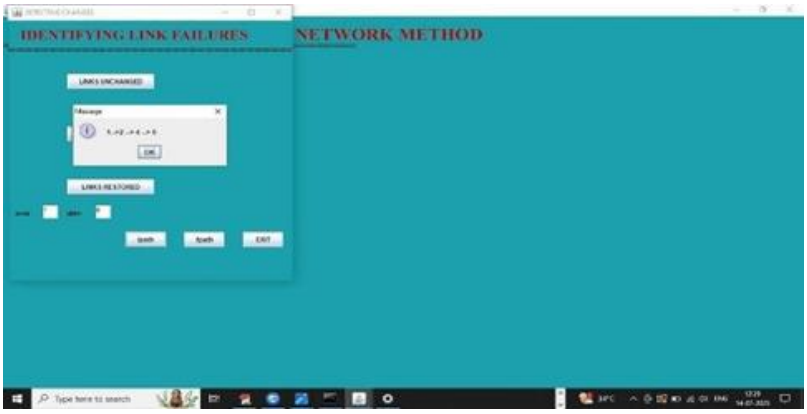


This window displays a list of failed links detected by comparing two graph states.

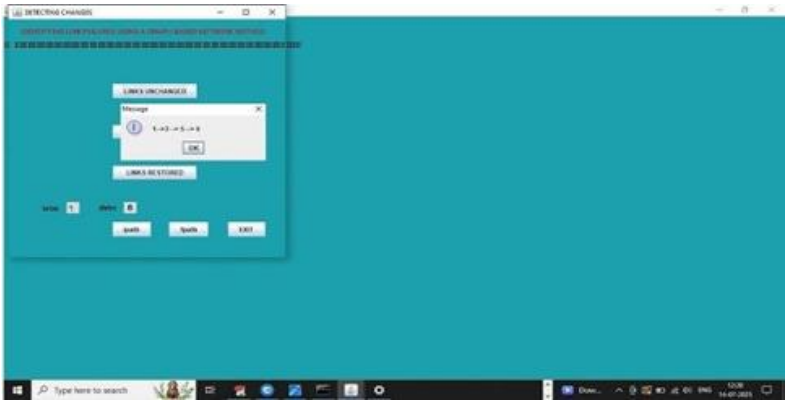


Content: This window shows links that were restored in the network indicating the system's ability to track recovered connections

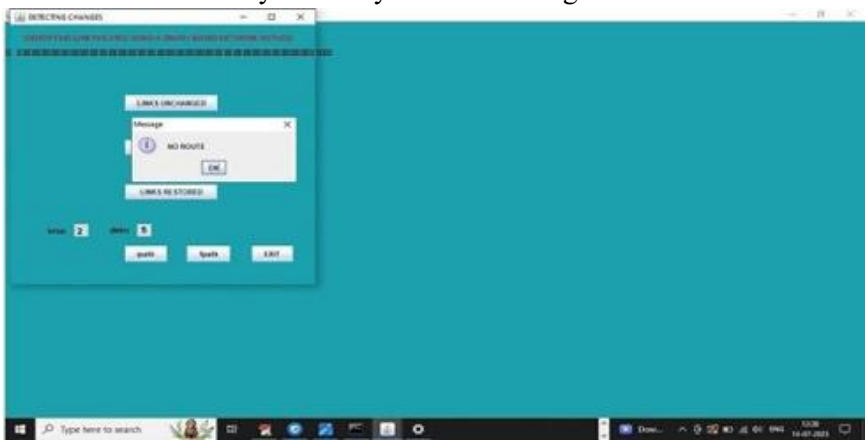
ISSN: 2582 - 6379
IJISEA Publications
International Journal for Interdisciplinary Sciences and Engineering Applications
IJISEA - An International Peer- Reviewed, Open Access Journal
2025, Volume 6 Issue 3
www.ijisea.org



This shows the unchanged link path from source to destination. It confirms the system can trace a valid route using graph logic.



This output shows a successful routing path from source node 1 to destination node 6: 1 → 3 → 5 → 6. It indicates that the network has dynamically rerouted through restored and available links.



This error message appears when there is no valid path between the selected source (2) and destination (5).

VI. CONCLUSION

ISSN: 2582 - 6379
IJISEA Publications
International Journal for Interdisciplinary Sciences and Engineering Applications
IJISEA - An International Peer- Reviewed, Open Access Journal
2025, Volume 6 Issue 3
www.ijisea.org

This project demonstrates a simple yet effective method to detect connectivity changes in a network using information collected from a single snapshot vantage point. By comparing two network snapshots represented as graphs, the system accurately identifies which links have failed, which have been restored or newly added, and which remain stable. This practical implementation of the FindChange concept proves that graph-based analysis can help locate the exact source of routing changes without needing complete live routing data. Overall, this work provides a valuable starting point for developing more advanced diagnostic tools for network management and inter-domain routing fault analysis.

FUTURE WORK:

Future enhancements can expand the Find Change approach to work with data collected from multiple monitoring points instead of just a single vantage point. This will help capture a more complete view of the network and improve the accuracy of fault detection. Additionally, the system can be extended to analyze more complex failure scenarios, including multiple link failures and simultaneous link recoveries, and to process real-time routing updates instead of static snapshots. Integration with live monitoring protocols and advanced visualization tools can also transform this project into a practical tool for real-world network operations.

REFERENCES:

[1] Patrick Naughton and Herbert Schildt, *The Complete Reference JAVA 2*.
[2] Andrew S. Tanenbaum, *Computer Networks*.
[3] Dietel and Deitel, *JAVA: How To Program*.
[4] Roger S. Pressman, *Software Engineering: A Practitioner's Approach*.
[5] Sommerville, *Software Engineering*.
[6] Grady Booch, James Rambaugh, and Ivar Jacobson, *The Unified Modeling Language User Guide*.
[7] *Swings*. Available: http://java.sun.com/docs/books/tutorial/uiswing/components/index.html
[8] *Object Oriented Modeling and Design*. Available: http://www.visualcase.com/tutorials/umltutorial.htm
[9] *Graph Theory of Internet*. Available: http://expertvoices.nsdl.org/cornellinfo204/2008/03/02/graph-theory-in-internet-hardware/
[10] R. Govindan and H. Tangmunarunkit, *Heuristics for Internet Map Discovery*, in *IEEE INFOCOM 2000*, pp. 1371–1380, Tel Aviv, Israel, March 2000. IEEE.
[11] Y. Rekhter and T. Li, *A Border Gateway Protocol (BGP-4)*, Request for Comment (RFC): 1771, March 1995.