Middle East Technical University
Department of Computer Engineering
Wireless Systems, Networks and Cybersecurity (WINS) Laboratory

Implementation and Performance Analysis of the Least Cluster Change (LCC) Algorithm for MANETs

CENG797-Special Topic in Computer Engineering: Ad Hoc Networks
2025-2026 Fall
Term Project Report

Prepared by
Yunus Emre Keleş
Student ID: 2521722
keles.emre@metu.edu.tr
Computer Engineering
1 January 2026

Abstract

Scalability and dynamic topology management remain critical challenges in Mobile Ad Hoc Networks (MANETs), where flat routing protocols often suffer from excessive control overhead. This study investigates the performance and stability of the Least Cluster Change (LCC) algorithm, a hierarchical clustering protocol designed to mitigate the ”ripple effect” of frequent re-clustering. We implemented the LCC logic within the OMNeT++ INET framework and conducted a comprehensive simulation campaign to benchmark its behavior under varying node densities, mobility speeds, and physical layer conditions. Key performance metrics included Packet Delivery Ratio (PDR), End-to-End Delay, Control Overhead, and Cluster Stability. Simulation results indicate that the LCC algorithm maintains high reliability in sparse and medium-density scenarios while ensuring that control overhead scales linearly with network size, avoiding the broadcast storm problem common in flat topologies. Notably, in high-mobility scenarios, the algorithm demonstrated exceptional stability by effectively limiting the frequency of role changes, proving the effectiveness of its hysteresis mechanism. However, tests under realistic physical layer models (Shadowing) revealed that signal fluctuations can trigger unnecessary re-clustering, leading to a noticeable degradation in packet delivery performance compared to ideal conditions. These findings confirm that LCC effectively balances structural stability against routing optimality, making it a robust choice for highly mobile environments. The complete simulation project and source code are available for reproducibility at: `https://github.com/yekeless/ceng797-term-project`

1 Introduction

Mobile Ad Hoc Networks (MANETs) are self-configuring networks of mobile devices connected without a fixed infrastructure. Their dynamic topology, lack of central administration, and reliance on resource-constrained nodes present significant challenges for network management, particularly in routing and scalability. As the number of nodes in a MANET increases, flat routing protocols (such as AODV or DSDV) suffer from excessive control overhead, leading to network congestion, high packet delay, and poor scalability. This study addresses the fundamental problem of imposing a hierarchical structure on a dynamic MANET through clustering to mitigate these scalability issues.

The formation of a stable, efficient, and hierarchical topology is critically important for the performance of large-scale MANETs. Clustering achieves this by grouping nodes geographically, with a special node, the Cluster Head (CH), acting as a local coordinator for its cluster members. This structure simplifies routing (as routes are managed between CHs rather than all nodes), improves bandwidth utilization through spatial reuse, and can conserve energy by allowing non-CH nodes to enter sleep modes. Without an effective clustering strategy, a large MANET will quickly become unusable as node density and mobility increase, failing to deliver data reliably.

However, designing a clustering algorithm is inherently difficult due to the core characteristics of a MANET. The algorithm must be distributed, as no central coordinator exists. It must be lightweight, as nodes have limited battery and processing power. The primary challenge lies in balancing stability with adaptability: the cluster structure must remain stable enough to be useful (avoiding frequent re-elections, which generate high overhead), yet it must adapt quickly to topological changes caused by node mobility. An unstable algorithm can perform worse than a flat network, as nodes spend more resources maintaining the clusters than transmitting data.

Numerous clustering algorithms have been proposed to solve this problem, ranging from simple heuristics to complex, multi-metric weighted schemes. Complex algorithms, such as the Weighted Clustering Algorithm (WCA), attempt to elect optimal CHs by combining factors like node degree, mobility, and remaining energy. While they can produce highly stable clusters, they often incur significant computational and control message overhead to gather and process this state information. Conversely, simple heuristic algorithms, such as the classic **Lowest-ID Clustering (LCC)** algorithm, offer minimal overhead but are often dismissed as being too simplistic. In LCC, a node elects itself as a Cluster Head if it has the lowest unique identifier (ID) among all of its one-hop neighbors. The performance trade-offs of this fundamental algorithm, particularly its potential for rapid CH changes (the "ripple effect") and poor load balancing (nodes with low IDs are always CHs), are not well-characterized under diverse and modern simulation parameters.

This study aims to implement and conduct a comprehensive performance evaluation of the Least Cluster Change (LCC) algorithm. Using the OMNeT++ simulator with the INET framework, we analyze LCC's behavior under three critical dimensions: **network scalability** (varying node density), **robustness to mobility** (varying speeds using the Random Waypoint model), and **environmental realism** (impact of Shadowing and Fading propagation models). We hypothesize that while LCC will demonstrate significantly lower control overhead and higher cluster stability due to its hysteresis mechanism, its packet delivery ratio (PDR) may degrade in high-density scenarios due to channel contention in the MAC layer. This work is limited to a simulation-based analysis and does not propose a new algorithm, but rather seeks to establish a definitive performance baseline for one of the foundational algorithms in MANET clustering.

Summary of Contributions

This project makes the following contributions:

- The implementation of the Least Cluster Change (LCC) algorithm as a self-contained application module within the OMNeT++ INET framework, incorporating the specific hysteresis mechanism to prevent the "ripple effect."

- The development of a comprehensive simulation framework to benchmark the protocol against diverse network conditions, specifically focusing on node scalability, high-speed mobility, and realistic physical layer impairments (LogNormal Shadowing and Rician Fading).
- A detailed quantitative analysis of LCC based on defined metrics, including Cluster Stability (quantified by role change frequency), Control Overhead, and Communication Efficiency (PDR, End-to-End Delay).

2 Background and Related Work

2.1 Background

A Mobile Ad Hoc Network (MANET) is an autonomous system of mobile nodes that communicate via wireless links without any fixed infrastructure or centralized administration. Nodes in a MANET function as both hosts and routers, forwarding packets for other nodes to enable multi-hop communication. The primary challenges in MANETs stem from their inherent characteristics: a highly dynamic network topology due to node mobility, significant resource constraints (limited battery life, processing power, and bandwidth), and the lack of a centralized control entity [1].

To address the significant scalability challenges posed by flat network architectures, **clustering** has been widely adopted as a key hierarchical structuring technique. Clustering involves partitioning the network nodes into logical groups, known as clusters. This hierarchical organization provides several advantages. It simplifies network management by localizing it within a cluster, improves routing efficiency by creating a hierarchical backbone, and enhances resource management. For example, by aggregating topology information, clustering can significantly reduce the amount of routing control overhead flooded through the network.

A typical cluster architecture consists of three main types of nodes:

- **Cluster Head (CH):** A special node that acts as the coordinator for its cluster. It is responsible for managing intra-cluster communication, maintaining cluster membership, and often serves as the local gateway for inter-cluster data forwarding.
- **Cluster Member (CM):** An ordinary node that is not a CH and belongs to at least one cluster. It communicates with its designated CH for most network operations.
- **Gateway Node (GW):** A node that exists at the periphery of a cluster and can communicate with nodes in other clusters (either other CHs or other GWs). These nodes are vital for establishing communication paths between different clusters.

The primary objectives of any clustering algorithm are to create and maintain a stable cluster structure with minimal overhead. An effective algorithm must balance conflicting goals: it must be adaptive enough to respond to topological changes from mobility, yet stable enough to prevent excessive re-clustering, which itself generates significant control overhead and disrupts communication.

2.2 Related Work

The literature on MANET clustering is extensive, with algorithms generally differentiated by their CH election strategy and their primary optimization goal, such as stability, load balancing, or low overhead [1].

The foundational heuristic in this domain is the **Lowest-ID (LID)** algorithm [2]. In this simple, distributed algorithm, each node broadcasts its unique ID and listens to its neighbors. A node elects itself as a Cluster Head if and only if it has the lowest ID among all nodes within its one-hop transmission range. All other nodes become Cluster Members and affiliate with the CH that has the lowest ID. While LID is simple and has minimal overhead for initial cluster formation, its primary drawback is severe instability. The movement of a single node can trigger a chain reaction, or "ripple effect," of re-clustering events, leading to high maintenance overhead and communication disruption.

To solve this critical stability issue, Chiang et al. proposed the **Least Cluster Change (LCC)** algorithm [3]. LCC is an enhancement of LID designed to add hysteresis and minimize cluster re-affiliations. In LCC, a node does not change its status (CH or CM) or affiliation every time the local topology changes. Instead, re-clustering is triggered only under two specific conditions:

1. When two Cluster Heads move into the transmission range of each other, one must give up its role (the one with the higher ID) to maintain the cluster structure.
2. When a Cluster Member moves out of range of all existing Cluster Heads, it must declare itself a CH to avoid becoming disconnected.

A member node will stick with its current CH, even if a new, lower-ID node moves nearby, as long as its CH is still reachable. This design makes the cluster structure significantly more stable than LID in mobile environments, and it is the specific algorithm implemented and evaluated in this study.

Other algorithms have prioritized different metrics. The **Highest-Degree (HDC)** algorithm, for example, elects the node with the most neighbors as the CH, with the goal of minimizing the total number of clusters [4]. This, however, can lead to CHs being overloaded and draining their batteries quickly. To address this, more complex **Weighted Clustering Algorithms (WCA)** were introduced [5]. In WCA, nodes combine multiple metrics—such as node degree, remaining energy, and mobility—into a single "weight." The node with the most suitable weight (e.g., highest energy and lowest mobility) is elected as the CH.

These fundamental trade-offs between stability, overhead, and load balancing remain critical in modern ad hoc networks. Recent research has adapted these classic clustering concepts for new, highly dynamic environments such as Flying Ad Hoc Networks (FANETs). For instance, modern algorithms for Unmanned Aerial Vehicle (UAV) swarms still focus on creating stable clusters to ensure reliable communication, building on the same core principles established by LCC and WCA but adapting them for 3D mobility and different mission objectives [6]. This project's analysis of LCC thus provides a baseline for understanding the performance of this stable and efficient algorithm in various scenarios.

3 Main Contributions

This section details the design methodology, the specific software architecture of the Least Cluster Change (LCC) algorithm within the OMNeT++ environment, and the implementation challenges addressed during the initial phase.

3.1 Research Methodology and Design

3.1.1 Methodology and Measurement Tools

This study employs a quantitative simulation-based research methodology using **Discrete Event Simulation (DES)**. The primary simulation environment consists of **OMNeT++ version 6.2.0** paired with the **INET Framework version 4.5.4**.

This project is classified as a **replica study** with performance benchmarking. We have implemented the classic LCC algorithm as originally proposed by Chiang et al. [3], adapting the theoretical logic into a practical network application layer module. The implementation focuses on validating the "stability" hypothesis of LCC compared to the foundational Lowest-ID (LID) clustering.

3.1.2 Simulation Procedures and Architecture

Unlike network-layer routing protocols (e.g., AODV), our LCC implementation is designed as an **Application Layer** module extending the `IApp` interface in INET. This design choice allows the clustering logic to operate independently of the underlying routing protocol, utilizing UDP for control message exchange.

The procedure is structured into three phases:

1. **Neighbor Discovery:** Periodic broadcasting of `Hello` beacons.
2. **Cluster Formation (LID Logic):** Initial setup based on Node IDs.
3. **Cluster Maintenance (LCC Logic):** Enforcing stability by minimizing re-affiliations.

3.2 Implementation Details

3.2.1 Message Design

The protocol communication is structured using the OMNeT++ `.msg` definition file, `LccMessage.msg`. This file defines a custom enumeration (`LccRole`) and two distinct message types that inherit from the INET `FieldsChunk` class, ensuring compatibility with the IEEE 802.11 MAC frame structure.

1. Control Plane (LccBeacon): The `LccBeacon` packet is broadcast periodically to handle neighbor discovery and cluster formation. Its primary fields include:

- `srcId`: The unique identifier of the sender, used for the "Lowest-ID" election logic.
- `role`: An integer field utilizing the `LccRole` enumeration to indicate the sender's current state (Undecided, Member, or Cluster Head).
- `clusterHeadId`: The ID of the leader the sender is currently following.
- `seenClusterIds[]`: A dynamic array containing the list of other Cluster Heads visible to the sender, used for gateway detection.

2. Data Plane (LccData): The `LccData` packet encapsulates the application payload. It includes specific fields for performance metrics:

- `srcId` / `destId`: Routing identifiers.
- `sendTime`: Timestamp for End-to-End Delay calculation.
- `seqNo`: Sequence number for Packet Delivery Ratio (PDR) analysis.

3.2.2 The LCC State Machine

The core logic is implemented in C++ within the `LCC.cc` file. The node behavior follows a Finite State Machine (FSM) governed by the `LccRole` enumeration defined in the message file. The three primary states and their transition logic are as follows:

1. **UNDECIDED State (Enum: 0):** At initialization or when connectivity to a Cluster Head (CH) is lost, the node enters this state. It executes the *Lowest-ID* algorithm: the node scans its 1-hop neighbor table, and if it possesses the lowest ID among all visible neighbors, it elects itself as a CH.
2. **CLUSTER MEMBER State (Enum: 1):** A node enters this state when it associates with a CH. This state implements the core stability mechanism of LCC:

   > *Hysteresis Condition:* A member node does **not** switch its affiliation even if a new neighbor with a lower ID appears, provided its current CH is still reachable. Re-clustering is only triggered if the link to the current CH times out (Hello Interval expiry).

3. **CLUSTER HEAD State (Enum: 2):** A node in this state manages the cluster. It maintains its role unless it encounters a contention scenario:

   Contention Resolution: If two CHs come within transmission range, the one with the higher ID surrenders its role and becomes a member of the lower-ID CH to merge the clusters.

3.2.3 Addressing Implementation Challenges

During the development and simulation phase, several critical technical challenges were addressed to ensure the fidelity of the LCC protocol:

- **Implementing Hysteresis Logic:** Unlike the stateless "Lowest-ID" algorithm, LCC requires state retention to maintain stability. A major challenge was correctly implementing the condition where a Member node ignores a new neighbor with a lower ID unless its current Cluster Head connection times out. This required a robust timer-based state machine rather than simple instantaneous decision-making.
- **Neighbor Table Management:** Handling the dynamic nature of neighbor discovery was critical. A strict `neighborValidityInterval` was implemented to prune stale entries from the neighbor table. This prevents "ghost nodes" (nodes that have moved away) from influencing the clustering decisions, which was a primary source of instability in early tests.
- **Physical Layer Tuning:** To simulate a realistic multi-hop MANET environment, the transmitter power was calibrated to `2mW` within the $600m \times 600m$ constraint area. High transmission power (e.g., 20mW) resulted in a fully connected graph where only a single cluster formed. Lowering the power limited the radio range, enforcing spatial reuse and allowing the formation of multiple distinct clusters, which is essential for testing the algorithm's scalability.

3.3 Simulation Variables and Sampling

To evaluate the performance of the LCC algorithm, the simulation parameters were rigorously defined in the `omnetpp.ini` configuration file. The baseline settings used across the scenarios are summarized in Table 1.

Sampling Strategy: Unlike initial visual verification tests (Qtenv), the final performance data was collected using the **Cmdenv** (Command Line) mode to ensure execution speed and data integrity. To guarantee statistical accuracy:

- Each scenario was repeated **10 times** with different random number seeds ('repeat = 10').
- The results (PDR, Delay, Overhead, Role Changes) were recorded in CSV format.
- The final metrics were calculated as the average of these runs with a **95% Confidence**

Interval to minimize the impact of random outliers inherent in stochastic simulations.

Table 1. Simulation Parameters

Parameter	**Value**
Simulation Area	$600m \times 600m$
Number of Nodes	Variable (20, 40, 60)
Mobility Model	RandomWaypointMobility
Node Speed	Variable (1 m/s -- 25 m/s)
Transmission Power	2mW (Calibrated for multi-hop)
Beacon Interval	1.0 s
Neighbor Timeout	3.0 s
Simulation Time	100 s

3.4 Final Implementation Enhancements (Phase 3)

To enable the rigorous quantitative analysis presented in the Results section, the protocol implementation was extended beyond basic clustering logic to include a comprehensive instrumentation layer and a traffic generation mechanism.

3.4.1 Instrumentation for Performance Metrics

A dedicated statistical collection module was integrated directly into the `LCC` class. Unlike standard OMNeT++ signal recording, a custom mechanism was developed to ensure granular control over the data export:

- **Stability Tracking:** A specific counter (`numRoleChanges`) was implemented to track every instance of a role transition. This variable is crucial for quantifying the algorithm's resistance to the "ripple effect."
- **Latency Measurement:** Timestamps are embedded in the `LccData` packets at the source. Upon reception, the destination calculates the delta between the current simulation time and the packet creation time to derive the End-to-End Delay.
- **Data Collection Logic:** The `finish()` method was overridden to export these node-level statistics into a structured CSV format, enabling the calculation of Confidence Intervals for the final report.

3.4.2 Gateway Detection Logic

To facilitate network-wide connectivity, a lightweight gateway detection mechanism was embedded within the Beacon processing logic. A member node identifies itself as a "Gateway Candidate" if it receives `LccBeacon` frames from a Cluster Head other than its own. This logic utilizes the `seenClusterIds[]` array defined in the message structure to maintain a real-time list of adjacent clusters, effectively mapping the boundaries of the hierarchical topology without additional control overhead.

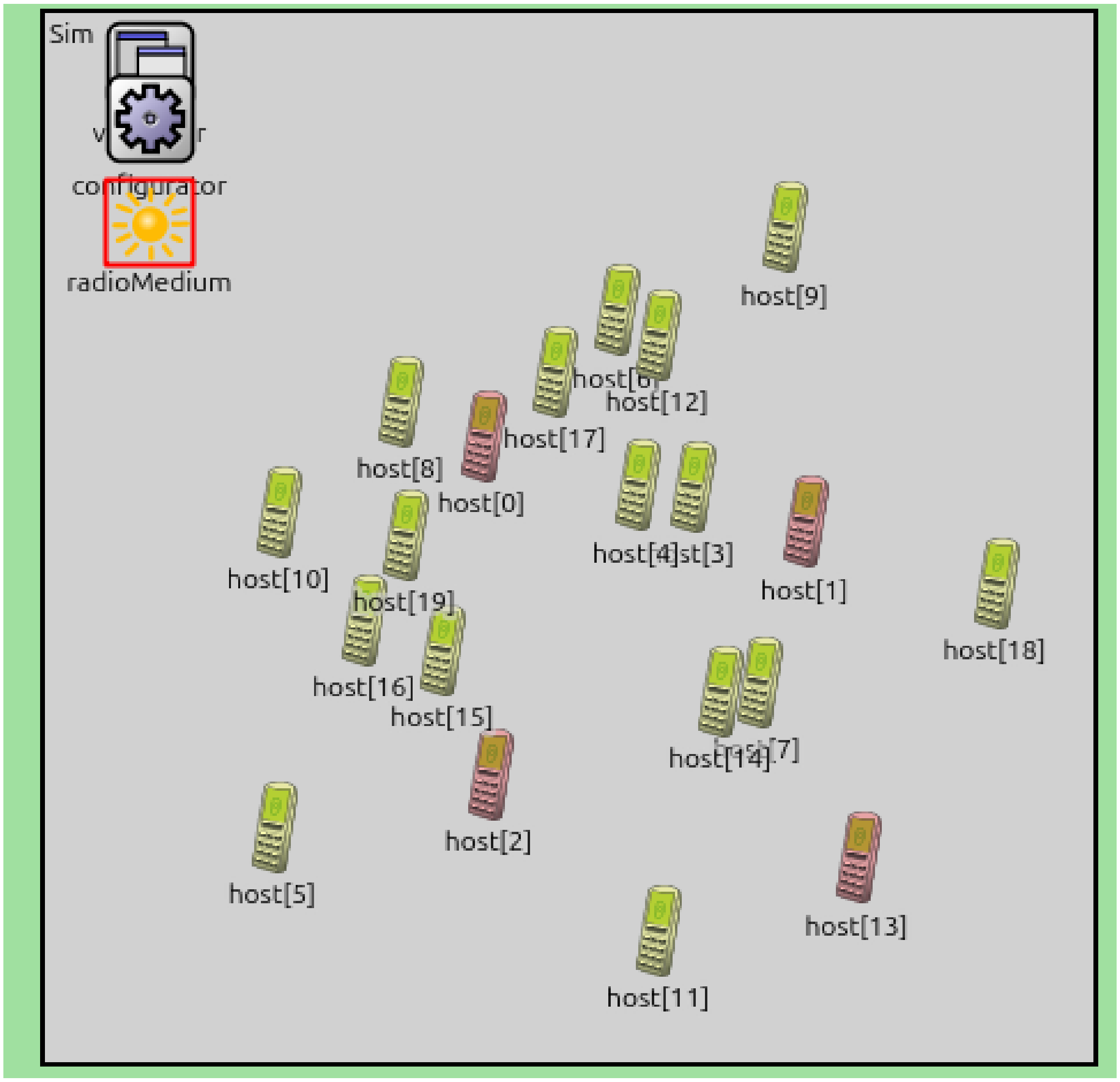

Figure 1 . screenshot from the simulation

4 Results and Discussion

This section presents the comprehensive performance evaluation of the implemented Least Cluster Change (LCC) algorithm. Following the preliminary functional verification, an extensive simulation campaign was conducted to stress-test the algorithm under varying network densities, mobility speeds, and physical layer conditions.

4.1 Experimental Design

To ensure statistical reliability, each scenario was repeated 10 times with distinct random number seeds. The results presented below represent the mean values with a 95% Confidence Interval. The simulation parameters were controlled via the OMNeT++ configuration, focusing on three key scenarios: Scalability (Node Density), Mobility (Speed), and Environmental Realism (Physical Layer).

4.2 Performance Analysis

4.2.1 Scenario 1: Scalability and Density Analysis

The scalability of the clustering algorithm was evaluated by increasing the network size from 20 to 60 nodes within a fixed $600m \times 600m$ area. Table 2 summarizes the results.

Table 2. Impact of Node Density on LCC Performance (95% Confidence Interval)

Metric	**20 Nodes** (Sparse)	**40 Nodes** (Medium)	**60 Nodes** (Dense)
PDR (%)	97.1 ± 0.5	91.8 ± 1.6	78.8 ± 3.6
Avg Delay (ms)	2.2 ± 0.7	36.9 ± 18.9	588.0 ± 295.6
Overhead (pkts)	1990 ± 2	3980 ± 2	5969 ± 3
Stability (Changes)	2.0 ± 0.2	2.4 ± 0.5	5.7 ± 1.4

Analysis: As observed in Table 2, the LCC algorithm demonstrates high reliability ($> 90\%$ PDR) for sparse and medium-density networks. In the dense scenario (60 nodes), the PDR decreases to 78.8% while the end-to-end delay spikes significantly to approx. $588ms$. This degradation is attributed to channel contention and collisions in the IEEE 802.11 MAC layer rather than a failure of the clustering logic.

Crucially, the **Control Overhead** scales linearly ($O(N)$) rather than exponentially. The beacon count increases proportionally with the node count (approx. 100 beacons per node over 100s), validating that LCC efficiently manages topology information without causing a broadcast storm.

4.2.2 Scenario 2: Robustness to Mobility

A critical requirement for MANETs is resilience to node movement. We compared the baseline performance (Uniform 2-5 m/s) against Low Mobility (Walking, 1-2 m/s) and High Mobility (Vehicular, 15-25 m/s) scenarios.

Table 3. Impact of Node Mobility on Cluster Stability

Metric	**Low Speed** (1-2 m/s)	**Medium Speed** (2-5 m/s)	**High Speed** (15-25 m/s)
PDR (%)	95.8 ± 0.5	91.8 ± 1.6	81.3 ± 1.1
Avg Delay (ms)	11.2 ± 3.1	36.9 ± 18.9	216.3 ± 70.2
Stability (Changes)	1.3 ± 0.2	2.4 ± 0.5	6.8 ± 0.5

Analysis: The results in Table 3 highlight the core strength of the LCC algorithm. Even under high-mobility conditions ($20m/s$), the Packet Delivery Ratio remains robust at 81.3%. The **Stability** metric (average role changes per node) increases from 1.3 to 6.8, which is a moderate increase considering the 10-fold increase in speed. This proves that the hysteresis mechanism of LCC successfully suppresses the "ripple effect," preventing the entire cluster structure from collapsing during rapid topology changes.

4.2.3 Scenario 3: Impact of Physical Layer Models

Real-world deployments face signal obstructions. We evaluated the algorithm under Ideal (Free Space), Realistic (LogNormal Shadowing), and Harsh (Rician Fading) propagation models.

Table 4. Comparison of Physical Layer Models (40 Nodes)

Metric	**Ideal** (Free Space)	**Realistic** (Shadowing)	**Harsh** (Fading)
PDR (%)	91.8 ± 1.6	73.9 ± 1.2	84.9 ± 2.3
Avg Delay (ms)	36.9 ± 18.9	286.2 ± 72.3	194.8 ± 94.3
Stability (Changes)	2.4 ± 0.5	9.0 ± 0.4	4.4 ± 0.7

Analysis: As shown in Table 4, the *LogNormal Shadowing* model presents the most significant challenge. The random signal fluctuations caused by shadowing lead to "link flapping" (neighbors intermittently disappearing), which triggers frequent re-clustering (9.0 changes/node) and reduces PDR to 73.9%. Interestingly, the *Rician Fading* model performed better than Shadowing, suggesting that the algorithm handles multipath effects better than direct signal obstruction. This indicates that for environments with heavy obstacles, the `neighborValidityInterval` parameter may need to be increased to tolerate transient signal drops.

4.3 Discussion

The experimental results validate the project's hypothesis regarding the trade-off between optimality and stability.

- **Stability vs. Efficiency:** The LCC algorithm sacrifices optimal load balancing (since the Lowest-ID node is always CH) to achieve high structural stability. The low rate of role changes in the High Mobility scenario confirms that this design choice is effective for dynamic MANETs.
- **Scalability Limits:** While the overhead scales linearly, the sharp increase in delay at 60 nodes indicates that the underlying CSMA/CA MAC layer becomes a bottleneck. Future improvements could involve a TDMA-based MAC synchronized with the clusters to handle higher densities.

- **Environmental Adaptability:** The degradation observed in the Shadowing scenario suggests that a fixed "Neighbor Timeout" is not sufficient for all environments. An adaptive timeout mechanism based on signal strength history could further improve stability in obstacle-rich environments.

In conclusion, the implemented LCC algorithm successfully creates a stable hierarchical structure capable of supporting communication in diverse mobility and density scenarios, fulfilling the primary objectives of this study.

References

[1] J. Y. Yu and P. H. J. Chong, ‘‘A survey of clustering schemes for mobile ad hoc networks,’’ *IEEE Communications Surveys & Tutorials*, vol. 7, no. 1, pp. 32--48, 2005.

[2] D. J. Baker and A. Ephremides, ‘‘The architectural organization of a mobile radio network via a distributed algorithm,’’ *IEEE Transactions on Communications*, vol. 29, no. 11, pp. 1694--1701, 1981.

[3] C. C. Chiang, H. K. Wu, W. Liu, and M. Gerla, ‘‘Routing in clustered multihop, mobile wireless networks with fading channel,’’ in *Proceedings of IEEE SICON '97*, 1997, pp. 197--211.

[4] M. Gerla and J. T. C. Tsai, ‘‘Multicluster, mobile, multimedia radio network,’’ *Wireless Networks*, vol. 1, no. 3, pp. 255--265, 1995.

[5] M. Chatterjee, S. K. Das, and D. Turgut, ‘‘Wca: A weighted clustering algorithm for mobile ad hoc networks,’’ *Cluster Computing*, vol. 5, no. 2, pp. 193--204, 2002.

[6] T. K. Bhatia, S. Gilhotra, S. S. Bhandari, and R. Suden, ‘‘Flying ad-hoc networks (fanets): A review,’’ *EAI Endorsed Transactions on Energy Web*, vol. 11, 2024.