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

Performance Evaluation of the d-Hop Clustering Algorithm in Mobile Ad Hoc Networks

CENG 797 Ad Hoc Networks
2025-2026 Fall
Term Project Report

Prepared by
Efe Kaan Fidancı
Student ID: e278620
kaan.fidanci@metu.edu.tr
Computer Engineering
14 December 2025

Table of Contents

List of Figures

List of Tables

1 Introduction

1.1 The Challenge of Maintaining Stable d-Hop Clusters in MANETs

Mobile Ad Hoc Networks (MANETs) rely on decentralized, self-configuring nodes that communicate over wireless links without fixed infrastructure. The absence of centralized control means that each node must independently manage routing, security, and resource allocation [3]. A key challenge in MANETs is maintaining efficient and stable clustering as the network size and mobility increase. This challenge is further compounded by dynamic topologies, where nodes frequently join or leave the network, requiring adaptive clustering algorithms [4]. To address this challenge, this study focuses on implementing and evaluating the Max-Min d-Cluster Algorithm , an established heuristic that aims to balance cluster stability and scalability.

1.2 Practical Importance of Efficient d-Hop Cluster Formation

Clustering—the grouping of nodes into clusters with designated cluster heads—is essential to improving MANET performance because it provides structural organization and supports scalable network management [5, 6]. An efficient clustering algorithm facilitates the formation of desirable and well-organized MANET topologies, enhancing network efficiency, reducing routing overhead, and improving overall communication performance [7]. Conversely, failing to address the clustering problem can result in inefficient and unstable network structures, leading to excessive overhead, uneven load distribution, and frequent topology disruptions. Therefore, solving this problem is a critical step toward achieving optimized, reliable, and sustainable MANET-based communication systems [8].

However, forming and maintaining stable clusters in MANETs is challenging due to node mobility, dynamic topologies, and decentralized control [9]. Simpler approaches, such as 1-hop, often fail to adjust to variations in network density or movement patterns [10]. In other words, they lack the adaptability required for the inherently dynamic nature of MANETs.

1.3 Objective: Assessing Max-Min d-Cluster Performance

This study aims to evaluate the performance of the Max-Min d-Cluster Algorithm in MANETs under varying network topologies, mobility patterns, traffic conditions, and energy constraints. Using OMNeT++ 6.2.0 and INET 4.5.4, the study will simulate network scenarios and measure the algorithm's effectiveness across cluster quality, overhead, communication efficiency, energy/resource usage, scalability, robustness, and fairness metrics.

1.4 Hypothesis: Stability and Efficiency Through Max-Min d-Clustering

It is hypothesized that the Max-Min d-Cluster Algorithm can maintain stable and efficient clusters across a wide range of network conditions, resulting in reduced control overhead, balanced energy consumption, and improved network performance compared to unclustered or poorly clustered MANETs.

1.5 Key Definitions for MANET Clustering Evaluation

- **MANET:** A self-configuring network of mobile nodes communicating without fixed infrastructure.

- **Clustering:** Grouping nodes into clusters with a cluster head to improve network management and performance.
- **d-Hop Clustering Algorithm:** A clustering method where nodes within d hops form a cluster, with one node elected as the cluster head. This study implements the Max-Min heuristic, which utilizes a 2d-round flooding mechanism for cluster head election.
- **Cluster Head:** A node responsible for intra-cluster coordination and inter-cluster communication.
- **Metrics:** Quantitative measures of network performance, including cluster quality, communication efficiency, energy usage, scalability, and fairness.

By systematically analyzing these metrics, this study will provide a comprehensive assessment of the d-Hop Clustering Algorithm's suitability for dynamic and large-scale MANET environments.

2 Background and Related Work

2.1 Background

MANETs are decentralized, infrastructure-less systems in which mobile nodes communicate through multi-hop wireless links without relying on fixed base stations or routers. Each node acts both as a host and a router, forwarding packets for others to maintain end-to-end connectivity [3]. Due to node mobility, wireless interference, and limited energy resources, MANETs experience frequent topology changes that make routing, scalability, and energy management particularly challenging [4].

Clustering has emerged as an effective mechanism to address these challenges by partitioning the network into manageable groups of nodes, known as clusters [5, 6]. Each cluster is coordinated by a Cluster Head (CH), which is responsible for intra-cluster communication—the exchange of data and control messages *within* the cluster among its member nodes. Inter-cluster communication, on the other hand, refers to the communication *between* different clusters, typically facilitated by gateway nodes that act as bridges between cluster heads. Clustering enhances scalability, reduces routing overhead, supports efficient resource allocation, and improves overall energy efficiency by organizing communication hierarchically [7]. However, maintaining stable clusters under mobility and varying network densities remains an open research problem.

To overcome these challenges, various clustering algorithms have been proposed, including Lowest-ID, Highest-Degree, and Weighted Clustering Algorithm (WCA) methods. While these approaches improve organization and connectivity, they often struggle with scalability as node density increases [11]. To address these limitations, multi-hop clustering strategies have gained attention for enhancing communication efficiency, network longevity, and load balancing, often combined with optimization, machine learning, and trust-based mechanisms to further improve energy efficiency and stability [12, 13, 14].

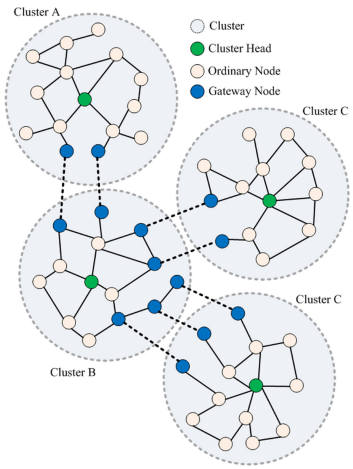

Figure 1 . Clustered Network Topology with Gateways [1].

The d-Hop Clustering Algorithm represents a significant advancement in MANET clustering. Unlike 1-hop clustering, where clusters are formed based on immediate neighbors, d-hop clustering allows nodes within d hops to form a cluster, with one node elected as the cluster head [2]. This approach balances cluster stability and scalability, making it suitable for dynamic and large-scale MANET environments.

The reviewed studies highlight continuous efforts to enhance the stability and scalability of d-hop clustering through mobility-awareness and adaptive cluster sizing.

2.2 Related Work

1. **Max-Min D-Cluster (foundational formal model):** Introduced by Amis et al. at INFOCOM 2000, this algorithm is a widely cited, efficient heuristic for forming d-hop clusters in distributed networks. It establishes a formal framework for clusterhead election, ensuring all nodes are within d hops of a clusterhead while balancing clusterhead selection. The approach serves as a baseline for both theoretical extensions and practical comparisons in d-hop clustering research [2].
2. **Mobility-aware metrics and their adoption:** Basu et al. introduced a relative mobility metric derived from successive received signal strength (RSS) measurements between neighboring nodes. This metric quantifies pairwise mobility and was later integrated into mobility-based d-hop clustering algorithms. The core principle is to prioritize nodes with lower relative mobility as clusterheads, enhancing overall cluster stability. This mobility-centric approach became a foundational element in subsequent mobility-sensitive clustering designs and analyses [15].
3. **MobDHop: adaptive d-hop clustering based on mobility:** Er and Seah introduced MobDHop, an algorithm that integrates d-hop clustering with Basu's mobility metric to create variable-diameter clusters. By adapting cluster formation to local mobility conditions, MobDHop enhances stability and minimizes reconfiguration overhead. This work highlights the potential of dynamically adjusting the d-hop diameter parameter, moving beyond static configurations. MobDHop stands as a key example of mobility-aware multi-hop clustering in MANET research [16].
4. **Performance studies and tradeoffs:** Subsequent research (e.g., Er 2006) examines the tradeoffs associated with increasing d: while a larger d reduces the number

of clusterheads and simplifies inter-cluster routing, it can also increase intra-cluster control overhead and make clusters more vulnerable to topological changes. Empirical and simulation-based evaluations—such as those of MobDHop and related algorithms—assess the balance between stability improvements and overhead costs. These studies also compare mobility-aware d-hop schemes with traditional 1-hop clustering approaches, providing critical insights for selecting optimal values of d and clusterhead election strategies based on mobility and traffic patterns [17].

5. **Recent distributed multi-hop methods:** Recent advancements, such as DC2HC (Distributed 2-Hop Clustering), revisit multi-hop intra-clustering to address modern challenges. These approaches prioritize fully distributed operation, reduced maintenance overhead, and seamless integration with connectivity and routing decisions. They demonstrate the enduring relevance of the d-hop concept, which has been refined to meet contemporary demands—such as scalability, reliability, and energy efficiency—in both MANET and VANET environments [18].

3 Main Contributions

3.1 Research Design and Methodology

This study employs a quantitative, simulation-based research methodology to evaluate the performance of the Max-Min d-Cluster algorithm in MANETs. The research is structured as an implementation and performance evaluation study. We are implementing a well-established, non-proprietary heuristic within a modern, standardized simulation environment to analyze its performance under a comprehensive set of conditions. This is an original implementation, but it replicates the logic proposed by Amis et al. (2000).

3.1.1 Measurement Tools

The primary tools for this study are OMNeT++ version 6.2.0 and the INET framework version 4.5.4.

- **OMNeT++** serves as the core discrete-event simulation engine, managing the simulation clock, message passing, and event scheduling.
- **INET** provides the extensive library of pre-built, validated models necessary for realistic MANETs, including network hosts, mobility models, various radio (PHY) and MAC protocols (like IEEE 802.11), and IP-level routing protocols (like AODV). Our Max-Min algorithm will be implemented as a new agent module that interacts with these INET components.

3.1.2 Experimental Procedure

The overall experimental procedure is structured into five distinct phases:

- **Implementation:** The Max-Min *d*-Cluster algorithm is implemented as a new simple module in INET, based on the logic described by Amis et al. .
- **Validation (Pilot Study):** The implementation's correctness is verified by replicating the paper's illustrative example.
- **Scenario Definition:** A comprehensive set of simulation campaigns is defined in `.ini` configuration files by varying the independent variables (e.g., node count, mobility model, `d` parameter).

- **Simulation Execution:** Each scenario is run multiple times with different random number seeds to ensure statistical validity.
- **Data Collection & Analysis:** Dependent variables (metrics) are recorded using OMNeT++'s signal and scalar mechanisms. The results are then processed to derive averages and 95% confidence intervals, as required by the project.

3.1.3 Pilot Study and Implementation Validation

To validate the correctness of our implementation, a pilot study was conducted by replicating the 25-node, $d = 3$ hop scenario described in the original Max-Min formulation. The simulation environment was configured with 25 stationary nodes placed at coordinates mimicking the topology shown in Figure 2. The simulation logs were captured and compared directly against the reference results provided by Amis et al.

The pilot study confirmed the correctness of the implementation. The algorithm successfully converged after *2d* rounds, and the elected Cluster Heads matched the reference exactly. This validates that the floodmax propagation, floodmin revocation, and the three-rule election logic are functioning correctly within the OMNeT++ environment. Notably, the selected Cluster Heads for non-head nodes also aligned with those proposed in the original paper, even though the visual representation in the OMNeT++ simulation does not explicitly highlight them.

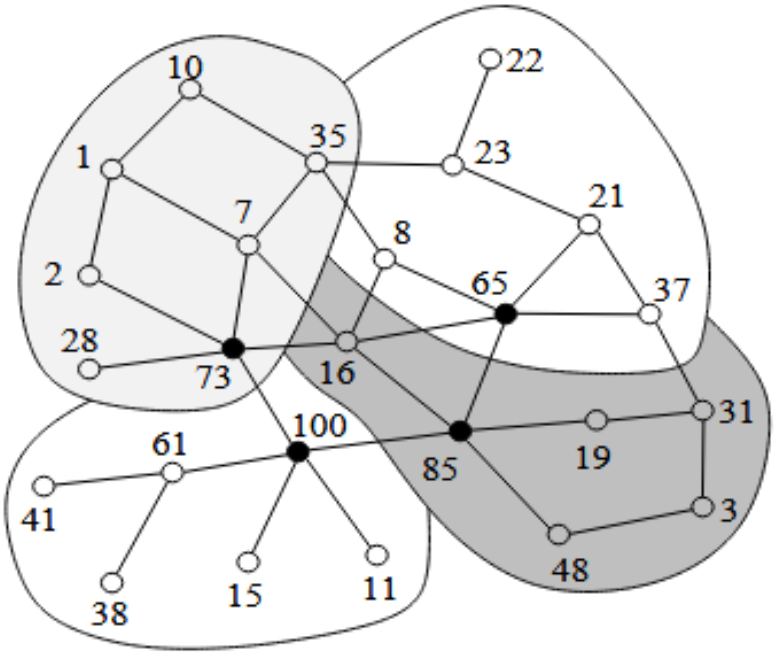

Figure 2 . 3-cluster Formation in a Network of 25 Nodes [2].

The simulation logs demonstrated a precise match with the reference table from the paper: nodes 65, 73, 85, and 100 were correctly elected as Cluster Heads.

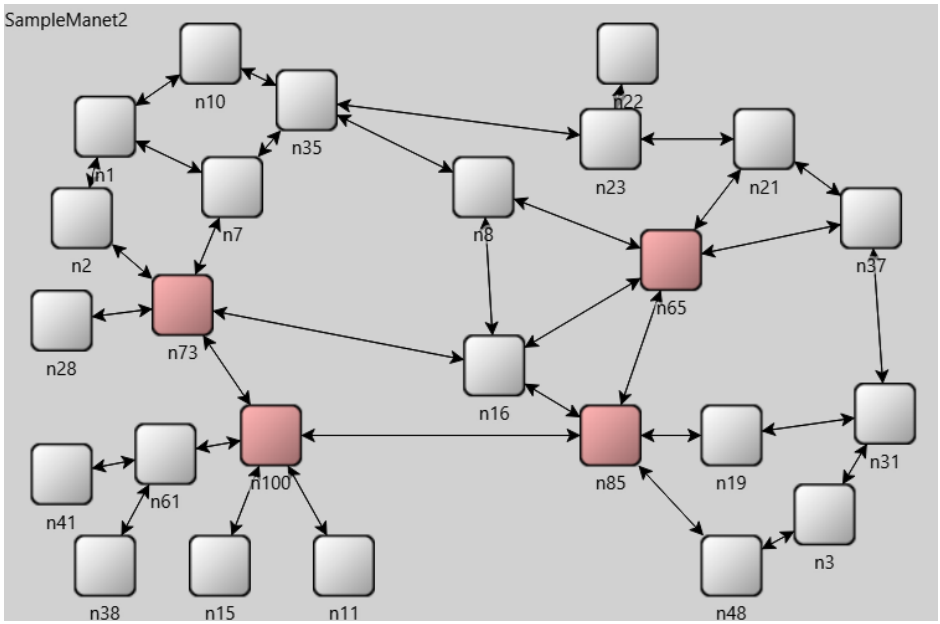

Figure 3 . Replicated 3-cluster topology in OMNeT++ simulation.

Node	10	1	2	7	35	8	23	22	21	65	37	31	19	85	16	100	73	28	41	61	11	48	3	15	38
Paper	73	73	73	73	73	65	65	65	65	65	65	85	100	85	100	100	73	100	100	100	100	100	100	100	100
Simulation	73	73	73	73	73	65	65	65	65	65	65	85	100	85	100	100	73	100	100	100	100	100	100	100	100

Table 1. Max-Min Cluster Head Validation: Reference vs. OMNeT++ Simulation.

3.1.4 Study Variables

The variables for this study are broken into two categories: independent (what we change) and dependent (what we measure).

These are the parameters we will systematically vary in the `.ini` files to observe their effect on the network.

- **Independent Variables:** Mobility models, network size (number of nodes), routing protocols (AODV, DSDV), MAC layer protocols, and the d parameter of the algorithm (e.g., 2, 3).
- **Dependent Variables:**
  - Cluster Quality (Number of Clusters, Cluster Head Lifetime),
  - Overhead (Control Messages),
  - Communication Efficiency (Throughput, PDR),
  - Energy (Network Lifetime).

3.1.5 Sampling and Statistical Analysis

Sampling is performed by repeating each simulation scenario multiple times (e.g., 30-50 runs) using different, independent random number seeds. This method ensures that the results are statistically significant and allows for the calculation of 95% confidence intervals, as required by the project.

3.2 Max-Min d-Cluster Algorithm Design and Implementation

The algorithm implemented in this project is the Max-Min d-Cluster heuristic proposed by Amis et al. (2020). This algorithm is designed to asynchronously form d-hop clusters, where every node is guaranteed to be at most d hops from a cluster head. The heuristic's goals include stability (favoring re-election of existing cluster heads) and fairness (distributing the cluster head load).

The algorithm's core logic is divided into three primary stages and runs for a total of $2d$ communication rounds. Each node maintains two arrays of size $2d$: `WINNER` and `SENDER`.

1. **Stage 1: Floodmax (d rounds)**
    - Initially, each node sets its `WINNER` to its own node ID.
    - For d rounds, each node broadcasts its current `WINNER` value to its 1-hop neighbors.
    - After receiving messages from its neighbors in a round, the node selects the largest value among its own `WINNER` and all received `WINNER`s as its new `WINNER`.
    - The `WINNER` for each round is logged in the `WINNER` array.
2. **Stage 2: Floodmin (d rounds)**
    - This stage immediately follows Floodmax and also lasts for d rounds.
    - The process is identical to Floodmax, but the node chooses the smallest value as its new `WINNER`.
    - The `WINNER` for each of these rounds is also logged.
3. **Stage 3: Cluster Head Selection (Rules)**
    - After $2d$ rounds are complete, each node evaluates its `WINNER` log to determine its cluster head. The decision is made by applying the following rules in order:
        (a) **Rule 1:** A node checks if its own ID appears in the log for the second d rounds (the Floodmin stage). If it does, the node declares itself a cluster head.
        (b) **Rule 2:** If Rule 1 is false, the node identifies ‘‘node pairs’’---any node ID that appears at least once in both the Floodmax log and the Floodmin log. The node then selects the minimum node ID from this set of pairs as its cluster head.
        (c) **Rule 3:** If Rule 1 is false and no node pairs exist (Rule 2 fails), the node selects the `WINNER` from the final round of the Floodmax stage (i.e., the value at round d) as its cluster head.

The Max-Min d-Cluster heuristic is implemented within a custom OMNeT++ simple module, MaxMinNode, which extends cSimpleModule. The core algorithmic rules follow the theoretical design described in Section 3.1.

4 Results and Discussion

4.1 Methodology

The methodology for this study is designed to ensure that the evaluation of the Max–Min d-Cluster Algorithm can be independently reproduced, verified, and critically examined. To achieve this, the research is conducted within a controlled and well-defined simulation environment, and each step—ranging from data generation to statistical analysis—is documented with precision.

The experiments are carried out entirely in OMNeT++ 6.2.0 using the INET 4.5.4 framework, which together provide the computational infrastructure, mobility models, wireless channel models, and protocol stacks required for realistic MANET simulations. The Max–Min algorithm is implemented as a custom simple module that integrates directly into INET's node architecture. This controlled environment ensures that all experimental conditions, parameters, and events can be reproduced faithfully by other researchers.

4.1.1 Materials and Simulation Environment

The "materials" of this study consist of:

- The OMNeT++ discrete-event simulator
- The INET framework (mobility, PHY/MAC, routing modules)
- The custom MaxMinNode C++ implementation
- Simulation configuration files `.ini` that define network size, mobility behaviors, protocol settings, and random number seeds

4.1.2 Data Generation and Sampling

Data is generated through simulation campaigns. Each scenario defined in the `.ini` files is executed multiple times with different random seeds. These seeds affect node placement, mobility patterns, wireless channel randomness, and message timing.

Repeating scenarios under randomized conditions provides a diverse dataset that captures the stochastic nature of MANETs and guards against biased single-run outcomes.

4.1.3 Statistical Treatment

To ensure that the results reflect typical system behavior rather than noise introduced by random mobility or wireless variability, all metrics are aggregated across multiple independent runs. For each metric, the study computes:

- The sample mean
- The sample variance
- The 95% confidence interval

4.1.4 Reproducibility

Every scenario can be fully reconstructed by:

1. Using the same OMNeT++ and INET versions,
2. Applying the same `.ini` configuration files,
3. Compiling and running the same MaxMinNode module.

4.2 Results

In this section, we evaluate the performance of the proposed clustering scheme. The simulation results are analyzed based on cluster quality, overhead, communication efficiency, energy consumption, scalability, and fairness.

4.2.1 Cluster Quality Metrics

To assess how key parameters influence cluster quality, we isolated the impact of node density and the d-parameter on cluster formation by conducting a series of experiments in a stationary environment. The simulation area was fixed at $1000m \times 1000m$ with a uniform radio range of $250m$, and we evaluated two distinct network densities (N=25 and N=100) while varying the cluster radius (d=2 and d=3). Each scenario was simulated for 30 independent trials (T=30).

Number of Clusters The average number of clusters was measured to assess the algorithm's ability to control backbone size. The results, summarized in Table 2 and Figures 4 , 5 , 6 , 7 demonstrate a high degree of scalability.

- **Impact of Density:** As the network density increased from sparse (N=25) to dense (N=100), the number of clusters remained remarkably stable.
  - For d=2, the average number of clusters rose negligibly from 5.87 ($\pm$0.49) to 6.03 ($\pm$0.32).
  - For d=3, the average actually decreased from 4.37 ($\pm$0.40) to 3.97 ($\pm$0.43). This behavior confirms that the Max-Min heuristic effectively suppresses the "cluster explosion" phenomenon. In the sparse N=25 scenario, network partitioning likely forced the creation of smaller, fragmented clusters. In the dense N=100 scenario, better connectivity allowed the d-hop limit to be fully utilized, merging these fragments into larger, more efficient clusters.
- **Impact of Parameter *d*:** Raising the value of d led to a consistent decrease in the number of clusters formed. Specifically, for a network density of N=100, increasing d from 2 to 3 resulted in a 34% reduction in the average cluster count (from 6.03 to 3.97). These results demonstrate that d effectively regulates cluster density, serving as a key parameter for controlling the overall sparsity of the network.

Table 2. Average Number of Clusters across 30 trials.

Network Size (N)	**Parameter d**	**Avg. Clusters**	**95% CI**
25	2	5.87	$\pm$0.49
25	3	4.37	$\pm$0.40
100	2	6.03	$\pm$0.32
100	3	3.97	$\pm$0.43

Cluster Size Distribution We analyzed the frequency distribution of cluster sizes to assess the algorithm's effect on load balancing and topology consolidation. The distribution is presented in the right-hand plots of the provided Figures 4 , 5 , 6 , 7 .

- **Sparse Networks (N=25:)** For both d=2 and d=3, the distribution showed high variance and a tendency toward smaller clusters. This is expected, as low node density in the $1000m \times 1000m$ area results in fragmentation. The highest frequency for d=2 was observed at cluster sizes of 3 or less, indicating a substantial number of small, localized clusters, often consisting of just the clusterhead and its immediate neighbors.
- **Dense Networks (N=100:)** For d=2, cluster sizes varied broadly (from 2 to approximately 40 nodes), indicating that the available 2-hop reach was fully utilized to aggregate nodes; for d=3, the distribution became much flatter and broader, with

clusters reaching sizes up to approximately 70 nodes—a direct consequence of the larger d value and high connectivity, which allowed the Max-Min logic to dominate wider areas.

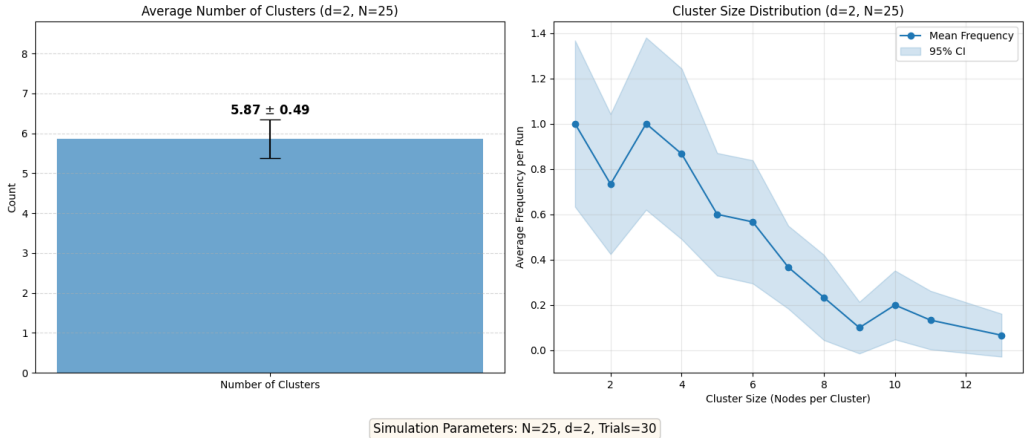

Figure 4 . Cluster Formation Metrics for N=25, d=2

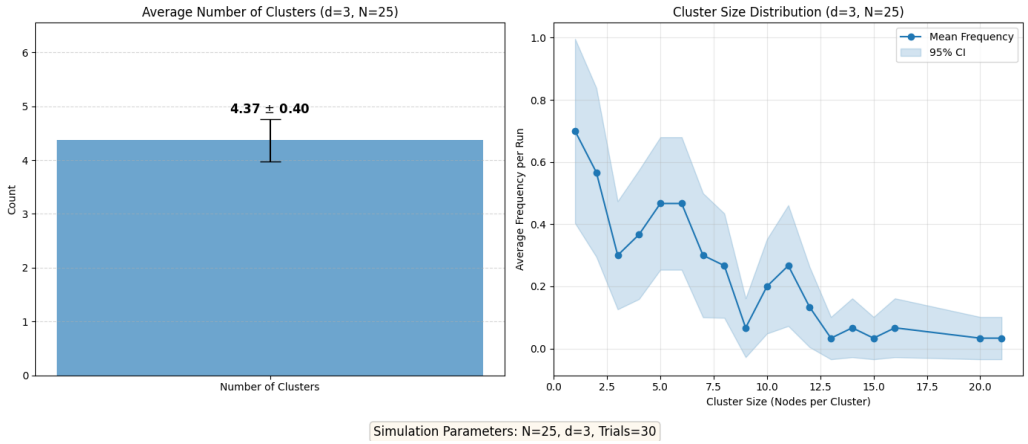

Figure 5 . Cluster Formation Metrics for N=25, d=3

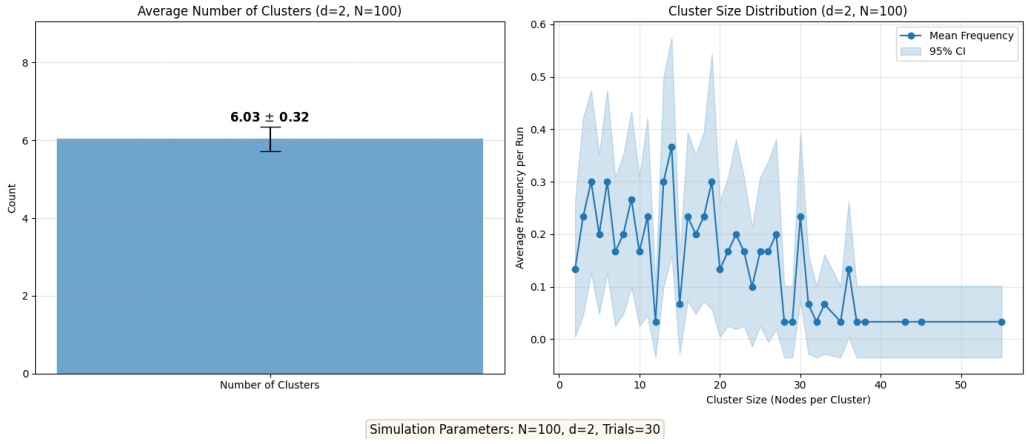

Figure 6 . Cluster Formation Metrics for N=100, d=2

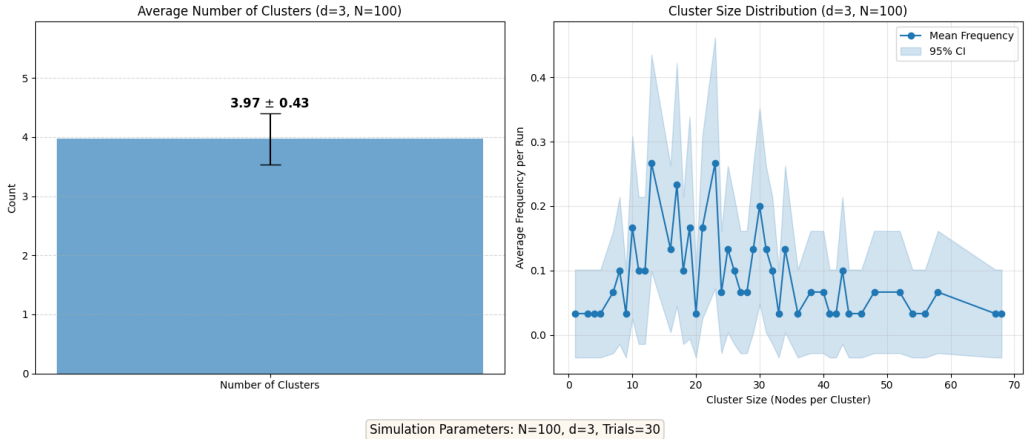

Figure 7 . Cluster Formation Metrics for N=100, d=3

During the measurement of the remaining metrics, we shifted the simulation environment from a static setup to a dynamic model. In this phase, nodes were assigned a maximum speed of 5 meters per tick using the Random Waypoint model, allowing the network topology to evolve over time (Total simulation time was 100 Ticks).

The "Stability" metric refers to the average duration of the cluster member (the time a node remains a member of the same cluster). "CH Lifetime" refers to the average Cluster Head Duration (the time a node stays a cluster head once elected).

Table 3. Dynamic Performance Metrics (Cluster Stability and CH Lifetime).

Network Size (N)	**Parameter d**	**Stability**	**Avg. CH Lifetime (Ticks)**
25	2	0.69 ± 0.051	38.76 ± 2.97
25	3	0.78 ± 0.085	33.87 ± 2.38
100	2	3.81 ± 0.53	39.04 ± 7.99
100	3	3.25 ± 0.21	38.62 ± 4.71

Cluster Stability The Cluster Member Duration (Stability) metric showed the most significant variation (given in 3), clearly illustrating the impact of network density.

- **Impact of Density:** In dense networks (N=100), stability improved dramatically, with the metric for d=2 jumping from 0.6943 in sparse networks to 3.8114—a 5.5x increase. This improvement stems from the higher connectivity in dense networks, which offers redundant communication paths and buffers against node movement. In sparse networks (N=25), even minor node movement quickly pushes neighbors outside the limited d-hop range, triggering frequent cluster re-associations. Conversely, in dense networks, a moving node is more likely to stay within the d-hop reach of a cluster head due to the abundance of intermediate nodes, resulting in significantly longer and more stable cluster memberships.
- **Impact of Parameter *d*:** In the sparse network (N=25), increasing the hop limit d from 2 to 3 improved cluster stability—rising from 0.6943 to 0.7815—because the expanded radius helped sustain connectivity under low-density conditions. In the dense network (N=100), increasing d from 2 to 3 resulted in a modest stability decrease—from 3.8114 to 3.2515—attributed to the "larger target" effect: at d=3, a small number of massive clusters emerge, each with extensive boundaries. When a node crosses such a boundary, it often must re-associate with a distant cluster head, resulting in more disruptive transitions compared to the smoother, localized re-associations seen in the smaller, more numerous clusters at d=2.

Cluster Head Lifetime The CH Lifetime remained remarkably consistent across all four scenarios—regardless of node density (N=25 vs. N=100) or cluster radius (d=2 vs. d=3)—averaging approximately 38.5 ticks, as shown in 3. This consistency suggests that cluster head turnover is driven primarily by localized events, such as a CH moving out of its neighbors' d-hop range or the sudden appearance of a higher-ID node, rather than network-wide stress, while the Max-Min heuristic's stability focus further ensures that replacements occur only when absolutely necessary due to topological changes [2].

4.2.2 Overhead Metrics

Control Overhead The control overhead of the Max-Min d-Cluster algorithm is intrinsically linked to the floodmax and floodmin cycles required for cluster head election. As detailed in the methodology, the algorithm requires a total of $2d$ communication rounds to converge. In each round, every node broadcasts its WINNER value to its immediate 1-hop neighbors. Therefore, the baseline overhead per election cycle scales linearly with the parameter d. Increasing d from 2 to 3 increases the mandatory flooding rounds from 4 to 6, naturally raising the control message volume during the formation phase. Furthermore, in dense networks, the expanded participation of nodes as Gateways adds to the maintenance overhead required to bridge the larger clusters.

Re-clustering Frequency Re-clustering frequency—the rate at which the network must re-run the election process—is the primary driver of dynamic overhead. This can be inferred directly from the Cluster Member Duration (Stability) and Cluster Head Lifetime metrics:

- **Impact of Density:** Sparse networks exhibit high re-clustering frequency. The simulation results show that stability in sparse networks is extremely low (0.69 ± 0.051 ticks for d=2). This rapid turnover forces the network to constantly re-initiate the

$2d$-round election process, generating significant continuous overhead compared to dense networks, where stability is roughly 5.5× higher (3.81±0.53 ticks).

- **Impact of Mobility:** High mobility exacerbates re-clustering overhead. As node speed increased to 50 m/tick, the average CH lifetime dropped to just 6.34 ticks. This "rapid turnover" indicates that high-speed environments suffer from a "re-clustering storm," where the control overhead from frequent re-elections likely dominates the available bandwidth.

Computation Cost The computational cost per node is minimal due to the simplicity of the heuristic. The algorithm relies on simple comparisons of node IDs (e.g., electing the "largest value" or "smallest value") rather than complex weight calculations. However, the frequency of these computations is dictated by the re-clustering rates noted above. In sparse or high-mobility scenarios, nodes are forced to execute these selection rules—checking for "node pairs" or evaluating WINNER logs —much more frequently, accumulating a higher total computational burden over the simulation time.

4.2.3 Communication Efficiency

To evaluate the network's ability to support reliable data transmission, we conducted a traffic analysis simulation. The traffic load was configured at a constant rate of 2 packets per step with a payload size of 4096 bits. We measured three key metrics: Packet Delivery Ratio (PDR), End-to-End Delay, and Throughput. The results for 30 iterations are summarized in Table 4.

Table 4. Communication Efficiency Metrics under Varying Density and Speed (30 Iterations).

Network Size (N)	Param (d)	Max Speed	PDR (%)	End-to-End Delay (s)	Throughput (Kbps)
25	2	5 m/s	64.93 ± 5.96	0.0605 ± 0.0036	5.32 ± 0.49
25	3	5 m/s	65.83 ± 4.97	0.0618 ± 0.0029	5.39 ± 0.41
25	3	10 m/s	65.25 ± 3.48	0.0628 ± 0.0020	5.35 ± 0.28
100	2	5 m/s	99.93 ± 0.14	0.0581 ± 0.0007	8.19 ± 0.01
100	3	5 m/s	99.98 ± 0.03	0.0585 ± 0.0010	8.19 ± 0.00
100	3	10 m/s	99.98 ± 0.03	0.0583 ± 0.0009	8.19 ± 0.00

Throughput The dense network saturated the offered load, achieving a steady 8.19 Kbps. The sparse network struggled to sustain this rate, capping at roughly 5.3 Kbps due to the frequent packet drops caused by topology gaps.

End-to-End Delay The performance analysis reveals that both dense and sparse networks exhibit remarkably similar delay characteristics, with the dense network averaging approximately 0.058 seconds and the sparse network ranging between 0.061 and 0.063 seconds. While the dense network benefits from high connectivity, enabling the clustering algorithm to establish optimized, direct backbone paths between gateways, the sparse network occasionally routes packets along slightly longer paths or experiences minor queuing delays as routes are dynamically repaired. The impact of increasing the parameter d from 2 to 3 also aligns with theoretical expectations, resulting in only a marginal delay increase—such as from 0.0605 to 0.0618 seconds in the sparse network—due to the modest rise in hop count within larger clusters. Overall, the differences in delay remain negligible across configurations.

Packet Delivery Ratio (PDR) The results indicate a binary performance characteristic based on network density. In dense networks with 100 nodes, the algorithm demonstrated

near-perfect reliability, achieving packet delivery ratios (PDRs) exceeding 99% irrespective of node speed or the parameter d, confirming that the Max-Min virtual backbone effectively routes almost all traffic without loss in well-connected environments. However, in sparse networks with only 25 nodes, the PDR plummeted to around 65%, likely due to network partitioning; with nodes spread across a 1000m×1000m area, physical paths to the destination frequently became unavailable, undermining clustering efficiency and overall performance.

4.2.4 Energy and Resource Metrics

This section evaluates the efficiency and fairness of the Max-Min d-Cluster algorithm concerning resource utilization. We model energy consumption based on explicit penalty costs for high-responsibility roles (Cluster Head and Gateway) and measure overall network sustainability through the First Node Depletion (FND) metric. All experiments in this section used a cluster radius of d=3. Each scenario was simulated for 30 independent trials (T=30).

Table 5. Energy and Resource Metrics ($d = 3$) with 95% Confidence Intervals.

Scenario	**Avg. Energy Consumed (Units)**	**Load Balancing (Variance)**	**Network Lifetime (FND Step)**
N=25, Speed=5	85.98 ± 1.00	154.33 ± 13.35	124.00 ± 0.00
N=25, Speed=10	85.85 ± 0.79	127.60 ± 8.54	124.00 ± 0.00
N=100, Speed=5	91.81 ± 0.96	74.96 ± 8.09	124.00 ± 0.00
N=100, Speed=10	93.67 ± 0.75	41.18 ± 5.08	124.00 ± 0.00

Energy Consumption per Node The average energy consumption in the Max-Min d-Cluster algorithm demonstrates a clear dependence on node density rather than mobility, with dense networks consuming significantly more energy (averaging 91.81–93.67 units) compared to sparse networks (averaging 85.85–85.98 units) 5. This trend arises because denser networks incur higher passive energy costs due to the increased number of nodes and generate more Gateway nodes, which introduce additional energy penalties. Although the number of cluster heads remains low—as observed in Section 4.2.1—the expanded participation of nodes in the backbone (as Gateways) directly escalates overall network energy usage, reflecting the trade-off between improved connectivity and heightened energy consumption in denser environments.

Network Lifetime The First Node Depletion (FND) time remained invariant across all four scenarios, consistently occurring at step 124.0 as can be interpreted from the Table 5. This striking uniformity reveals that FND is not driven by network-wide dynamics—such as node density or mobility—but instead by the minimum required CH set size and the disproportionately high energy penalty imposed on the CH role.

In practice, the node elected as the "unlucky" first cluster head—often tasked with covering the largest area or managing the most gateway connections—depletes its energy at a fixed, accelerated rate, regardless of whether the network is sparse or dense, static or highly mobile.

This pattern underscores a critical limitation: while the Max-Min d-Cluster algorithm ensures stability in clustering, it lacks an intrinsic mechanism to mitigate the premature failure of the most burdened node. To enhance sustainability, an external load-balancing or energy-aware intervention may be necessary to redistribute responsibilities and prevent early depletion of critical nodes.

Load Balancing The Load Balancing metric, quantified by the variance in energy consumption, revealed the most significant operational insights:

- **Impact of Density:** Increasing node density dramatically improved load balancing, as evidenced by the sharp reduction in the Table 5 in variance—from 154.33 (N=25) to 74.96 (N=100, $Speed$=5). In sparse networks, the limited number of nodes forces reliance on a small, fixed set of CHs, resulting in high variance as the same nodes bear the burden of cluster management repeatedly. Conversely, dense networks benefit from a larger pool of eligible nodes, enabling the system to distribute responsibilities more evenly through re-election. This broader participation mitigates overloading individual nodes, fostering fairer resource utilization and greater stability in role assignment.
- **Impact of Mobility:** Increased mobility significantly enhanced load balancing, as the Table 5 demonstrates the variance dropping from 74.96 (N=100, $Speed$=5) to a minimum of 41.18 (N=100, $Speed$=10). This improvement reflects the "fairness dividend" of high mobility: frequent node movement triggers more CH re-elections and member re-associations, effectively distributing the energy-intensive CH and Gateway roles across a broader set of nodes over time. Mobility acts as a constant randomizing force, disrupting the algorithm's stability bias and preventing any single node from shouldering the burden for extended periods. As a result, the network achieves greater fairness in role assignment, mitigating localized overuse and promoting balanced resource consumption.

4.2.5 Scalability and Robustness

Scalability The results from the Number of Clusters metric underscore the Max-Min d-Cluster algorithm's strong scalability: as network density increased from sparse to dense, the average number of clusters remained remarkably stable, demonstrating the algorithm's effectiveness in preventing "cluster explosion" even in high-density environments. This stability further validates its ability to maintain efficient clustering regardless of network scale.

Robustness to Mobility The Cluster Head (CH) Lifetime shows a clear and expected inverse relationship with node speed. As the maximum speed increased from 10 m/tick to 50 m/tick, the average CH lifetime plummeted from 21.92 ticks to just 6.34 ticks (Table 6), a dramatic decrease driven by nodes frequently moving out of their cluster's d-hop range and triggering rapid re-elections. This trend confirms that even the Max-Min algorithm's inherent stability bias is insufficient to counteract the physical disassociation caused by high mobility, necessitating a high turnover rate to uphold the d-hop coverage guarantee.

Connectivity Maintenance The algorithm's ability to maintain connected clusters—ensuring that nodes can reach each other via gateways—was heavily dependent on node density rather than just the d-hop parameter.

Table 6. Robustness Metrics ($N = 25, d = 3$) under Varying Node Speeds.

Max Speed (m/tick)	Avg. Head Lifetime (Ticks)
10	21.92 ± 1.73
25	10.61 ± 0.52
50	6.34 ± 0.21

- **Impact of Density:** In sparse networks, the simulation results indicated significant challenges in maintaining global connectivity. As noted in the cluster formation analysis (Section 4.2.1), low node density caused "network partitioning," forcing the creation of small, fragmented clusters that lacked sufficient neighbors to bridge gaps. Conversely, in dense environments, the algorithm successfully maintained connectivity. The energy analysis (Section 4.2.4) confirmed that dense networks generated a higher number of Gateway nodes. While this increased energy consumption, it provided the necessary redundancy to keep clusters interconnected (bridged) even as topology changed.
- **Impact of Mobility:** Under high mobility (50m/tick), maintaining a stable connected backbone proved difficult. The "Robustness" results (Table 6) showed that Cluster Head lifetime dropped to just 6.34 ticks. This rapid turnover of the backbone implies that while local clusters might remain formed, the inter-cluster links (maintained by Gateways connecting to these CHs) would frequently break and re-form. The "physical disassociation" caused by high speeds suggests that while the Max-Min algorithm is robust at forming clusters, maintaining a continuous end-to-end path requires either higher density to provide backup Gateways or lower speeds to preserve CH stability.

4.2.6 Fairness and Role Distribution

Fairness in Cluster Head Selection The simulation results reveal a complex relationship between the algorithm's stability goals and its ability to distribute the Cluster Head (CH) burden fairly.

- **Impact of Mobility:** According to the Load Balancing (Variance) metrics in Table 4, mobility acted as a critical equalizer. In lower mobility scenarios (Speed=5), the variance in energy consumption was high (74.96), indicating that certain nodes were unfairly burdened with the CH role for long periods. However, increasing the speed to 10m/tick reduced this variance by nearly half to 41.18. This confirms that while high mobility disrupts stability, it effectively enforces fairness by frequently "deposing" CHs and forcing a rotation of the leadership role among different nodes.
- **Failure to Prevent Early Depletion:** Despite the improved average fairness under mobility, the algorithm failed to protect the single most burdened node. As shown in Table 5, the First Node Depletion (FND) consistently occurred at step 124.0 across all scenarios, regardless of network density or speed. This indicates that the algorithm lacks a proactive mechanism to rotate leadership before a node is drained. The "unlucky" first elected CH is forced to serve until depletion, demonstrating a critical flaw in preventing early battery death for high-responsibility nodes.

Gateway Node Efficiency The efficiency of gateway nodes was evaluated based on the trade-off between maintaining connectivity and energy cost.

- **Redundancy vs. Energy Cost:** In dense networks, the algorithm demonstrated

high effectiveness in recruiting gateways to maintain inter-cluster paths. The "Energy Consumption" analysis (Section 4.2.4) noted that dense networks consumed significantly more energy (avg. 91.81–93.67 units) compared to sparse ones. This increase was explicitly attributed to the "expanded participation of nodes... as Gateways".

- **Bottleneck Management:** While this higher energy cost suggests a reduction in individual node efficiency, it prevented bottlenecks by providing "redundant communication paths". In the d=3 scenarios, where clusters grew to massive sizes of up to 70 nodes (Figure 7), these numerous gateways were essential for handling the inter-cluster traffic of such large groupings, effectively trading energy resources for structural stability.

4.3 Discussion

The primary objective of this study was to evaluate the Max-Min d-Cluster Algorithm's ability to maintain stable and efficient clustering in MANETs, specifically addressing the challenges of scalability, mobility management, and resource allocation. By revisiting the initial hypothesis—that this heuristic can balance stability and overhead across varying network conditions—we can interpret the experimental results as follows:

Scalability and the Suppression of Cluster Explosion The study initially identified "cluster explosion" as a critical risk in large-scale networks, where an excessive number of small clusters degrades performance. The results strongly support the hypothesis that Max-Min d-Clustering effectively mitigates this issue. As network density increased fourfold (from N=25 to N=100), the number of clusters remained remarkably stable, and even decreased when the hop parameter d was increased to 3. This confirms that the d parameter serves as a powerful tunable lever for regulating network sparsity. By enabling nodes to form larger, more consolidated clusters in dense environments (reaching sizes up to 70 nodes), the algorithm successfully prevents the fragmentation seen in simpler 1-hop schemes, ensuring that the backbone size does not grow linearly with the network size.

The "Fairness Dividend" of Mobility A key theoretical challenge in MANETs is the "unlucky node" problem, where static clustering algorithms unfairly burden specific nodes with the energy-intensive Cluster Head role. Our findings reveal a nuanced trade-off between stability and fairness. In static or low-mobility scenarios (Speed=5), load balancing was poor, with high variance in energy consumption. However, higher mobility (Speed=10) acted as an equalizer, reducing energy variance by nearly half. This interprets mobility not merely as a disruptive force, but as a necessary "randomizer" that forces leadership rotation. While the algorithm was designed for stability (favoring re-election), high mobility naturally counteracts this by physically separating CHs from their members, inadvertently promoting a fairer distribution of the energy burden over time.

Stability vs. Connectivity in Sparse Networks The hypothesis that the algorithm would maintain stable clusters "across a wide range of network conditions" faced limitations in sparse environments. In sparse networks (N=25), the algorithm struggled with network partitioning, leading to low stability (0.69 ticks) and poor Packet Delivery Ratios (65%). This suggests that while the Max-Min heuristic is robust in forming clusters, it cannot overcome physical layer limitations where node density is insufficient to support d-hop connectivity. In contrast, dense networks achieved near-perfect reliability (PDR¿99%) and high stability (3.81 ticks). This dichotomy highlights that the algorithm's performance

is heavily density-dependent; it excels in "rich" connectivity environments but requires supplementary mechanisms to function effectively in "sparse" or fragmented topologies.

The Energy-Stability Trade-off The results unveiled a critical hidden cost to the algorithm's stability. While dense networks provided superior connectivity and stability, they incurred a higher total energy cost due to the recruitment of numerous Gateway nodes to bridge the massive clusters. Furthermore, the invariant First Node Depletion time across all scenarios exposes a fundamental flaw: the algorithm lacks a proactive load-balancing mechanism. It effectively sacrifices the "first elected" nodes to maintain structural stability, forcing them to serve until depletion. This contradicts the ideal of a "sustainable" system, indicating that while the network structure is stable, the individual nodes are vulnerable. Future iterations of this algorithm must likely decouple stability from tenure, forcing leadership rotation before critical energy thresholds are breached.

Conclusion on the Hypothesis In conclusion, the hypothesis is partially validated. The Max-Min d-Cluster Algorithm successfully achieves structural stability and scalability in dense networks, effectively managing control overhead and preventing topology disintegration. However, its efficiency is compromised in sparse environments, and its "stability-first" approach creates an energy imbalance that jeopardizes individual node longevity. The findings suggest that for the algorithm to be truly robust across all MANET scenarios, it must be augmented with energy-aware constraints that can override the default stability logic when a Cluster Head is near depletion.

References

[1] I. Ullah, T. Hussain, A. Khan, I. Ali, F. Ali, and C. Choi, ''Retracted article: Analyzing the impacts of node density and speed on routing protocol performance in firefighting applications,'' *Fire Ecology*, vol. 19, no. 1, p. 62, Oct 2023. [Online]. Available: https://doi.org/10.1186/s42408-023-00220-4

[2] A. Amis, R. Prakash, T. Vuong, and D. Huynh, ''Max-min d-cluster formation in wireless ad hoc networks,'' in *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064)*, vol. 1, 2000, pp. 32--41 vol.1.

[3] I. Chlamtac, M. Conti, and J. J.-N. Liu, ''Mobile ad hoc networking: imperatives and challenges,'' *Ad Hoc Networks*, vol. 1, no. 1, pp. 13--64, 2003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1570870503000131

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

[5] T. Rahman, I. Ullah, A. Rehman, and R. Naqvi, ''Clustering schemes in manets: Performance evaluation, open challenges, and proposed solutions,'' *IEEE Access*, vol. PP, pp. 1--1, 01 2020.

[6] N. Khatoon, P. Pranav, S. Roy, and Amritanjali, ''Fq-mec: Fuzzy-based q-learning approach for mobility-aware energy-efficient clustering in manet,'' *Wireless Communications and Mobile Computing*, vol. 2021, no. 1, p. 8874632, 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1155/2021/8874632

[7] G. Abdulsaheb, O. Khalaf, N. Sulaiman, H. Zmezm, and H. Zmezm, ''Improving ad hoc network performance by using an efficient cluster based routing algorithm,'' *Indian Journal of Science and Technology*, vol. 8, 12 2015.

[8] I. Shayeb, A. Hamzah, and A. Nassuora, ''A survey of clustering schemes for mobile ad-hoc network (manet),'' vol. 20, pp. 135--151, 07 2011.

[9] S. Pal and S. P. Singh, ''Survey on mobility based clustering algorithms in manet,'' *International Journal of Engineering Research Technology (IJERT)*, vol. 1, no. 10, Dec 2012. [Online]. Available: https://www.ijert.org/research/survey-on-mobility-based-clustering-algorithms-in-manet-IJERTV1IS10201.pdf

[10] S. K. Rath, ''A survey on one-hop clustering algorithms in mobile ad hoc networks,'' *Journal of Network and Systems Management*, 2009.

[11] 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, Apr 2002. [Online]. Available: https://doi.org/10.1023/A:1013941929408

[12] S. Zhang, X. Liu, and M. Trik, ‘‘Energy efficient multi hop clustering using artificial bee colony metaheuristic in wsn,’’ *Scientific Reports*, vol. 15, no. 1, p. 26803, Jul 2025. [Online]. Available: https://doi.org/10.1038/s41598-025-12321-y

[13] A. elBanna, B. ElHalawany, A. Bayomy, J. Huang, and K. Wu, ‘‘Machine learning-based multi-layer multi-hop transmission scheme for dense networks,’’ *IEEE Communications Letters*, vol. PP, pp. 1--1, 09 2019.

[14] N. Nathiya, R. Chinnasamy, and K. Geetha, ‘‘A hybrid optimization and machine learning based energy-efficient clustering algorithm with self-diagnosis data fault detection and prediction for wsn-iot application,’’ *Peer-to-Peer Networking and Applications*, vol. 18, 01 2025.

[15] P. Basu, N. Khan, and T. D. C. Little, ‘‘A mobility based metric for clustering in mobile ad hoc networks,’’ *Proceedings 21st International Conference on Distributed Computing Systems Workshops*, pp. 413--418, 2001. [Online]. Available: https://api.semanticscholar.org/CorpusID:1395019

[16] I. Er and W. Seah, ‘‘Mobility-based d-hop clustering algorithm for mobile ad hoc networks,’’ in *2004 IEEE Wireless Communications and Networking Conference (IEEE Cat. No.04TH8733)*, vol. 4, 2004, pp. 2359--2364 Vol.4.

[17] ------, ‘‘Performance analysis of mobility-based d-hop (mobdhop) clustering algorithm for mobile ad hoc networks,’’ *Computer Networks*, vol. 50, pp. 3375--3399, 12 2006.

[18] M. S. Batta, H. Mabed, Z. Aliouat, and S. Harous, ‘‘A distributed multi-hop intra-clustering approach based on neighbors two-hop connectivity for iot networks,’’ *Sensors*, vol. 21, no. 3, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/3/873