The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82-98

Optimizing Network Systems: A Study of Star, Wheel, Ring and Mesh Topologies in Real-world Applications

R. Kanaka Raju[1], P. Likhita[2], K. Kala Harshini[3], B. Tarun Kumar[4], K. Mohit Vinayak[5]
[1,2,3,4,5]Department of Computer Science and Engineering, GVPCDPGC(A), Rushikonda, Visakhapatnam.
mailID:rkanakaraju@gvpcdpgc.edu.in,5211411047.gvpcdpgc.edu.in

Abstract

There are various types of topologies which are used in networks, and contribute a lot to the efficiency, scalability and fault tolerance of the communication networks. This study provides a comprehensive comparison of four basic topologies: star, wheel, ring and complete graph. Different topologies have unique properties affecting network performance metrics such as average path length, clustering coefficient, and fault tolerance. Star Topology consists of a central hub that connects to peripheral nodes. Wheel topology is a hybrid topology features from star topology and ring topology. Ring topology offers equal connectivity among nodes but also longer length paths. Complete graphs connect every node with every other node and maximize connectivity but are less practical, as they feature high edge density. This review evaluates these topologies under different types of network parameters, their scalability and suitability for different applications.

Keywords:

Network Topology Comparison, Wiener index, Average Path Length, Clustering Coefficient, Fault Tolerance, Network Scalability, average degree of N/W, Central Node Architecture

1.Introduction

Network topologies acts as a major role in modern communication systems, influencing the efficiency, scalability, and resilience of networks. The choice of topology can significantly impact network performance, affecting its metrics such as data transmission speed, fault tolerance, and overall system reliability [1]. Among the various network architectures, star, wheel, ring, and complete graph topologies are fundamental structures that have been mostly adopted in different contents.

Star Topology: In this topology, all the nodes are connected by a central node which is also as head node/ cluster node these all central nodes are connected to a sink node [2]. Star networks are simple and easy to manage but vulnerable to central node failures.

Wheel Topology: In this topology is the combining elements of star and ring structures. wheel networks offer improved and better connectivity and redundancy compared to star networks [3].

Ring Topology: In a ring topology, nodes are connected in a circular way, providing equal

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82–98

connectivity among nodes but potentially it finds difficulty from longer path lengths [4].

Complete Graph or Mesh Topology: Complete graph topology in which every node is connected to every other node, complete graphs maximize connectivity but are often impractical due to its high edge density and complexity [5].

Understanding the strengths and limitations of each topology is crucial for designing efficient and resilient networks. This study aims to provide a comprehensive comparison of these architectures, exploring their network metrics, scalability, and application suitability [6]. By examining the trade-offs between different parameters like connectivity, efficiency, and complexity, this analysis will guide network designers in selecting the most appropriate topology for specific applications, contributing to the development of more effective network architectures.

2. Methodology

This comparative analysis of star, wheel, ring, and complete graph architectures, it involves detail evaluation of the networks in various aspects to conclude a results for the topologies

1.Network Construction

Star Topology: A central node is connected to all peripheral nodes, with no connections between internal nodes.

Wheel Topology: A central node is connected to all peripheral nodes, and peripherals are also connected in a ring.

Ring Topology: Nodes are connected in a circular structure, with each node connected to its two neighbours.

Complete Graph Topology: Every node is connected to every other node.

2. Network Metrics Calculation

Average Degree: The average number of edges per node, calculated as the total number of edges divided by the number of nodes [7].

Clustering Coefficient: Measures the degree to which nodes tend to cluster together, calculated using the formula for local clustering coefficient [8].

Average Path Length: The average number of edges between any two nodes, calculated using shortest path algorithms [9].

Network Diameter: The maximum shortest path between any two nodes in the network.

Fault Tolerance: Assessed by simulating node failures and measuring the impact on network connectivity.

3. Simulation and Analysis

Network X Library: Utilized for constructing and analyzing network topologies.

Python Scripts: Developed to automate the calculation of network metrics and simulate node failures.

Visualization Tools: Employed to visualize network structures and facilitate understanding of

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82–98

topology differences.

4. Comparison and Evaluation

Quantitative Comparison: Network metrics are compared across topologies to highlight differences in efficiency and resilience.

Qualitative Analysis: The practicality and suitability of each topology for different applications are evaluated based on their characteristics.

5. Conclusion and Recommendations

Summary of Findings: A comprehensive summary of the strengths and weaknesses of each topology.

Application Recommendations: Guidance on selecting the most appropriate topology [10-15] for specific network applications based on the analysis.

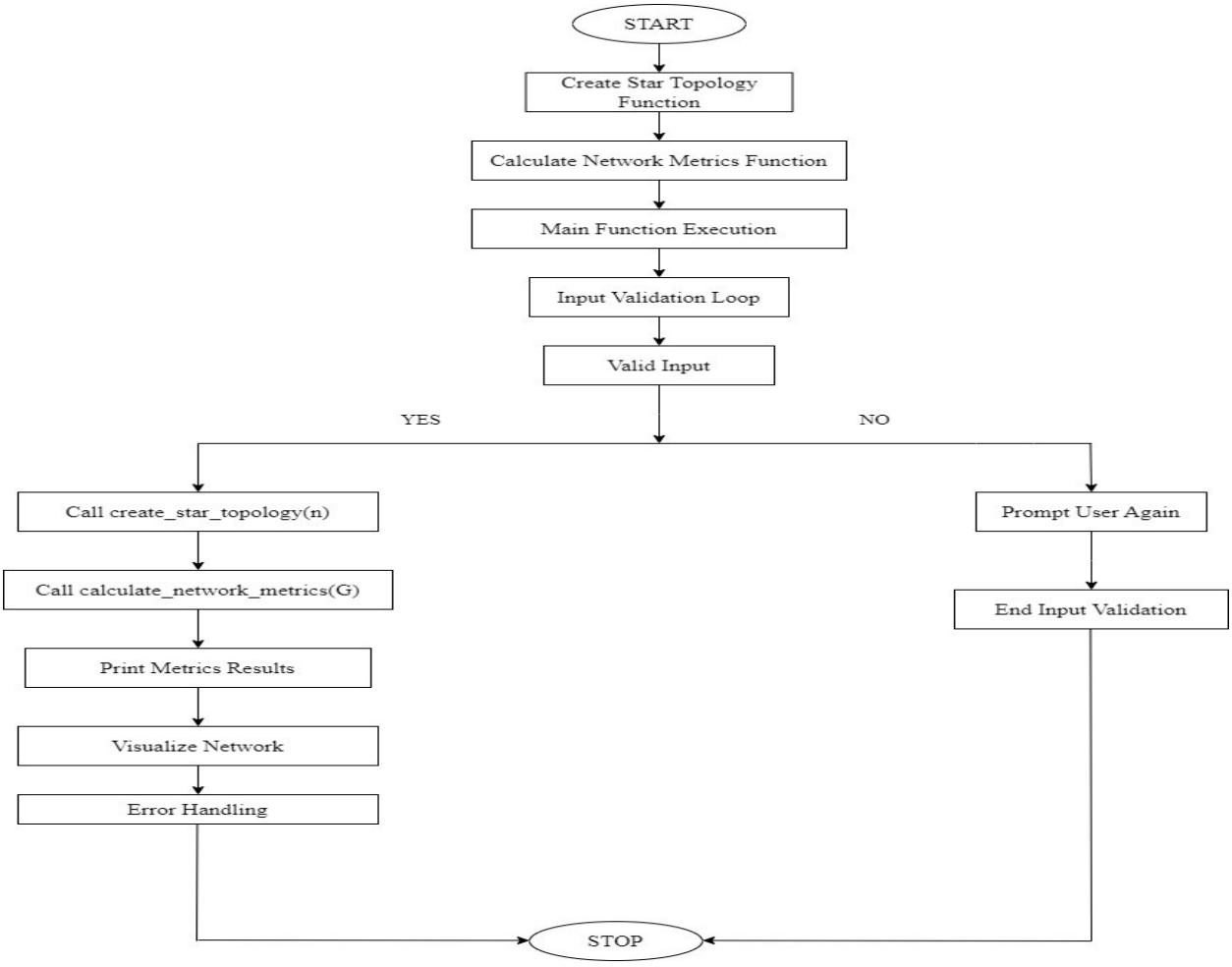

Fig. 1. Flowchart representation to choose the choice of different Cluster

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82-98

Algorithm to choose the choice of different Cluster

Step 1: Import Required Libraries
- Import the necessary libraries:
  - `networkx` for graph manipulation.
  - `numpy` for numerical operations.
  - `matplotlib.pyplot` for visualization.

Step 2: Define Function to Create Star Topology
1. Function Definition: Define a function `create_star_topology(n)` that takes an integer `n` as input.
2. Initialize Graph: Create an empty graph `G` using `nx.Graph()`.
3. Define Nodes:
   - Set the central node as `0`.
   - Create a list of peripheral nodes from `1` to `n-1`.
4. Add Edges: Loop through each peripheral node and add an edge from the central node to each peripheral node using `G.add_edge()`.
5. Assign Positions:
   - Define the position of the central node at the origin `(0, 0)`.
   - Calculate angles for peripheral nodes using `np.linspace()` to evenly distribute them in a circular layout.
   - Assign positions to each peripheral node based on calculated angles using trigonometric functions (`cos` and `sin`).
6. Return Graph and Positions: Return the graph `G` and the position dictionary `pos`.

Step 3: Define Function to Calculate Network Metrics
1. Function Definition: Define a function `calculate_network_metrics(G)` that takes a graph `G` as input.
2. Initialize Metrics Dictionary: Create an empty dictionary `metrics`.
3. Calculate Wiener Index:
   - Use nested loops to calculate the sum of shortest path lengths between all pairs of nodes (excluding pairs where source equals target).
   - Store the result in `metrics['wiener_index']`.
4. Calculate Clustering Coefficient:
   - Use `nx.clustering(G)` to calculate clustering coefficients for all nodes and store in `metrics['clustering_coefficient']`.
5. Calculate Average Path Length:

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82-98

- Check if the graph is connected using `nx.is_connected(G)`. If connected, calculate average path length using `nx.average_shortest_path_length(G)`, otherwise store "Graph is not connected".

6. Count Edges: Store the number of edges in `metrics['edge_count']` using `G.number_of_edges()`.

7. Calculate Average Degree:
   - Retrieve degrees of all nodes using `dict(G.degree())`.
   - Calculate average degree by summing degrees and dividing by the number of nodes.
   - Store in `metrics['average_degree']`.

8. Return Metrics: Return the metrics dictionary.

Step 4: Main Function Execution

1. Input Validation Loop:
   - Use a loop to prompt the user for input until a valid integer (≥ 2) is provided for total nodes.

2. Create Star Topology:
   - Call `create_star_topology(n)` with user input and store results in variables `G` and `pos`.

3. Calculate Network Metrics:
   - Call `calculate_network_metrics(G)` and store results in variable `metrics`.

4. Print Metrics: Output each metric from the metrics dictionary.

5. Visualize Network:
   - Use Matplotlib to create a figure and draw the graph with specified positions, colors, and labels.
   - Set a title for the plot and display it.

Step 5: Error Handling

- Wrap the main execution block in a try-except structure to catch and print any exceptions that occur during execution.

Step 6: Run Main Function

- Check if this script is being run as the main program, and if so, call the main function.

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82–98

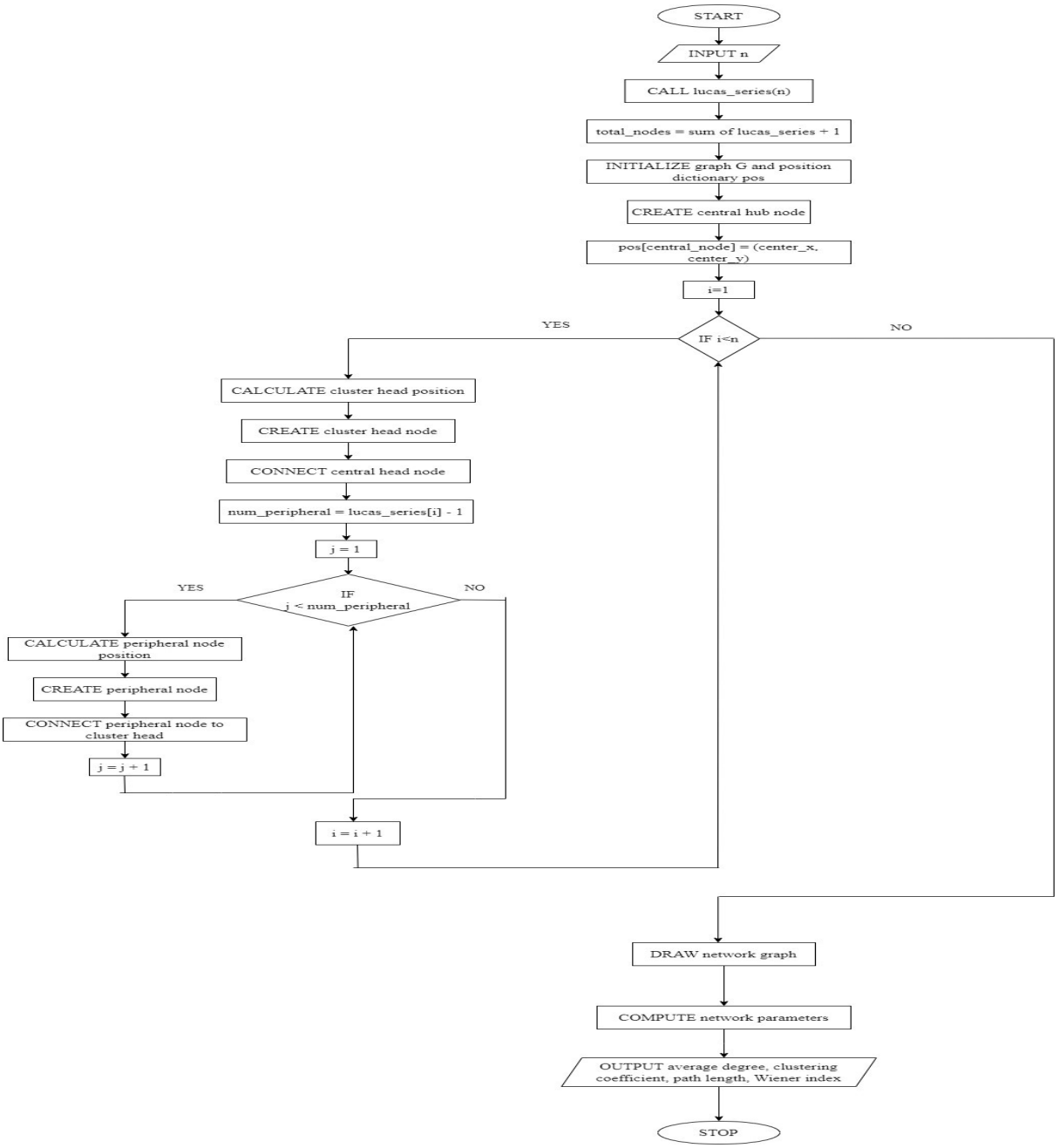

Fig. 2. Flowchart representation of Star of Star Network using Lucas Series

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82–98

Algorithm of Star of Star Network using Lucas Series

Step 1: Input
1. Read the number of clusters, `n`.

Step 2: Generate Lucas Series
2. Compute the first `n` terms of the Lucas series using the recurrence relation:
   - \(L_0 = 2 \), \(L_1 = 1 \)
   - \(L_n = L_{n-1} + L_{n-2} \) for \(n \geq 2 \)
3. Calculate the total number of cluster nodes as the sum of the Lucas series values.
4. Compute the total number of nodes: `total_nodes = total_cluster_nodes + 1` (including the central hub).

Step 3: Initialize Graph
5. Create an empty graph `G`.
6. Initialize a position dictionary `pos` for node placements.

Step 4: Create Central Hub
7. Define the central hub node at `(center_x, center_y)`.
8. Add the central hub to the graph `G`.

Step 5: Create Clusters
9. Set cluster parameters:
   - `cluster_radius`: Distance from the hub to cluster heads.
   - `node_radius`: Distance from the cluster head to its peripheral nodes.
10. For each cluster \(i = 1 \) to \(n \):
    - Compute the position of the cluster head using circular placement around the hub.
    - Create the cluster head node and connect it to the central hub.
    - Determine the number of peripheral nodes as `lucas_numbers[i] - 1`.
    - For each peripheral node \(j = 1 \) to `num_peripheral`:
      - Compute the position of the node around the cluster head.
      - Add the node to the graph and connect it to the cluster head.

Step 6: Visualize and Compute Metrics
11. Draw the graph with node colors:
    - Central hub → Gold
    - Cluster heads → Red
    - Peripheral nodes → Light blue
12. Compute and output network metrics:

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82–98

- Average degree
- Clustering coefficient
- Average shortest path length
- Wiener index (sum of shortest path distances)

Step 7: Output Results
13. Display the Lucas series and total nodes.
14. Print the number of nodes in each cluster.
15. Print the computed network parameters.

3. Result

These are the different topologies where it shows the connectivity among nodes after entering the 'n' no. of clusters
It is analyzed through parameters such as wiener index, edges of network, cluster coefficient, Average degree of the network, clustering of a network and Average path length of a Network.

Star topology without Lucas

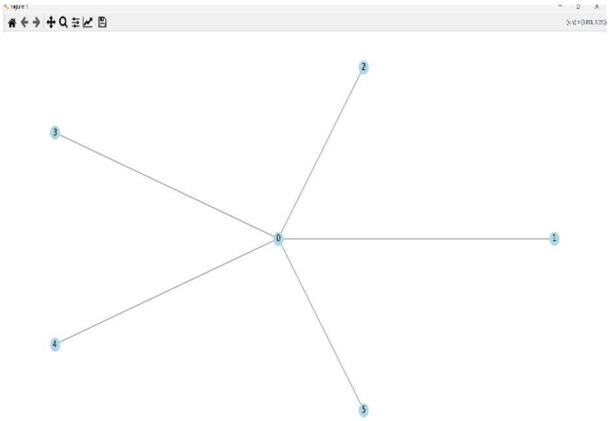

Fig 1 For 6 nodes

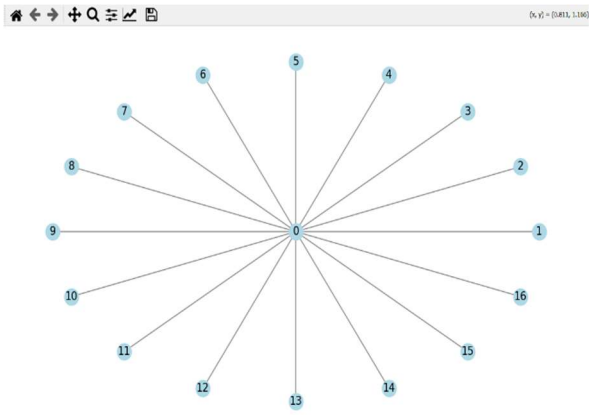

Fig 2 For 17 nodes

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82–98

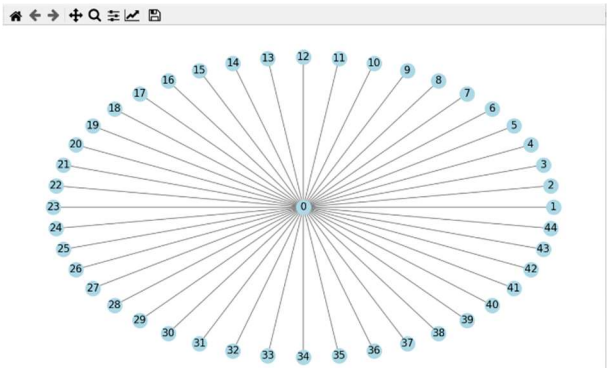

Fig 3 For 45 nodes

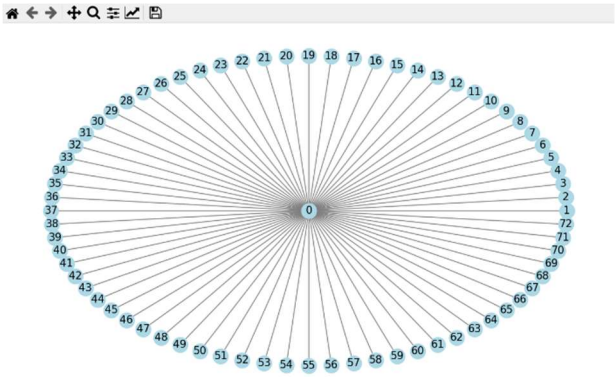

Fig 4 For 73 nodes

Star topology with Lucas

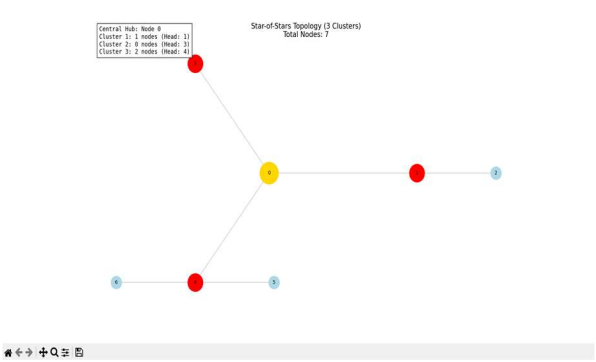

Fig 5 For 3 Clusters

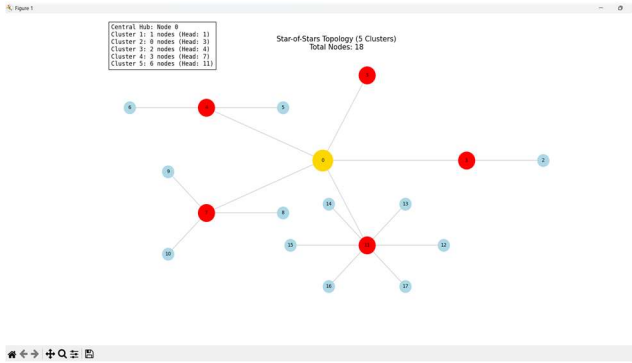

Fig 6 For 6 Clusters

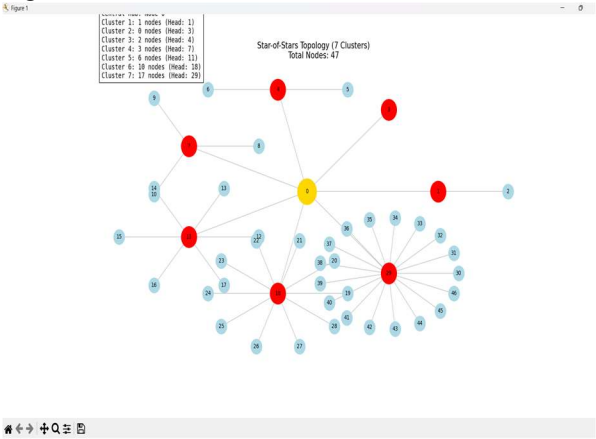

Fig 7 For 7 Clusters

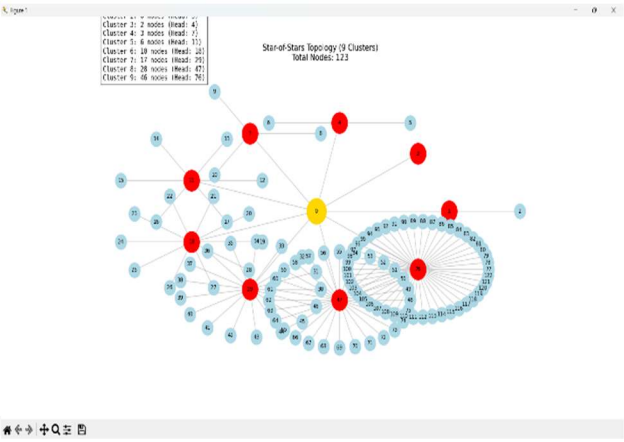

Fig 8 For 9 Clusters

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82–98

Comparative analysis:

The analysis of different network topologies (Star, Ring, Wheel, and Mesh) with and without Lucas configurations reveals distinct trade-offs between efficiency, connectivity, and complexity [16-20]. Lucas configurations consistently reduce the Wiener index, indicating improved efficiency and shorter average path lengths, while increasing edge counts and average degrees, enhancing connectivity. For example, Ring networks with Lucas achieve significantly lower Wiener indices compared to their non-Lucas counterparts, but at the cost of higher clustering coefficients and edge density. Mesh networks exhibit exponential growth in edges and clustering coefficients with Lucas, resulting in hyper connectivity and exceptional efficiency gains but raising scalability concerns. Star networks maintain minimal clustering and moderate efficiency improvements, while Wheel networks balance efficiency and connectivity effectively [21-23]. Overall, Lucas configurations optimize network performance by reducing path lengths and enhancing connectivity, but the extent of these benefits varies across topologies based on their structural characteristics.

Comparison of Wiener Index in Topologies with and without Lucas

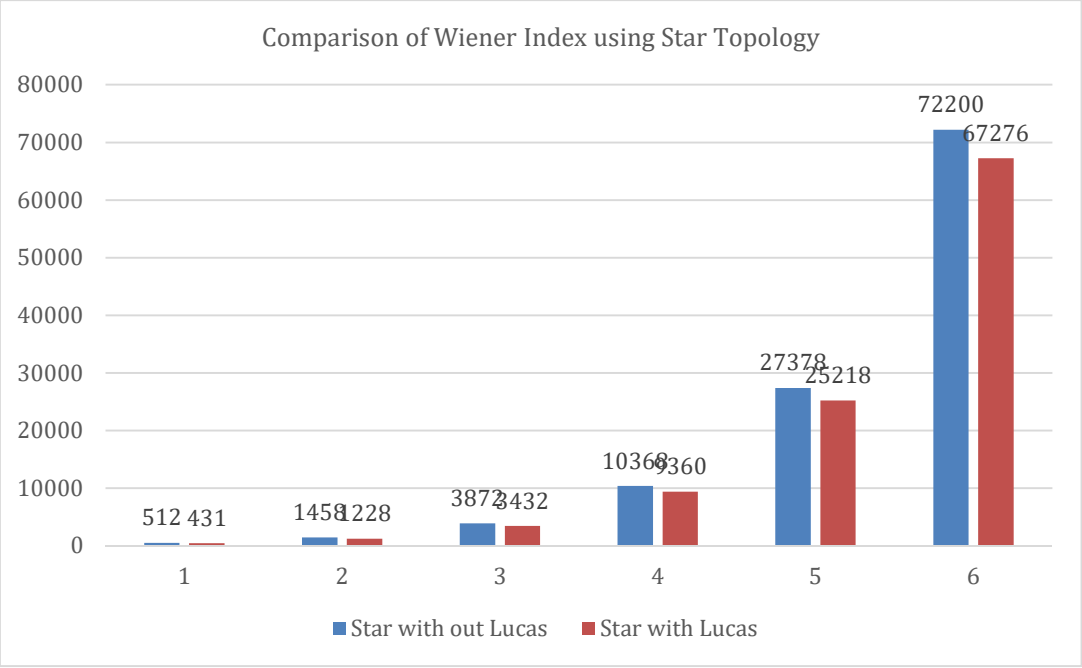

Fig. 9 Comparison of Wiener Index using Star Topology with and without Lucas

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82– 98

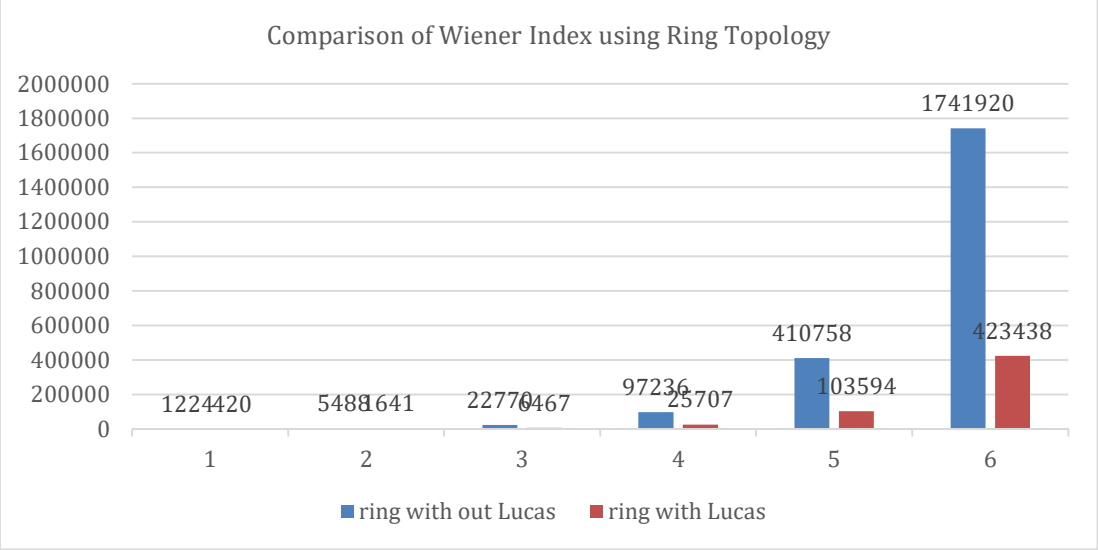

Fig. 10 Comparison of Wiener Index using Ring Topology with and without Lucas

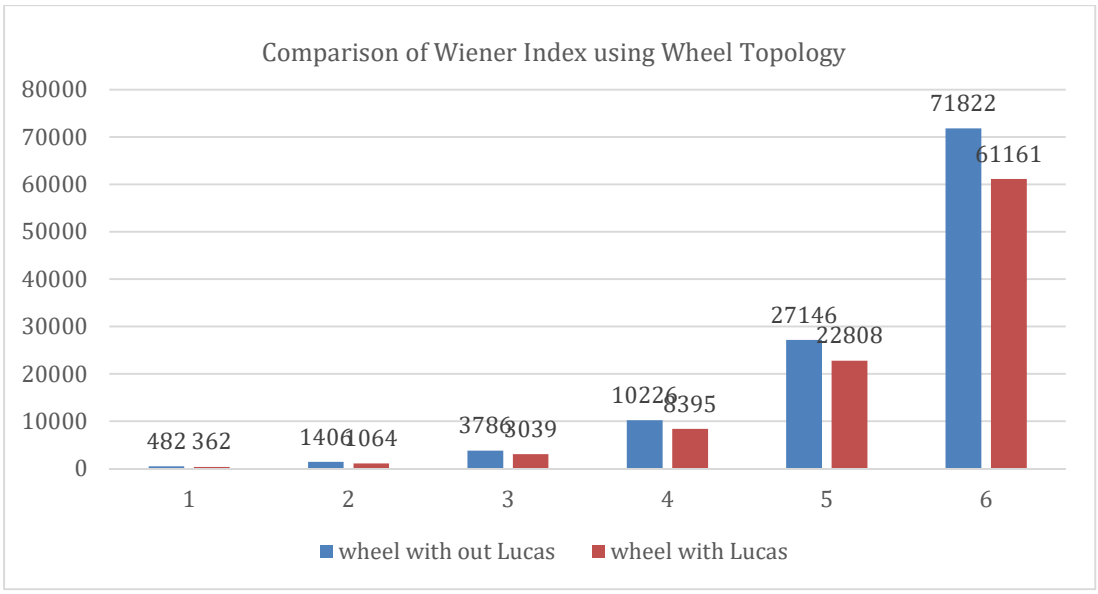

Fig. 11 Comparison of Wiener Index using Wheel Topology with and without Lucas

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82–98

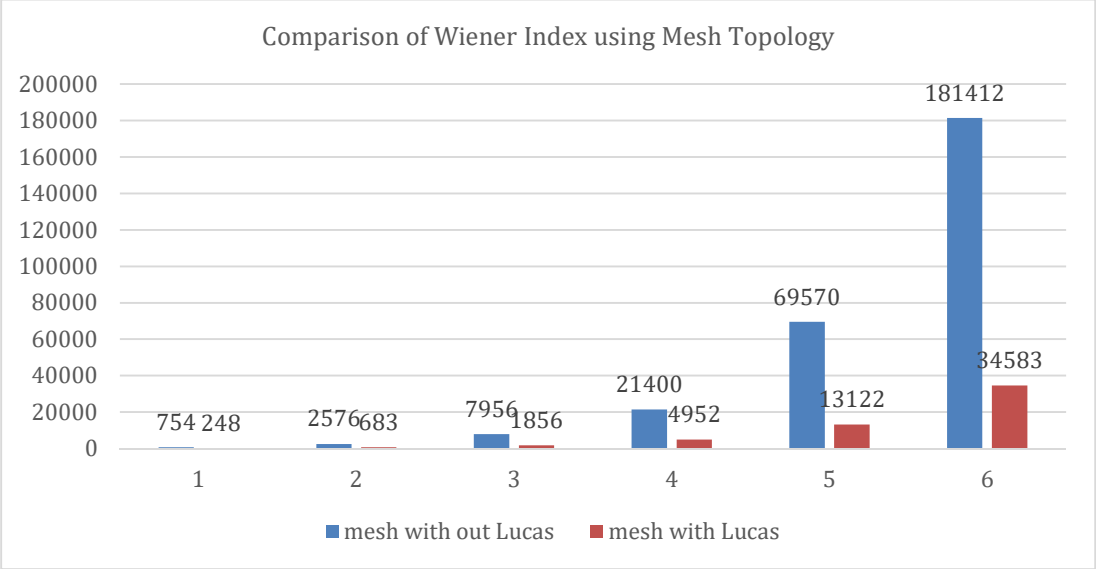

Fig. 12 Comparison of Wiener Index using Mesh Topology with and without Lucas

Comparison of Average Degree in Topologies with and without Lucas

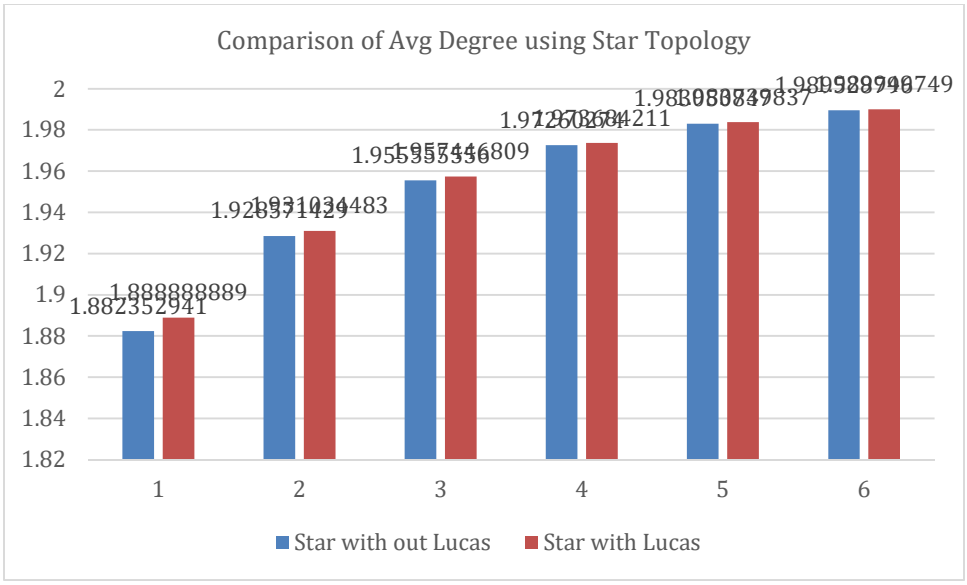

Fig. 13 Comparison of Average Degree using Star Topology with and without Lucas

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82–98

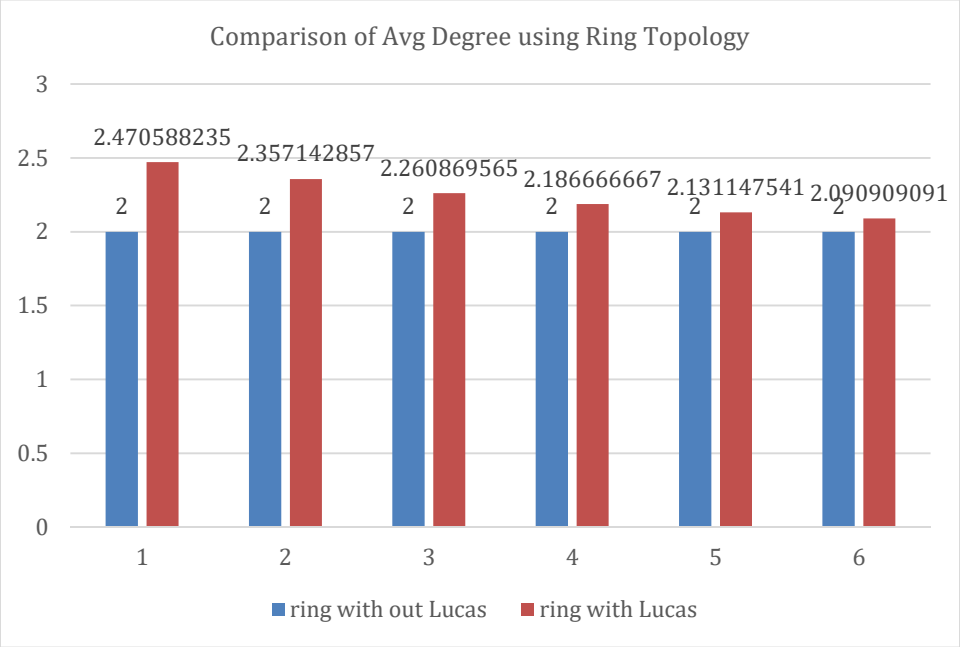

Fig. 14 Comparison of Average Degree using Ring Topology with and without Lucas

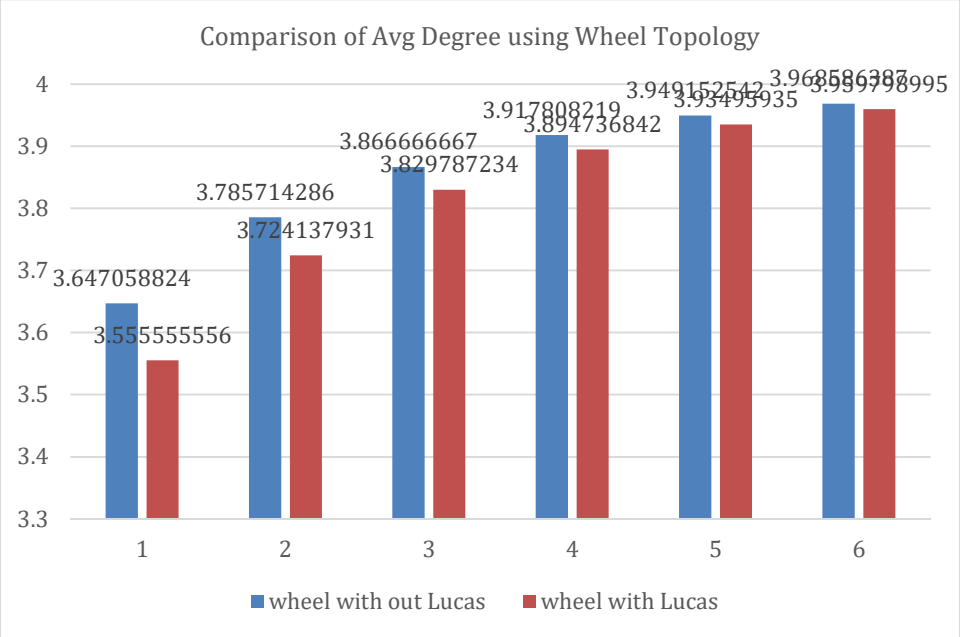

Fig. 15 Comparison of Average Degree using Wheel Topology with and without Lucas

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82–98

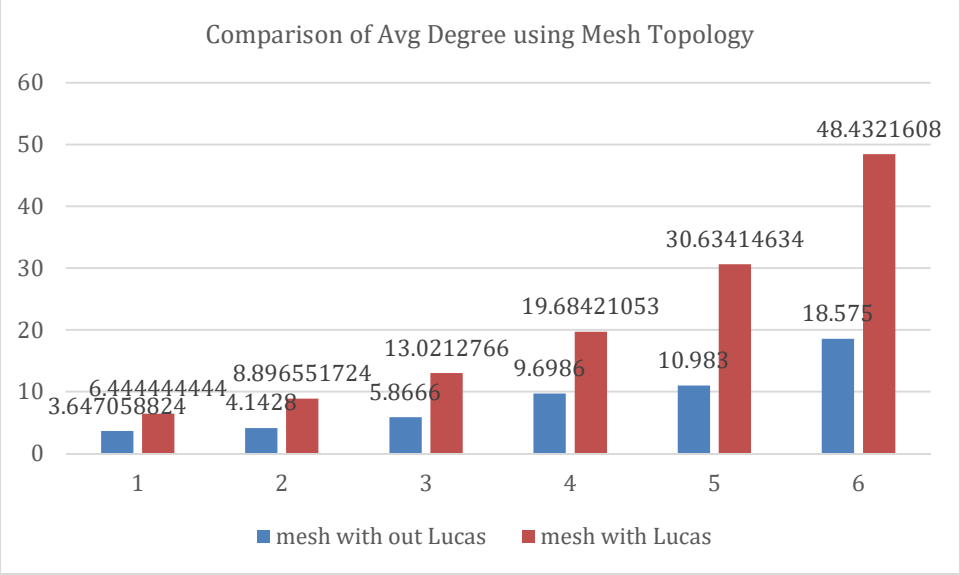

Fig. 16 Comparison of Average Degree using Mesh Topology with and without Lucas

Comparison of Average Path Length in Topologies with and without Lucas

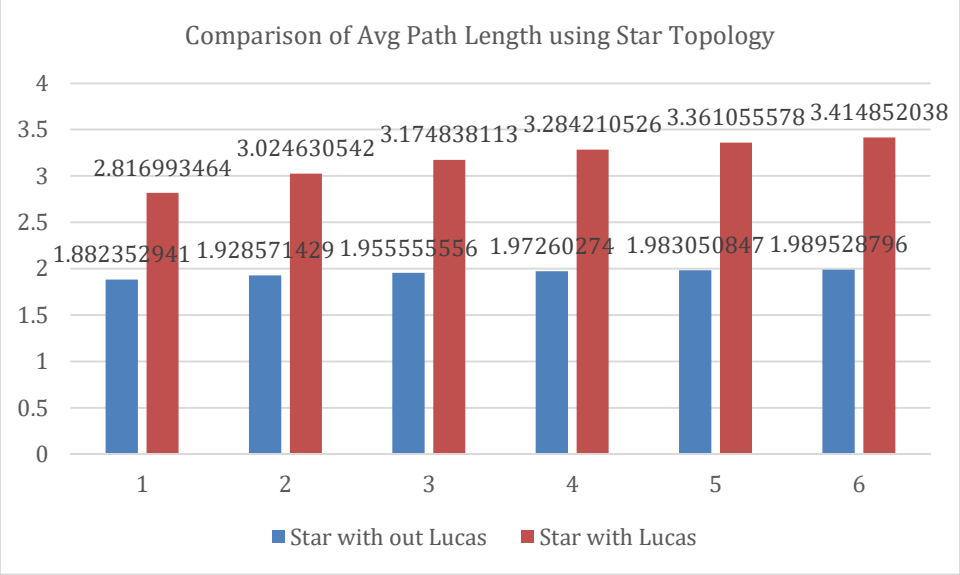

Fig. 17 Comparison of Average Path Length using Star Topology with and without Lucas

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82–98

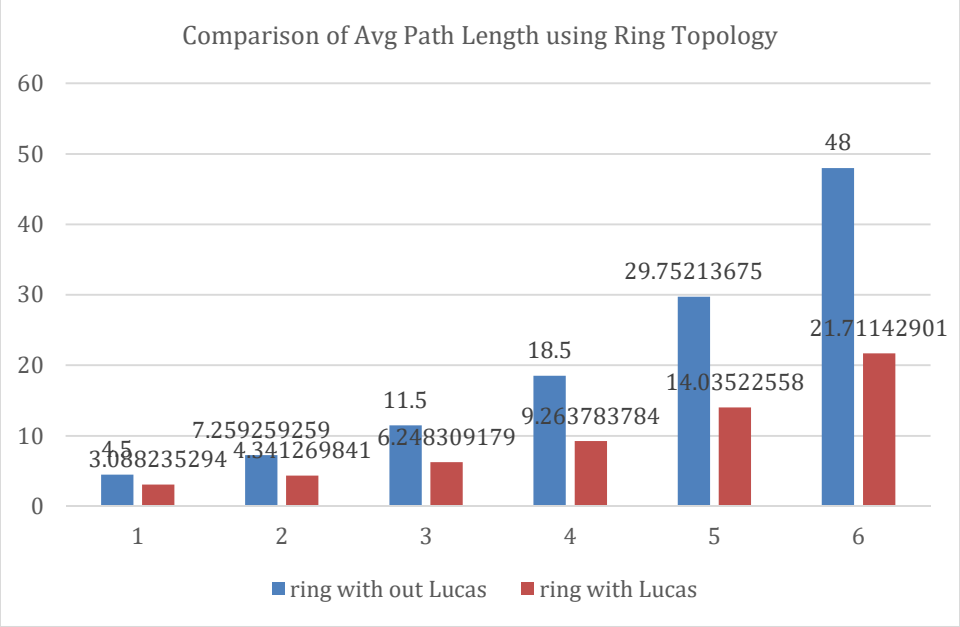

Fig. 18 Comparison of Average Path Length using Ring Topology with and without Lucas

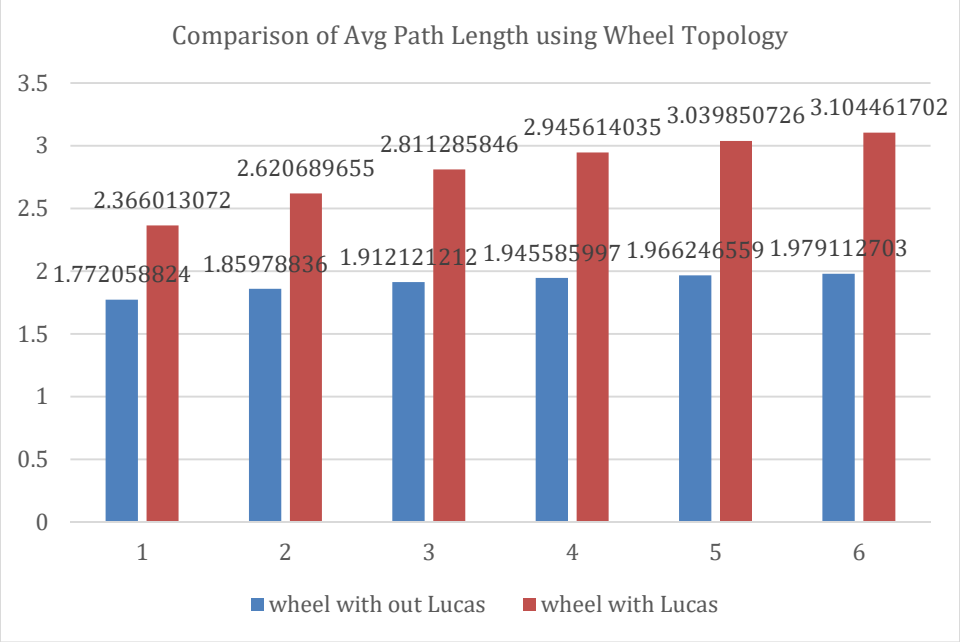

Fig. 19 Comparison of Average Path Length using Wheel Topology with and without Lucas

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82–98

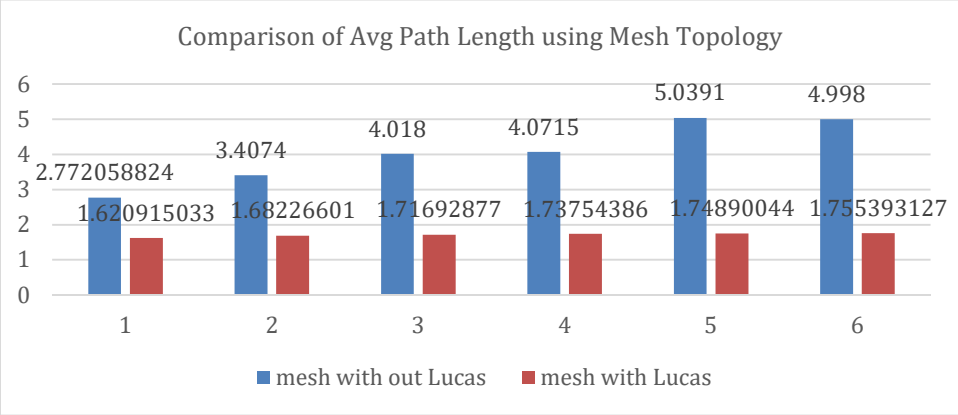

Fig. 20 Comparison of Average Path Length using Mesh Topology with and without Lucas

4. Conclusion

This paper introduces a comparative analysis of star, wheel, ring, and complete graph architectures where the network analysis is done through different parameter. These comparison shows that complete graph or mesh topology have a better network connectivity which can be seen through by seen one of the parameters i.e. wiener index whose values determine the better connectivity of a topology when compare to other topology. Future research will aim to integrate this strategy with parameter like power consumption that solves the central problem of energy-efficient network lifetime maximization in power-limited environments. By utilizing the inherent properties of Lucas sequences, the introduced approach describes an adaptive yet efficient framework for power-aware sampling rate adaptation

5. Future Work

Future studies on network topologies with Lucas configurations can explore a number of promising directions. AI integration for dynamic optimization is one potential strategy. Machine learning models are used to predict traffic patterns and adjust network configurations in real-time for increased efficiency. Furthermore, utilizing game theory can improve network resilience by fine-tuning connections according to node advantages. Genetic algorithms can also be improved to address intricate optimization problems more efficiently. Machine learning can also be used to find novel network topologies that offer improved performance. Testing these theoretical optimizations in real-world situations is essential for gaining practical understanding. . Further investigation of the mathematical properties of Lucas numbers can enhance theoretical foundation and practical implementation in real-world smart homes can validate effectiveness and incorporate and user-centric design principle for improved usability.

The Journal of Computational Science and Engineering (TJCSE)
ISSN 2583-9055 (Media Online)
Vol 3, No 5, May 2025
PP 82–98

References

[1] Rajana, K. R., & Amiripalli, S. S. (2025). A Novel Lucas-based Clustering Optimization for Enhancing Survivability in Smart Home Design. Engineering, Technology & Applied Science Research, 15(1), 19903-19909.
[2] Banoun, D. O., Boyle, E., & Cohen, R. (2024). Information-Theoretic Topology-Hiding Broadcast: Wheels, Stars, Friendship, and Beyond. Cryptology ePrint Archive.
[3] Jagtap, S. N., Potharaju, S., Tambe, S. B., & Srinivas Amiripalli, S. (2024). Hybrid Ensemble Feature Selection Using Symmetrical Uncertainty and Multi-Layer Perceptron. International Journal of Computing and Digital Systems, 17(1), 1-10.
[4] Daousis, S., Peladarinos, N., Cheimaras, V., Papageorgas, P., Piromalis, D. D., & Munteanu, R. A. (2024). Overview of protocols and standards for wireless sensor networks in critical infrastructures. Future Internet, 16(1), 33.
[5] Song, Z., & Zhang, H. (2024). Resilient Fiber–Wireless Networks Featuring Scalability and Low Latency: Integrating a wheel-and-star architecture with wireless protection. IEEE Access.
[6] Jouhari, M., Saeed, N., Alouini, M. S., & Amhoud, E. M. (2023). A survey on scalable LoRaWAN for massive IoT: Recent advances, potentials, and challenges. IEEE Communications Surveys & Tutorials, 25(3), 1841-1876.
[7] Lorincz, J., Klarin, Z., & Begusic, D. (2023). Advances in improving energy efficiency of fiber–wireless access networks: a comprehensive overview. Sensors, 23(4), 2239.
[8] Kirmani, S., Mazid, A., Khan, I. A., & Abid, M. (2022). A survey on IoT-enabled smart grids: technologies, architectures, applications, and challenges. Sustainability, 15(1), 717.
[9] Toma, C. I., Popescu, M., Constantinescu, M. D., & Popa, I. C. (2022). Application of Electrical Substation, Ring Topology versus Star Topology. Ann. Univ. Craiova Electr. Eng. Ser, 46(46), 63-68.
[10] Karakoc, D. B., & Konar, M. (2021). A complex network framework for the efficiency and resilience trade-off in global food trade. Environmental Research Letters, 16(10), 105003.
[11] Kanakaraju, R., Lakshmi, V., Amiripalli, S. S., Potharaju, S. P., & Chandrasekhar, R. (2021). An Image Encryption Technique Based on Logistic Sine Map and an Encrypted Image Retrieval Using DCT Frequency. In Recent Trends in Intensive Computing (pp. 1-8). IOS Press.
[12] Nelson, J., Andoh, C., Comia, A., Echeveria, L., Hopkins, J., Maniti, M., & Pierce, T. (2021, January). Wireless sensor network with mesh topology for carbon dioxide monitoring in a winery. In 2021 IEEE Topical Conference on Wireless Sensors and Sensor Networks (WiSNeT) (pp. 30-33). IEEE.
[13] Nurlan, Z., Zhukabayeva, T., Othman, M., Adamova, A., & Zhakiyev, N. (2021). Wireless sensor network as a mesh: Vision and challenges. IEEE access, 10, 46-67.
[14] Ompal, Mishra, V. M., & Kumar, A. (2021). Zigbee internode communication and FPGA synthesis using mesh, star and cluster tree topological chip. Wireless Personal Communications, 119(2), 1321-1339.
[15] Oroza, C. A., Giraldo, J. A., Parvania, M., & Watteyne, T. (2021). Wireless-sensor network topology optimization in complex terrain: A bayesian approach. IEEE Internet of Things Journal, 8(24), 17429-17435.
[16] Al-Hamadi, H., Saoud, M., Chen, R., & Cho, J. H. (2020). Optimizing the lifetime of IoT-based star and mesh networks. IEEE Access, 8, 63090-63105.
[17] Romanov, A. Y. (2019). Development of routing algorithms in networks-on-chip based on ring circulant topologies. Heliyon, 5(4).
[18] Ezran, P., Haddad, Y., & Debbah, M. (2017). Availability optimization in a ring-based network topology. Computer Networks, 124, 27-32.
[19] Agnihotri, M., Chirikov, R., Militano, F., & Cavdar, C. (2016, April). Topology formation in mesh networks considering role suitability. In 2016 IEEE Wireless Communications and Networking Conference (pp. 1-7). IEEE.
[20] Noureddine, A., Brahim, A., & Zitouni, A. (2016). HMSR: A Hybrid Network-on-Chip Topology Synthesis Based on Star-Ring-Mesh at System Level. International Journal of Computer Science and Information Security, 14(9), 264.
[21] Singh, H., Singh, S., & Malhotra, R. (2013). Modeling, Evaluation and Analysis of Ring Topology for Computer Applications Using Simulation. International Journal of Computer Science and Mobile Computing, 2(1), 1-3.
[22] Kim, J. Y., Park, J., Lee, S., Kim, M., Oh, J., & Yoo, H. J. (2010). A 118.4 gb/s multi-casting network-on-chip with hierarchical star-ring combined topology for real-time object recognition. IEEE Journal of Solid-State Circuits, 45(7), 1399-1409.
[23] Labbé, M., Laporte, G., Martín, I. R., & González, J. J. S. (2004). The ring star problem: Polyhedral analysis and exact algorithm. Networks: An International Journal, 43(3), 177-189.