Journal of the ACS Advances in Computer Science
Volume 16, Issue 1, 2025

The impact of Integrating K-Means, Mean-Shift and Centrality Clustering with Leach-C on Wireless Sensor Network Lifetime

Imane Aly Saroit, Dina Tarek
Information Technology Department, Faculty of Computers and Artificial Intelligence, Cairo University, Cairo, Egypt,
i.saroit@fci-cu.edu.eg, d.tarek@fci-cu.edu.eg,

Abstract:
Wireless Sensor Networks (WSNs) consist of numerous sensor nodes deployed over a specific area to monitor environmental factors such as temperature, humidity, pressure and motion. These nodes communicate wirelessly and collaborate to transmit the collected data to a central base station for further processing. WSNs have become essential in various domains, including environmental monitoring, healthcare systems, military surveillance, industrial automation, and the development of smart cities. Despite their broad range of applications, a major challenge in WSNs is the limited battery power of sensor nodes. Since replacing or recharging these batteries is often impractical, especially in remote or hazardous locations, energy efficiency becomes a critical consideration in WSN design. To address this, clustering techniques are widely adopted to enhance energy efficiency and extend network lifetime. In clustering, sensor nodes are grouped into clusters, each managed by a Cluster Head (CH). The CH is responsible for aggregating data from its cluster members and forwarding it to the base station, thereby minimizing redundant transmissions and optimizing energy usage. LEACH (Low-Energy Adaptive Clustering Hierarchy) is one of the most well-known hierarchical clustering protocols for WSNs. LEACH-C, an enhanced version, introduces centralized control by having the base station form clusters based on the nodes' energy levels. This paper proposes integrating LEACH-C with three clustering algorithms—K-means, Mean-Shift and Closeness Centrality; to evaluate their impact on energy efficiency and network lifetime. Simulation results show that LEACH-C combined with K-means clustering achieves the best performance, significantly reducing energy consumption and prolonging the network's lifetime.

Keywords: *Wireless Sensor Networks (WSNs), Energy Efficiency, Network Lifetime, K-means Clustering, Mean-Shift Clustering, Closeness Centrality.*

1. Introduction

Wireless Sensor Networks (WSNs) [1],[2] consist of spatially distributed autonomous sensor nodes that monitor environmental conditions such as temperature, humidity, pressure and motion. These sensor nodes communicate wirelessly and collaborate to transmit collected data to a base station. WSNs have applications in environmental monitoring, healthcare, military surveillance

DOI:

and smart cities. However, the limited battery life of sensor nodes poses a significant challenge, making energy efficiency a critical consideration in WSN design. The nodes in the WSNs may be uniformly distributed such as in smart agriculture fields, smart cities, structural health monitoring … etc. They also may not be uniformly distributed such as in case of precision agriculture, traffic monitoring, environmental Monitoring … etc.

Routing in Wireless Sensor Networks (WSNs) is the process of efficiently transmitting data from sensor nodes to the Base Station (BS), also known as sink node; while minimizing the energy consumption. Since sensor nodes have limited power, storage and processing capabilities, routing protocols in WSNs are designed to be energy-efficient and scalable.

Clustering [3],[4] is an effective technique for improving the energy efficiency and scalability of WSNs. In clustering, sensor nodes are grouped into clusters, each managed by a Cluster Head (CH). The cluster heads aggregate data from cluster members and transmit it to the base station, reducing energy consumption by minimizing redundant transmissions. Different clustering methods can impact network lifetime, stability and efficiency.

Choosing an appropriate clustering method is essential for maximizing the lifetime of a WSN [5]. An effective clustering algorithm should minimize the communication overhead, evenly distribute energy consumption and adapt to network topology changes. The selection of clustering techniques significantly affects the network's performance.

LEACH is a widely used hierarchical clustering-based distributed routing protocol designed for WSNs. A large number of enhancements have been proposed to improve its performance. Among these is LEACH-C, which employs centralized control. The base station forms the clusters based on the nodes' energy and location. This paper proposes integrating the LEACH-C protocol with three well-known clustering methods; K-means, Mean-Shift and Closeness Centrality, to evaluate their effectiveness in enhancing energy efficiency and network lifetime. A detailed simulation program for LEACH-C was developed to analyze the impact of these clustering techniques under varying data rates and node distributions.

Energy consumption in WSNs depends on various factors, including the transmission distance, the node density and the clusters’ structure. The energy model used in [6],[7], help to estimate the energy consumption in different clustering schemes.

The remainder of this paper is organized as follows: Section 2 defines WSNs, then reviews the related work on routing in WSNs. Section 3 reviews clustering and explains in details the three experienced methods used in the paper; K-Means, Mean-Shift and Centrality. Section 4 details the methodology used for this study. Section 5 presents the simulation results and analysis. Finally, Section 6 concludes the paper and suggests directions for future research.

2. Literature Review

Wireless Sensor Networks (WSNs) [1],[2] are groups of small, wireless devices called sensor nodes that collect information from their surrounding area, like temperature, humidity, or motion. These sensor nodes communicate with each other and send the data to a central device called a base station or sink node (Figure 1). The base station either processes the data or forwards it to a computer system for analysis. WSNs are used in many areas like agriculture, healthcare, environmental monitoring and smart cities. WSNs can be placed in hard-to-reach places and work without cables. WSNs need to balance power use, data accuracy and security to work effectively.

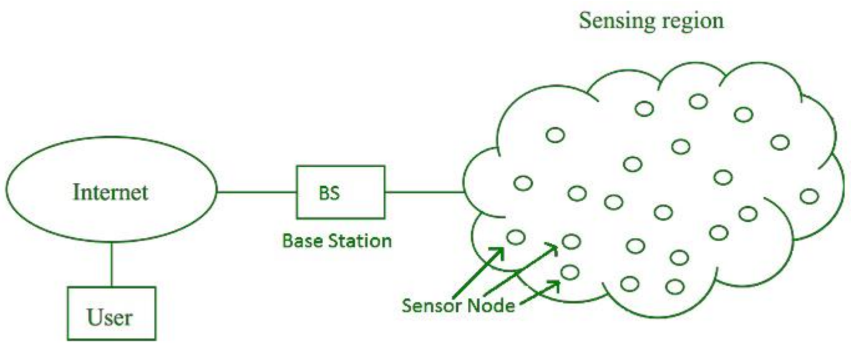


Figure 1: Wireless Sensor Networks [14]

The main problem in routing for WSNs is to find a balance between minimizing energy consumption and maintaining reliable communication. Since sensor nodes have limited-energy batteries, excessive energy use can cause nodes to die, leading to network partitioning and data loss. Routing protocols need to be energy-efficient, scalable, fault-tolerant and adaptable to the network's dynamic nature to ensure the network's lifetime and efficiency.

LEACH [15],[16] is the most famous hierarchical clustering-based routing protocol designed for wireless sensor networks (WSNs). It was introduced to enhance energy efficiency and network life time by dynamically forming clusters and rotating the role of cluster heads. In this algorithm, the sensors are grouped into clusters, each with a cluster head responsible for aggregating data and forwarding it to the base station. The cluster heads are randomly selected. The main advantage of LEACH is the reduction of energy consumption by using rotating cluster heads, while its main disadvantage is the random selection of cluster heads, which may lead to poor cluster formation (e.g., low-energy nodes becoming cluster heads, unevenly distributed cluster heads).

Many modifications were done on the LEACH algorithm to improve its performance, following are some centralized protocols inspired from the LEACH protocol

LEACH-C (Centralized LEACH) [17], uses the base station (BS) to select the cluster heads based on energy levels and location. The base station collects energy and location information from all nodes. It then selects the cluster heads randomly from the nodes with energy above the average. The main advantage of LEACH-C is; it ensures better energy efficiency, while its main disadvantage is the excess overhead on the BS, which leads to scalability issue.

LEACH-G [18] is used in large networks, it introduces gateway nodes between cluster heads and the base station. The cluster heads send the data to a gateway, which forwards it to the BS. The main advantage of LEACH-G is the reduction of the energy consumption, while its main disadvantage is the extra overhead used for managing gateway nodes.

LEACH-B [19] aims to achieve a more balanced energy consumption among sensor nodes. It refines cluster head selection by considering both energy levels and distance from the base station. The main advantage of LEACH-B is the avoidance of excessive energy drain on specific nodes, while its main disadvantage is the increase of the initial computation overhead.

LEACH-SM (Split and Merge LEACH) [20] uses split and merge mechanism for clusters. The base station monitors the clusters’ size centrally. If a cluster becomes too large, it is split to balance the communication load. Conversely, if a cluster becomes too small due to node failures, it is merged with a neighboring cluster to maintain network connectivity. The main advantage of LEACH-SM is; it helps to distribute the energy load evenly among nodes, so extending the network lifespan, while its main disadvantage is the increase of computational load on the base station due to the use of split and merge mechanism causing delay.

HELA-LEACH (Hybrid Energy & Load Aware LEACH) [6] selects the cluster heads by considering both the nodes’ energy levels and load. The main advantage of HELA-LEACH is the prevention of energy depletion of heavily loaded nodes, while its main disadvantage is the increased computation at the base station for load balancing.

LEACH-F (Fixed Clustering LEACH) [21] improves LEACH by maintaining fixed clusters throughout the network's lifetime. Meaning that, LEACH-F selects the cluster heads only once during initialization. The main advantage of the LEACH-F is the reduction setup overhead, so saving the energy used in CHs selection, while its main disadvantage is the lack of adaptability to network changes, so it cannot deal with nodes' death or fail.

Also, following are some distributed protocols inspired from the LEACH protocol:

V-LEACH (Vice-Cluster Head LEACH) [22] introduces a vice-cluster head (VCH) in each cluster. The VCH takes in case of the death of the primary cluster head, ensuring continuous data transmission and reducing packet loss. The main advantage of the V-LEACH is the reduction of packet loss due to cluster head failures, while its main disadvantage is the extra energy consumption needed for maintaining a VCH.

DE-LEACH (Distance and Energy Aware LEACH) [23] considers both the residual energy of nodes and their distances to the base station when electing cluster heads. The main advantage of DE-LEACH is; it ensures that nodes with higher energy levels take on the more demanding role of cluster head, while its main disadvantage is the increase of both the processing time and energy consumption during the cluster head election phase.

Several other variations of LEACH have been proposed [24],[25],[26].

As any clustering methods can be used with modified centralized LEACH's, the aim of this paper is to compare the effect of using K-Means, Mean-Shift and closeness centrality clustering on the WSN lifetime.

3. Clustering

Clustering [3],[4] is an unsupervised machine learning technique used to group similar data points into clusters based on their characteristics or relationships. The goal is to maximize intra-cluster similarity and inter-cluster dissimilarity. Meaning that points within the same cluster should be highly similar, while unsimilar points should be in different clusters. Many clustering methods exist, this paper experiences three of the most famous ones; K-Means, Mean-Shift and Centrality.

3.1 K-means

K-means [8],[9] aims to partition a dataset into K distinct, non-overlapping clusters. Each data point is assigned to the cluster with the nearest mean (also called a cluster centroid). It takes the following steps:

1. Choose the number of clusters K: Define the number of clusters you want to form.
2. Initialize centroids: Randomly place K cluster centroids in the data space.
3. Assign each point to the nearest centroid: Each data point x_i is assigned to the closest centroid based on the distance.

$$Ci = argmin_{\,j} \left\| x_i - \mu_j \right\| \tag{1}$$

where: C_i is the cluster assigned to point x_i

μ_j is the centroid of cluster j

$\left\| x_i - \mu_j \right\|$ is the Euclidean distance between the point and the centroid.

4. Update centroids: Compute the mean of all points in each cluster and update the centroid's position. For each cluster n with L points, the new centroid is $\mu_n(x, y)$

$$\mu_n(x, y) = \frac{1}{L} \sum_{i=1}^{L} x_i \tag{2}$$

where: $\mu_n(x, y)$ is the new centroid of cluster n

L is the number of points in the cluster.

x_i represents all points in cluster n.

5. Repeat the two previous steps 3 & 4: Until centroids changes are very small.

$$||\mu_n(t + 1) - \mu_n(t)|| < Threshold, \quad \forall n \tag{3}$$

where: $\mu_n(t)$ is the centroid of cluster n at iteration t.

We may also stop if a predefined number of iterations is reached

6. Final clusters: Once the centroids remain stable, clusters are finalized.

$$C = \{C_1, C_2, C_3, \dots \dots, C_K\} \tag{4}$$

In the procedure described above, the number of clusters is predetermined, which is not ideal for WSN clustering since the optimal number of clusters should be determined dynamically. The most famous and widely used methods for determining the optimal number of clusters in K-Means are [27],[28]:

- **Elbow Method:** Finds the "elbow point" where adding more clusters barely reduces variation within clusters WCSS (Within-Cluster Sum of Squares).
- **Silhouette Method:** Measures how well-separated and cohesive clusters are. Higher-scores means better clusters.
- **Gap Statistic:** Compares clustering results to random dataset to find the best number of clusters.

The Silhouette Method [27] is often preferred over the Elbow Method and Gap Statistic because it provides a clear, interpretable measure of clustering quality over the other two ones.
A simpler definition to the silhouette method; it is a technique that measures how well each data point lies within its assigned cluster, comparing its cohesion (how close it is to other points in the same cluster) and separation (how distant it is from points in other clusters). It takes the following steps:

1. Apply K-Means: Run the K-Means algorithm on the dataset for various values of K (where K takes a value between K_{min} and K_{max} inclusive).
2. Compute the Silhouette Coefficient for each point: For each point i calculate S_i as follows:

$$s_i = \frac{b_i - a_i}{max(a_i, b_i)} \tag{5}$$

   where: a_i is the average distance from i to all other data points in the *same* cluster (Cohesion).

   b_i is the minimum average distance from i to all points in the *nearest different cluster* (Separation).

   - If S_i is close to 1, the data point is well clustered.
   - If S_i is close to 0, the point lies on the decision boundary between two clusters.
   - If S_i is negative, the data point might be assigned to the wrong cluster.
3. Compute the Average Silhouette Score: Calculate the average silhouette coefficient for all points to obtain a measure for that particular value of K:

$$S(K) = \frac{1}{n}\sum_{i=1}^{n} S_i \tag{6}$$

where: n is the total number of points.

4. Determine the Optimal Number of Clusters: The optimal number of clusters K is the one that has the maximum silhouette score $S(K)$.

$$\widehat{K} = \arg max_{S(K)} \tag{7}$$

5. Final clusters are produced:

$$\hat{C} = \{\hat{C}_1, \hat{C}_2, \hat{C}_3, \dots\dots, \hat{C}_{\widehat{K}}\} \tag{8}$$

3.2 Mean-shift

Mean-shift [10],[11] is a centroid-based algorithm that moves each data point toward the nearest area of highest data density by iteratively shifting points toward the mean of points in a defined window (called a kernel or bandwidth). Unlike the two other methods; the number of clusters is not defined in advance.

It takes the following steps:

1. Start with initial centroids: Start with each point in the dataset as candidate centroid.
2. Compute the mean shift: For each point x_i, finds all points within a given bandwidth (h).

$$N(x_i) = \{x_j \in D \mid \|x_j - x_i\| < h\} \tag{9}$$

where: D is the points dataset

h is the Bandwidth (window radius).

x_j are the neighboring points within the bandwidth h.

3. Move cluster centers: Each point x_i is shifted towards the mean of its neighbors.

$$x_i(t+1) = \frac{1}{|N(x_i)|}\sum_{x_j \in N(x_i)} x_j \tag{10}$$

where: $x_i(t+1)$ is the new mean at iteration (t+1), i.e. the new position of x_i

$|N(x_i)|$ is the number of neighbors within the bandwidth.

4. Repeat the two previous steps 2 &3: Until the shifts become very small.

$$\|x_i(t+1) - x_i(t)\| < Threshold, \quad \forall i \tag{11}$$

5. Remove redundant clusters: Merge clusters that are close together.

$$\text{If } \|C_i - C_j\| < \text{Merge\_Threshold, merge } C_i \text{ and } C_j \tag{12}$$

where: C_i and C_j are cluster centers

6. Assign points to clusters: Each data point is assigned to the nearest cluster center.

$$Ci = argmin_{\,j} \|x_i - c_j\| \tag{13}$$

where: C_i is the cluster assigned to point x_i

c_j is the closest cluster center

7. Final clusters are produced:

$$C = \{C_1, C_2, C_3, \dots \dots, C_K\} \tag{14}$$

There are several methods to select the bandwidth for Mean Shift clustering, including [28]:

- **Scott's Rule:** This approach chooses a bandwidth proportional to the data's standard deviation.
- **Silverman's Rule:** The bandwidth is selected based on the median interquartile range of the data. Note that the interquartile range is a measure indicating the spread of the middle 50% of a dataset. It is calculated as the difference between the third quartile and the first quartile.
- **Cross-Validation:** This method involves running Mean Shift clustering with various bandwidth values and assessing the algorithm's performance on a separate test set.
- **Expert Knowledge:** If you have domain-specific expertise or historical data, you can manually adjust the bandwidth.

In this study, the Silverman's Rule is preferred because it's simple, quick to compute and generally provides a reliable bandwidth estimate for most datasets. Unlike cross-validation or expert knowledge, it doesn't require extensive computation or deep domain expertise. Additionally, it is preferred over Scott's Rule as it works better with smaller sample sizes and doesn't hide important details in the data. Overall, it balances simplicity and accuracy for a wider range of datasets. The Bandwidth is calculated using the Silverman's Rule formula [29]:

$$h = 1.06 * \sigma * N^{-\frac{1}{5}} \tag{15}$$

where: h is the Bandwidth.

σ is the standard deviation of the data.

N is the number of the points.

3.3 Centrality

Centrality [12],[13] is a measure in graph theory and network analysis. It quantifies the importance of a node within a network. It helps identifying key nodes that play a crucial role in the network. To form K Clusters using centrality for a graph; the following steps are used:

1. Construct a graph: Represent data as a network where nodes are data points and edges define relationships.
   Graph G (V,E) is defined
   where: V= Points.
   E= Edges between the points.
2. Define an adjacency matrix A where $A_{ij} = 1$ if there is a connection between v_i and v_j, otherwise it is equal to zero.
3. Calculate shortest paths: Compute the shortest paths between all points using any algorithm like Dijkstra's.
4. Compute the centrality: Calculate the centrality *C(v)* for each point (this must be done according to the used centrality method).
5. Identify central nodes: Select the high-centrality points as initial cluster centers.
   $$v_c = \arg\max C(v) \quad (16)$$
   where: *C(v)* is the chosen centrality measure.
   Select the top K highest as cluster centers. If multiple nodes have similar centrality values, additional criteria like node degree, proximity, or randomness can be applied to select the top K highest as cluster centers.
6. Group nodes into clusters: For each point v_i, assign it to the nearest central node using shortest path distance:
   $$Ci = argmin_{\,c}\|v_i - v_c\| \quad (17)$$
   where: Ci is the assigned cluster for node v_i.
   v_c is the cluster center.
7. Final clusters are produced:
   $$C = \{C_1, C_2, C_3, \dots\dots, C_K\} \quad (18)$$

The most famous and widely used centrality measures are [30],[31]:

- **Degree Centrality** – Measures node importance based on the number of direct connections (edges) it has.

- **Betweenness Centrality** – Identifies nodes that act as bridges by measuring how often a node appears on shortest paths between other nodes.
- **Closeness Centrality** – Ranks nodes by their average shortest path distance to all other nodes, favoring those that can quickly reach others.
- **Eigenvector Centrality** – Assigns higher importance to nodes connected to other highly connected nodes.

As closeness centrality measures how close a node is to all other nodes in a cluster, this means that nodes with high closeness centrality are centrally located within their clusters, making them good representatives of cluster centers. In contrast to other centrality methods; the nodes with high closeness centrality have shorter average paths to all other nodes, this ensures faster data arrival to the base station. That’s why Closeness centrality is used in this study.

In the procedure described above, the number of clusters is predetermined, which is not ideal for WSN clustering since the optimal number of clusters should be determined dynamically. Here is the used procedure used to find the optimal number of clusters using closeness centrality [32],[33],[34],[35]:

1. Centrality Calculation: For each node v_i in the graph, calculate the closeness centrality $r(v(i))$ as follows:

$$r(v(i)) = \frac{N-1}{\sum_{i \neq j} d(v(i),v(j))} \tag{19}$$

   where: N is the number of nodes in the graph.

   $d(v(i), v(j))$ is the shortest path distance between nodes i and j.

2. Apply Closeness Centrality: Run the Closeness Centrality algorithm described earlier on the graph for K (where K takes a value between K_{min} and K_{max} inclusive).
   - Cluster the graph into K clusters, $C_{K1}, C_{K2}, C_{K3}, \dots \dots, C_{KK}$
   - Identify the central node $\hat{C}_{Kk}$ for each cluster K based on the maximum centrality:

$$\hat{C}_{Kk} = \arg max_{v(i) \in C_{Kk}} r(v(i)) \tag{20}$$

3. Compute Cluster Compactness:
   - For each cluster k (under K), calculate the cluster compactness y_{Kk}:

$$y_{Kk} = \sum_{i \in C_{Kk}} d\left(v_{Kk}(i), \hat{C}_{Kk}\right)^2 \tag{21}$$

   - Get the average compactness for all clusters:

$$Y_K = \frac{1}{N}\sum_{k=1}^{K} y_{Kk} \tag{22}$$

4. Average Central Error (ACE) Calculation:
    - For each cluster k, using its mean and variance, calculate the central error Z_{Kk}

$$Z_{Kk} = Y_K - \frac{N_k - 2}{N_k}\sigma^2 \tag{23}$$

where: Z_{Kk} is ACE of cluster k.

N_k is the number of nodes in cluster k.

σ^2 is the variance variation.

- Calculate the Average Central Error (ACE) across all clusters Z_K

$$Z_K = \sum_{k=1}^{K} N_k \, Z_{Kk} \tag{24}$$

5. Choose the optimal number of clusters: Using the ACEs, the optimal number of Clusters K^* that minimizes the ACE.

$$\widehat{K} = \arg min_K \, Z_K \tag{25}$$

6. Final clusters are produced:

$$\hat{C} = \{\hat{C}_1, \hat{C}_2, \hat{C}_3, \dots \dots, \hat{C}_{\widehat{K}}\} \tag{26}$$

4. Methodology

This section discusses the methodology used to integrate the LEACH-C protocol with three clustering methods; K-means, Mean-Shift and Closeness Centrality. Then evaluate their effectiveness in enhancing energy efficiency and network lifetime. A detailed simulation program was developed for integrating LEACH-C with these clustering techniques under various data rates and node distributions.

It is divided into two subsections: the first outlines the key assumptions made during the simulation, while the second details the unified algorithm developed to monitor network performance across the different clustering techniques.

4.1 Assumptions

In order to integrate LEACH-C with K-Means, Mean-Shift and Centrality clustering, then study their impact on the Wireless Sensor Network lifetime, a simulation methodology was used. Taking into accounts the following key assumptions:

1. A WSN may be uniformly distributed [36], i.e. the nodes are evenly spaced in a structured grid or pattern, such as smart agriculture where sensors are placed at almost fixed distances to monitor soil moisture, temperature and humidity. Also, a WSN may be not be uniformly distributed [36], i.e. the nodes are placed irregularly, often due to environmental constraints or application-specific needs, such as in precision agriculture where sensors are placed more densely in areas prone to drought or disease, while fewer sensors are placed in stable regions. The built algorithm was checked on both cases, to find if there is a deviation while using each.

2. Power control is a technique used to adjust a device's transmission power based on the distance to the receiving node, optimizing the communication network performance [37] and minimizing the power consumption, this extends the battery life; especially in wireless sensor networks and mobile devices; while maintaining adequate signal strength for reliable communication and reducing errors.

3. The energy model [6],[7] used in this study is based on the radio energy dissipation model, which accounts for the energy consumed by both transmission and reception of data in a wireless sensor network. Power control is used to adjust transmission power based on the distance to the receiver. The model considers two primary energy dissipation components:

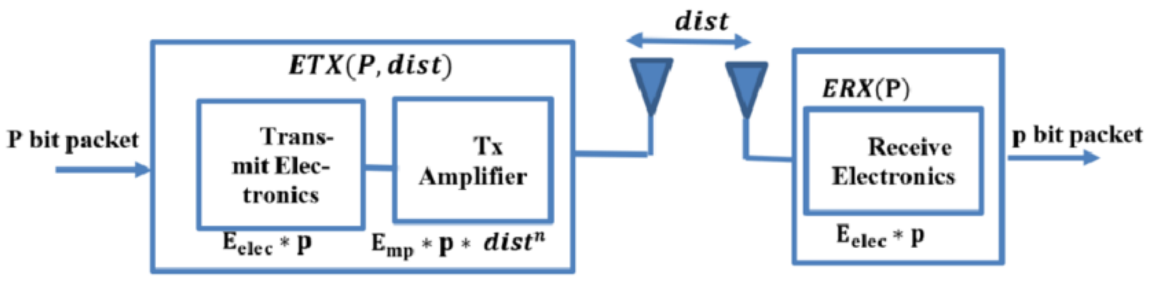


Figure 2: Radio Energy Dissipation Model

- **Transmitter Energy Consumption:** the transmitter expends energy on radio electronics and power amplification. The transmitter adjusts the transmitting power based on the distance to the receiver. Two factors are considered while calculating the transmitter energy consumption. The first factor is the electronic energy (E_{elec}), which depends on many factors such as the digital coding, modulation, filtering and spreading of the signal, while the second factor is the amplification energy (E_{amp}), which depends on the distance to the receiver and the acceptable bit-error rate. If the distance is less than a threshold (d_O) value then the free-space model is used, otherwise the multipath-fading model is used. In other words; the distance-based power loss model assumes different path loss exponents (2 for free space and 4 for multipath fading) depending on distance. So, the energy used to transmit a p-bit packet over a distance d is calculated using this equation:

$$E_{Tx}(p,d) = \begin{cases} p * E_{elec} + p * E_{fs} * d^2, & d < d_o \ (free - space\ model) \\ p * E_{elec} + p * E_{amp} * d^4, & d \geq d_o \ (multi - path\ model) \end{cases} \tag{27}$$

where: $E_{Tx}(p,d)$ is the energy consumed to send a p-bit packet over a distance d.

E_{elec} is the energy consumed per bit by the transmitter or the receiver.

E_{fs} and E_{amp} are the amplifier parameters of transmission corresponding to the free-space model and the multi-path fading model, respectively.

d_o is a threshold distance that determines when to switch between models, which is calculated using this equation:

$$d_O = \sqrt{\frac{E_{fs}}{E_{amp}}} \tag{28}$$

- **Receiver Energy Consumption**: the receiver consumes energy only for the radio electronics. So, the energy required to transmit a p-bit packet over a distance d is given by

$$E_{Rx}(p,d) = p * E_{elec} \tag{29}$$

where: $E_{Rx}(p,d)$ is the energy consumed to receive a p-bit packet over a distance d.

As reception does not involve amplification, the reception power is lower than that for transmission.

4. To accurately compare the effects of K-Means, Mean-Shift and Centrality clustering on WSN lifetime, it is essential to use consistent inputs. Therefore, the same packet file should be used across all methods. We adopted a standard approach to simulate packet arrivals using a Poisson process [38],[39],[40] with an exponential inter-arrival time distribution, following these steps:
    1. Generate a uniform random variable y in the interval [0,1]. This random variable is used to generate inter-arrival times that follow an exponential distribution.
    2. Compute the inter-arrival time:

$$Inter - arrival\ Time = -\frac{1}{\lambda}\ln(1 - y) \tag{30}$$

where: λ is the packet arrival rate (packets per second).

y is the random generated number between 0 and 1.

Note that using 1- y instead of y to avoid issues with ln (0), which is undefined. The negative sign ensures positive inter-arrival times since ln(1−y) is negative.

3. Starting from an initial time (typically 0), the next packet arrival time is calculated as follows:

$$\text{New Time} = \text{Old Time} + \text{Inter-arrival Time} \qquad (31)$$

5. Packets are aggregated in Wireless Sensor Networks (WSNs) to improve energy efficiency, reduce network congestion and enhance data processing efficiency [41],[42]. Aggregation may be done either by number of packets or by time interval. Aggregating by the number of packets has the advantage of ensuring a fixed number of packets are combined, maximizing data utilization. However, if the packet generation rate is low, it can lead to significant delays. On the other hand, aggregation by time interval offers a predictable delay, which is good for synchronization, but it has a main drawback; if the interval is too short, there may not be enough packets to aggregate effectively. On the other hand, if the interval is too long, it can lead to long delays. Therefore, we can conclude that aggregating by the number of packets is effective when the packet generation rate is stable and predictable. Otherwise, it is better to use aggregation by time interval.
   Both methods are applicable in our simulation.

6. As mentioned earlier; LEACH (Low-Energy Adaptive Clustering Hierarchy) uses a round-based approach to select cluster heads which helps distributing energy usage across the network, in order to extend the overall network lifetime. Similarly, this study adopts a round-based approach; however, since a centralized method is used, the selection is not based on probability like in LEACH using a Chosen_as_CH flag for each node to be set once the node was used as CH. The flags of all nodes will be reset once every node has served as a Cluster Head (CH).

4.2 Algorithm

To evaluate the network performance using the three different clustering methods, a unified algorithm was developed for LEACH-C. It monitors the packet generation and transmission by each node to its respective CH and tracks the packets sent by each CH to the base station. Additionally, it manages the commands issued by the base station to define clusters … etc. Of course, the clustering process itself varies depending on the method used. Also, a common network topology must be used for each run of the simulation program. So, both the packets' list (packet number, source node and generation time) and the network's topology will be generated in separate of the simulation algorithm to be commonly used for all the three clustering methods to ensure getting accurate comparisons.

For the simulation, a secondary list similar to the original one will be generated. This list will be expanded while running the program based on time.

During the simulation process, we will primarily focus on the following four events:

1. **New packet generation:** This occurs based on the generated packets' list, indicating the creation of a new packet.
2. **New packet arrival to the cluster head**: This happens when a node sends a packet to its cluster head.
3. **Aggregation:** a CH aggregates multiple packets and sends the resulting aggregated packet to the base station.
4. **Cluster generation:** form new clusters based on the cluster update interval.

Figures 3 to 8 demonstrate the algorithms utilized in this study.

```
Algorithm 1: LEACH-C Algorithm
Inputs: • N: Total number of sensor nodes in the network
        • BS: Base Station location
        • Initial-Energy of each node in the network
        • Control Packet Size
        • Data Packet size
        • Cluster Update Interval: time interval used to form new clusters.
        • Aggregation Threshold: Number of packets to be combined for aggregation.
        • $E_{elec}$: Electronic Energy
        • $E_{fs}$: Amplification Energy for the free-space model
        • $E_{amp}$: Amplification Energy for the multipath model
        • $E_{aggr}$: Aggregation Energy
        • Packets' List generated using the algorithm in figure 4.
        • Network Topology: Nodes' Coordinates using the algorithm in figure 5.
```

Outputs:
- Death Time of each Node.
- Number of Processed Packets before the Death of all the Network's Nodes.

Procedure:

- Create the secondary list described earlier.
- The base station executes one of the clustering algorithms (described in figures 6, 7 and 8) to form new clusters.
- The base station broadcasts a control packet to inform all nodes of the updated cluster structure.
- Set the Chosen_as_CH flag of the nodes chosen as CH.
- The head of the cluster broadcasts a control packet to be used by the clusters' members for power control.
- Update the energy of all the nodes using equations (27), (28) and (29)

Repeat

Check the four event's status time described earlier, then acts accordingly:

1- **New packet generation time:** obtained from the original list.
   - Get the packet's source node.
   - Get the packet's generated time.
   - Get the cluster to which the source node belongs.
   - Send this packet to this cluster's CH.
   - Calculate the updated time as follows:
     Updated time = generated time + transmission time + queueing time (if exist).
   - Add this packet as a new record in the appropriate position of the secondary list (based on the updated time), marking it as a (new packet arrival to the cluster head).
   - Update the energy for both the source node and the CH using equations (27), (28) and (29).
   - Verify if any of the two node is out of energy: Increase the number of dead nodes by one, and record the time of its failure.

2- **New packet arrival time to the cluster head**: the time when a packet sent by a node reaches its cluster head.
   - Enqueue the packet in the CH and update its occupied time.
   - Increase the number of enqueued packet in the CH.
   - Check if the aggregation time was reached.

3- **Aggregation Time:** occurs if the number of packets enqueued in a CH, reached the specified threshold for aggregation.
   - Aggregate the packets enqueued in the CH.

- Send the aggregated packet to the base station.
- Increase the number of processed packets by this number.
- Update the energy of this CH using equations (27) and (29)
- Verify if the CH is out of energy: Increase the number of dead nodes by one, and record the time of its failure.

4- **Cluster generation time:** this event is reached based on the cluster update interval, i.e. when the time to form new clusters based on the cluster update interval is reached.

- If all the nodes' Chosen_as_CH flag were set, unset them.
- Each node sends a control packet to its cluster's CH including its residual energy.
- Each CH aggregates the control packets and sends the aggregated packet to the base station.
- Also, each CH sums the residual energy of all the non-dead nodes in the cluster and sends these two numbers to the base station.
- The base station set the CH Energy Threshold (the minimum required energy for a node to be eligible for selection as a CH) to the average of the residual energy of the whole non-dead nodes.
- The base station executes one of the clustering algorithms (described in figures 6, 7 and 8) to form new clusters.
- The base station broadcasts a control packet to inform all nodes of the updated cluster structure.
- Set the Chosen_as_CH flag of the nodes chosen as CH.
- Update the energy of all the nodes using equations (27), (28) and (29).
- The head of the cluster broadcasts a control packet to be used by the cluster' members for power control.
- Verify if any node is out of energy: Increase the number of dead nodes by one, and record the time of its failure.

Until all nodes are dead

Return • The death time of each node.
• The number of processed packets before the death of all the networks' nodes.

Figure 3: LEACH-C Algorithm

Algorithm 2: Packet Generation Algorithm

Inputs: • N: Number of nodes

```
            • Packet rate.
            • End time

Outputs:  • List of Packets.

Procedure:
  for (i=1 to N)
    Repeat
      Generate packets using equations (30) and (31).
    Until End time
  end for
  Sort the generated packets by the generated time. The resulted list includes the packet number,
  the source node and the generation time.
Return The generated sorted packets' list.
```

Figure 4: Generation Packets Algorithm

```
Algorithm 3: Network Topology Algorithm
Inputs: • N: Number of nodes
        • Network Size
        • Type of Network Topology: Nodes are uniformly or non-uniformly distributed

Outputs:  • The nodes' coordinates.

Procedure:
  for (i=1 to N)
    Generate random coordinates taken into consideration the network's topology type.
  end for
Return The nodes' coordinates.
```

Figure 5: Network Topology Generation Algorithm

Algorithm 4: K-means Algorithm

```
Inputs: • N: Number of alive sensor nodes (nodes with energy) together with their locations.
        • N*: Nodes that can be used as clusters in this iteration (the ones with energy exceeding
          the CH Energy Threshold).
        • $K_{min}$ & $K_{max}$: Minimum and Maximum number of clusters respectively.
        • Threshold: Convergence Threshold.

Outputs: • C: Cluster heads (centroids) $C \in N^*$.
         • Cluster assignment for each node $n \in N \rightarrow C$

Procedure:
  for (K=$K_{min}$ to $K_{max}$)          // K is the number of clusters (centroids).
      Select K random nodes from $N^*$ as initial centroids C, taking into account their
      locations.
      Repeat
          for (n=1 to N)
              Assign node $n_i$ to the nearest centroid $c_j$ using equation (1).
          end for
          for (j=1 to K)
              Update the centroid $c_j$ as the mean of all assigned points using equation (2), such
              that the new $c_j \in N^*$
          end for
          Check for Convergence (centroids changes are very small) using equation (3).
      Until Convergence
      Cluster Heads (CHs) are selected, and nodes are assigned to them.
              $C_K = \{C_1, C_2, C_3, \dots \dots, C_K\}$
      for (n=1 to N)
          Calculate the Silhouette Coefficient using equation (5).
      end for
      Calculate the average Silhouette Score using equation (6).
  end for
  The optimal cluster size $\widehat{K}$ is determined using equation (7).
  The cluster heads are obtained $\hat{C} = \{\hat{C}_1, \hat{C}_2, \hat{C}_3, \dots \dots, \hat{C}_{\widehat{K}}\}$ together with the node's
  assignment.
Return: The clusters' structure; the CHs and their assigned nodes.
```

Figure 6: K-means Algorithm

```
Algorithm 5: Mean-Shift Algorithm
Inputs: • N: Number of alive sensor nodes (nodes with energy) together with their locations.
        • N*: Nodes that can be used as clusters in this iteration (the ones with energy exceeding
              the CH Energy Threshold).
        • h: Bandwidth.
        • Threshold: Convergence threshold to check centroid stability.
        • M_Threshold: Merge threshold to combine close clusters

Outputs: • C: Cluster heads (centroids) $C \in N^*$.
         • Cluster assignment for each node $n \in N \rightarrow C$

Procedure:
    Calculate h using equation (15).
    Set C= $\emptyset$ (empty set of centroids).
    Assign random nodes $n_i \in N^*$ as initial centroids $c_i$.
    Repeat
        for (n=1 to $N^*$)
            Compute the mean of all points within a given bandwidth (distance) to this
                point using equation (9).
            Update the new centroid (mean) location using equation (10).
        end for
        Check for Convergence (centroids changes are very small) using equation (11).
    Until Convergence
    for (i=1 to Number of formed centroids)
        Check for the need for merging this centroid to others using equation (12)
        If there are close centroids, merge them using equation (13).
        If there are no close centroids, add Ci to C
    end for
    for (n=1 to N)
        Assign each node to the nearest Cluster head $C_i$ using equation (14).
    end for
    The cluster heads are obtained $C = \{C_1, C_2, C_3, \dots \dots, C_K\}$ together with each node's
    assignment.
Return: The clusters' structure; the CHs and their assigned nodes.
```

Figure 7: Mean-Shift Algorithm

Algorithm 6: Closeness Centrality Algorithm

```
Inputs: • N: Number of alive sensor nodes (nodes with energy) together with their locations.
        • N*: Nodes that can be used as clusters in this iteration (the ones with energy exceeding
              the CH Energy Threshold).
        • $K_{min}$ & $K_{max}$: Minimum and maximum number of clusters respectively.

Outputs:  • C: Cluster heads (centroids) $C \in N^*$.
          • Cluster assignment for each node $n \in N \rightarrow C$

Procedure:
    Construct the graph representing the network
    for (K=$K_{min}$ to $K_{max}$)                // K is the number of clusters (centroids).
        for (n=1 to N)
                Calculate the closeness centrality $r(v(i))$ using equation (19).
        end for
        Use any method to get initial CH and their assigned nodes.
        for (Cluster= 1 to K)
                Identify the central node $\hat{C}_{Kk} \in N^*$ using equation (20).
                Calculate the cluster compactness $y_{Kk}$ using equation (21).
        end for
        Calculate the average compactness $Y_K$ for all clusters using equation (22).
        for (Cluster= 1 to K)
                Calculate its Average Central Control s $Z_{Kk}$ using equation (23).
        end for
        Calculate the Total Average Central Error (ACE) $Z_K$ using equation (24).
    end for
    The optimal cluster size $\widehat{K}$ is determined using equation (25).
    The cluster heads are obtained $\hat{C} = \{\hat{C}_1, \hat{C}_2, \hat{C}_3, \dots \dots, \hat{C}_{\widehat{K}}\}$ together with the node's
    assignment.
Return: The clusters' structure; the CHs and their assigned nodes.
```

Figure 8: Closeness Centrality Algorithm

5. Results and Analysis

The algorithms described in the previous section was built using C++. The comparison was conducted using different data rates and different networks with both uniformly and non-uniformly distributed nodes. This section presents the performance metrics the comparison results and their corresponding analysis.

5.1 Performance Metrics

The following metrics were used to evaluate the networks' performance while running the built integrated protocol described earlier.

5.1.1 Various Nodes' Death Times

From the algorithms described above, we can accurately measure the exact time when each node's death (runs out of energy). However; it would be impractical to create graphs for every required comparison, including data rate changes and topology variations, so a selection of representative results is presented. We will plot the time of death for the first node and the last node. Additionally, we will graph the deaths of 25%, 50%, and 75% of the network's nodes as representative points.

5.1.2 Number of Data-Packets Processed

By using various packet rates, topologies, and clustering methods, it is difficult to predict when all the network's nodes will fail (run out of energy). Therefore, we generate a large set of packets and observe how many can be processed before the network's complete failure. It can be obtained will running the algorithms described in the previous section.

Table 1: Common Simulation Parameters

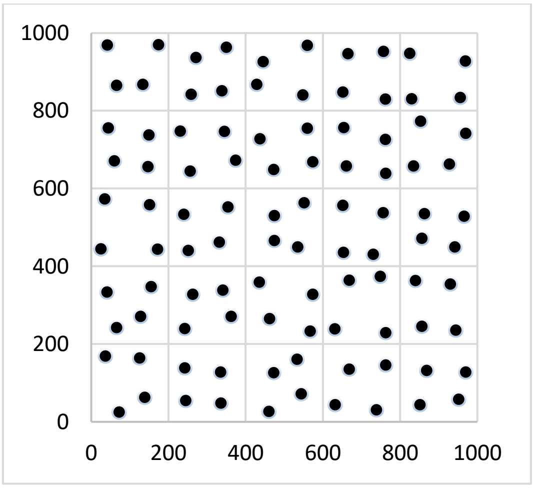
Description	Value
Network Area	1000 m * 1000 m
Total Number of Sensor Nodes in the Network	100 nodes
Base Station Location	(500,500)
Packets' Rate	varies from 25 to 150 packets/sec
Initial-Energy of each Node in the Network	3 Joules
Electronic energy (E_{elec})	50 nJ/bit

Amplification Energy (E_{amp}) for the multipath model	0.0013 pJ/bit/m^4
Amplification Energy (E_{fs}) for the free-space model	10 pJ/bit/m^2
Aggregation Energy	5 nJ/bit
Control Packet Size	200 bits
Data Packet Size	6400 bits
Cluster Update interval	5 sec
Transmission Time	0.0001 sec

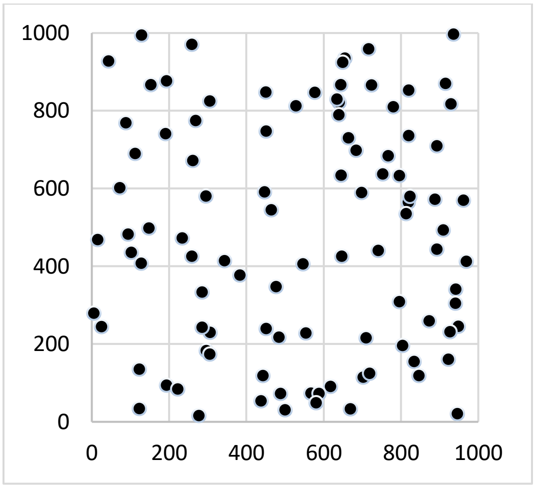
5.2 Results

The simulations commonly used for the comparison are outlined in Table 1. The scientific parameters, such as energy values, control and data packet sizes and others, were obtained from [43]. To ensure accurate results, each value in the following charts represents the average of five runs of the simulation program.

The used networks consist of a randomly generated set of nodes that are either uniformly or non-uniformly distributed [36]. Examples of these network distributions are shown in Figure 9.



(a): Almost Uniformly Distributed



(b): Non-Uniformly Distributed

Figure 9: Examples of the Used WSN's Topology

5.2.1 Changing Packets' Rate

This subsection describes and analyses the results obtained running the above algorithms while changing the network's packets rate from 25 to 150 packets/sec. Figure 9 describes the uniformly and non-uniformly distributed networks.

Figures 10 to 13 illustrate these results. Of course, as the data rate increases from 25 to 150 packets/sec, there is an increase in energy consumption across all the clustering methods. This higher transmission rate leads to quicker energy depletion, resulting in shorter node lifespans and reduced network lifetime.

It is clear that there are differences in results between uniform and non-uniform distributions, which happens due to the inherent nature of each distribution and how clustering methods respond to these differences.

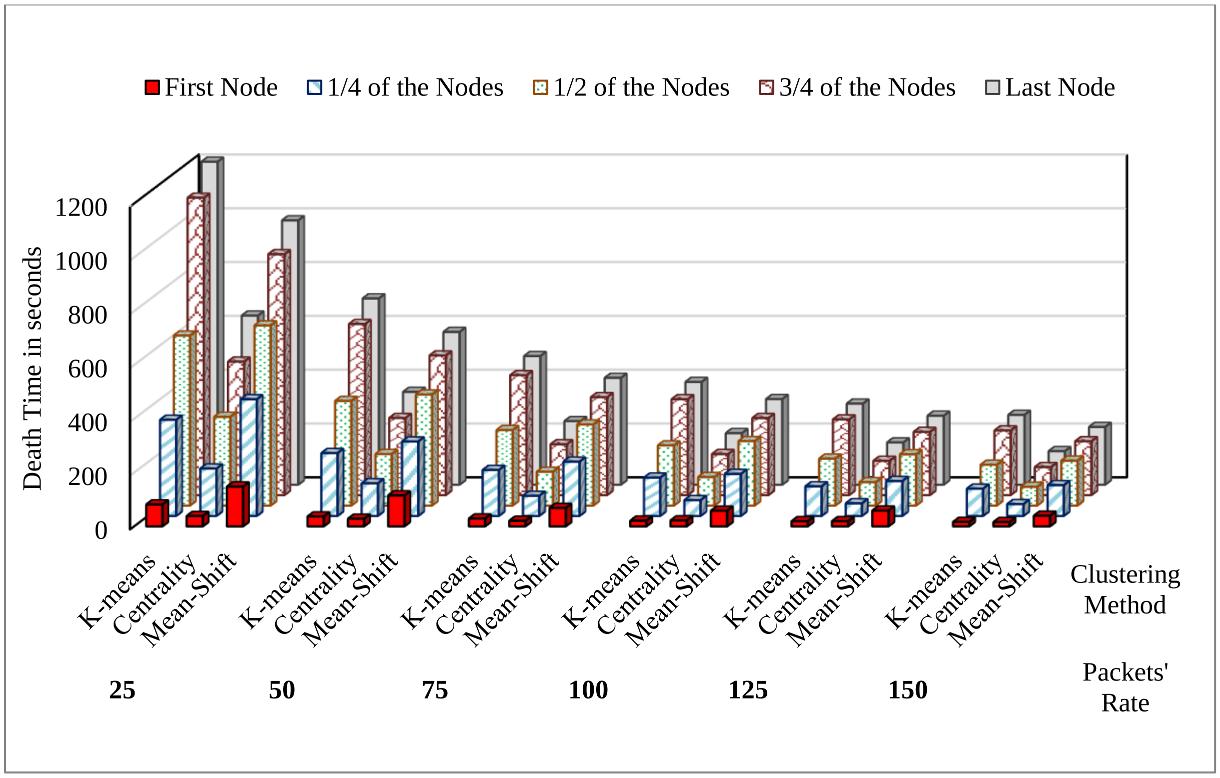


Figure 10: Death Times of Nodes (First Node, 25%, 50%, 75%, Last Node) for the Integrated LEACH-C Protocol Using the Three Clustering Methods under Various Data Rates in Uniformly Distributed WSNs

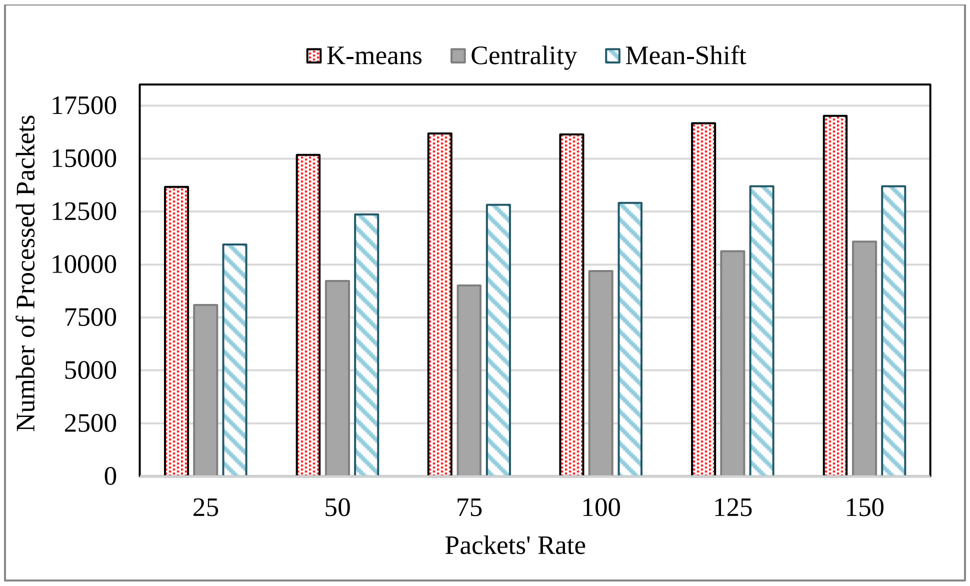


Figure 11: Number of Data-Packets processed

for the Integrated LEACH-C Protocol Using the Three Clustering Methods

under Various Data Rates in Uniformly Distributed WSNs

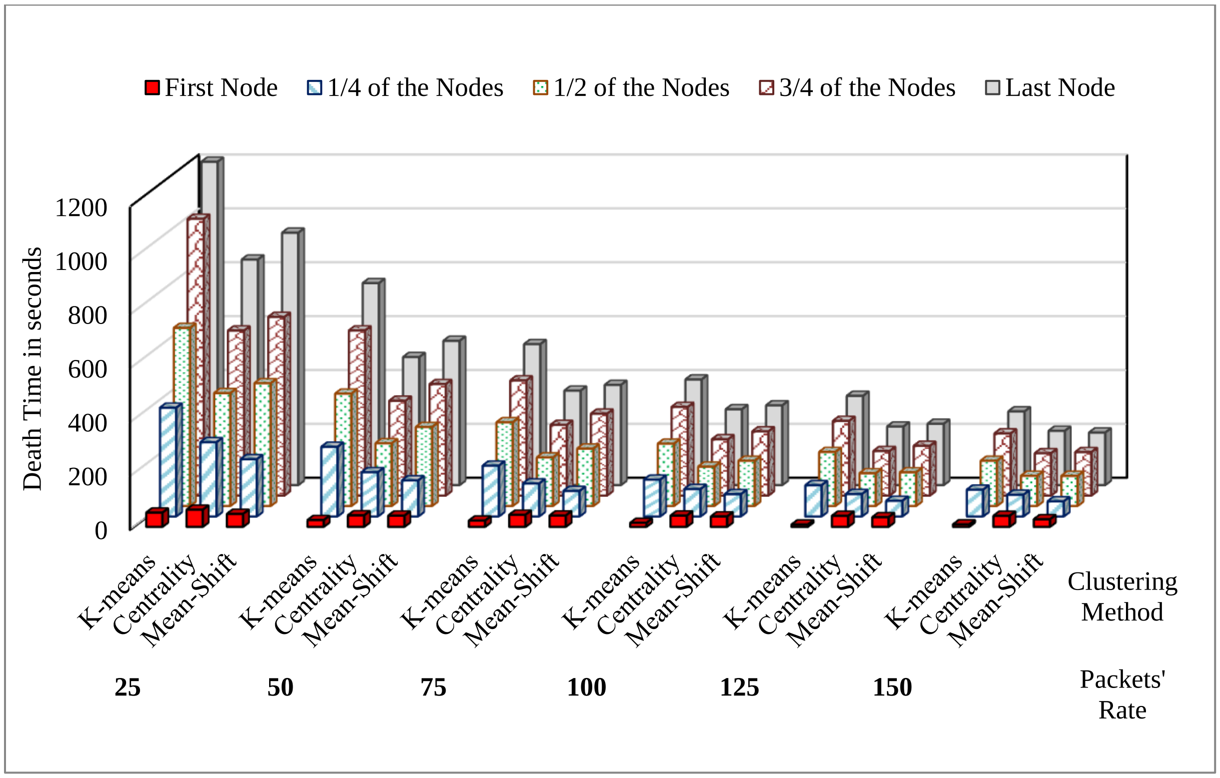


Figure 12: Death Times of Nodes (First Node, 25%, 50%, 75%, Last Node)

for the Integrated LEACH-C Protocol Using the Three Clustering Methods

under Various Data Rates in Non-Uniformly Distributed WSNs

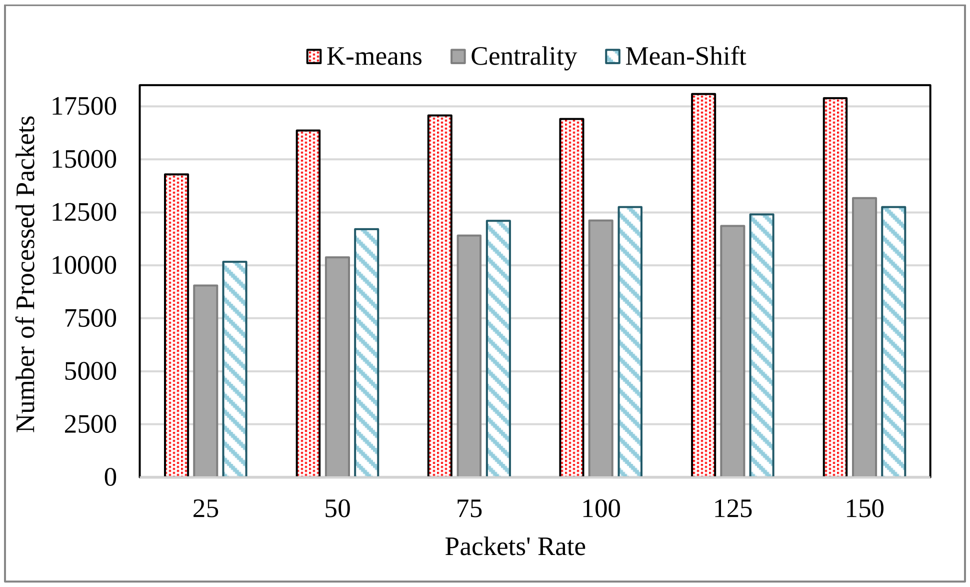


Figure 13: Number of Data-Packets processed
for the Integrated LEACH-C Protocol Using the Three Clustering Methods
under Various Data Rates in Non-Uniformly Distributed WSNs

The figures show that K-means clustering method consistently achieves longer the nodes' life times across all node percentages (first node to last node) compared to Mean-Shift and Closeness Centrality. The main reasons of this; K-means effectively balances the clusters, minimizing the load on individual nodes, which extends their lifespan. While Mean-Shift, being density-based, might form clusters with unequal sizes, leading to faster node death in dense regions. Also, Closeness Centrality mainly focuses on minimizing the paths' length, which can result in increased communication for central nodes of the network, reducing their lifespan. K-means consistently outperforms both Closeness Centrality and Mean-Shift. The improvement of K-means over Closeness Centrality ranges from 46.7% to 50%. Compared to Mean-Shift, K-means shows a moderate improvement ranging from 12.7% to 20%.

As a result of its longer network's life, K-means consistently processes the highest number of data packets across all data rates, followed by Mean-Shift, while Closeness Centrality processes the least. The improvement of K-means over Closeness Centrality ranges from 30.8% to 40%. Compared to Mean-Shift, K-means shows a moderate improvement ranging from 16.7% to 17.9%.

5.2.2 Changing Network’s Topology

To ensure that the obtained results are applicable to a variety of topologies and not limited to the previously used ones, we modified the network topology and run the algorithms again. For the comparison, we use a packets’ rate of 125 packets/sec.

Figures 14 to 17 illustrate the results using various topologies (uniform and non-uniform) under the chosen packets’ rate.

Across all network topologies, K-means achieves the longest network’s death time, compared to the other two methods. Naturally; it also processes the highest number of data packets across all topologies.

The differences between the clustering methods become more apparent with non-uniform topologies because such topologies create more diverse and irregular node distributions. This increased complexity can highlight variations in how each method manages communication overhead, load balancing and energy efficiency.

Of course, the improvement range varies across different topologies. For the death time, the improvement of K-means over Closeness Centrality ranges from 63.6% to 70% and 28.6% to 30.8% for Mean-Shift. While for the number of processed packets, the improvement of K-means over Closeness Centrality ranges from 44.4% to 48.0%, and 21.9% to 23.3% for Mean-Shift.

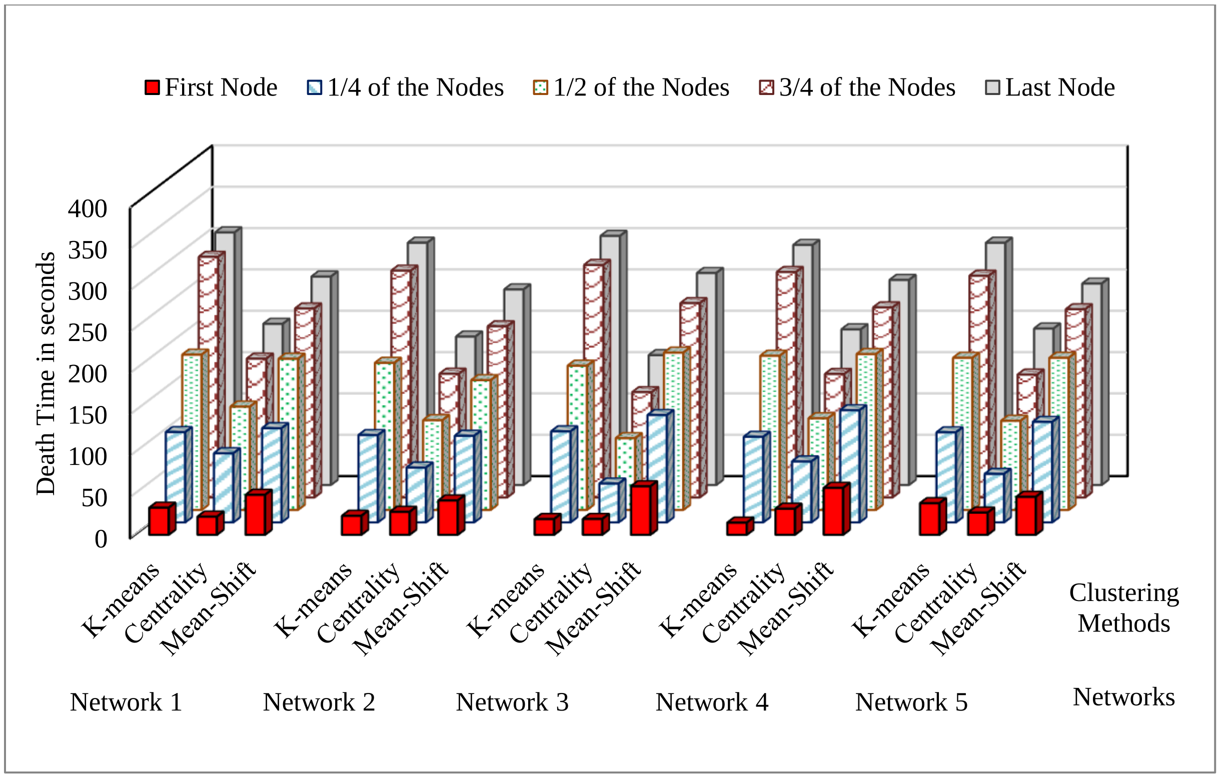


Figure 14: Node Death Times (First Node, 25%, 50%, 75%, Last Node) for the Integrated LEACH-C Protocol Using the Three Clustering Methods under Different Uniformly Distributed Network Topologies

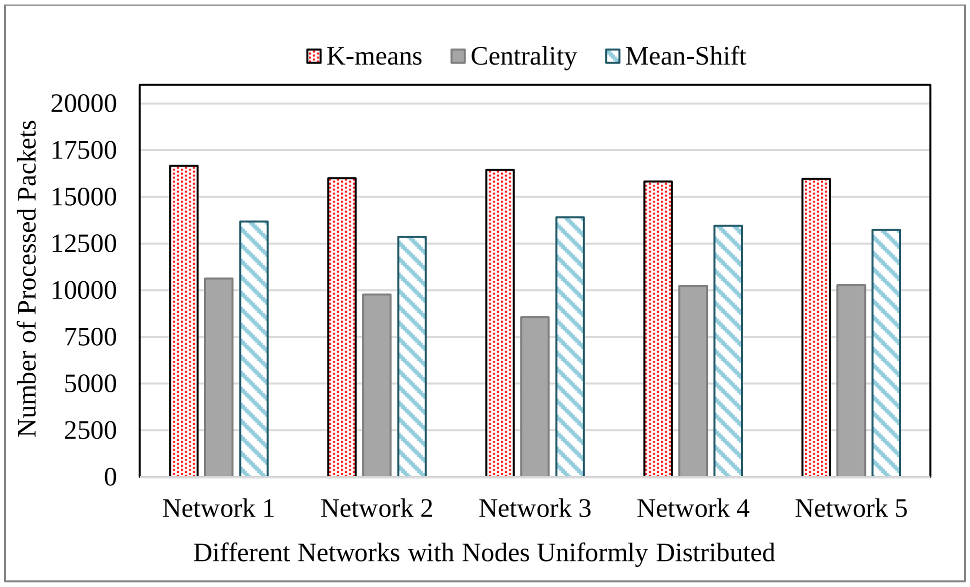


Figure 15: Number of Data-Packets processed for the Integrated LEACH-C Protocol Using the Three Clustering Methods

under Different Uniformly Distributed Network Topologies

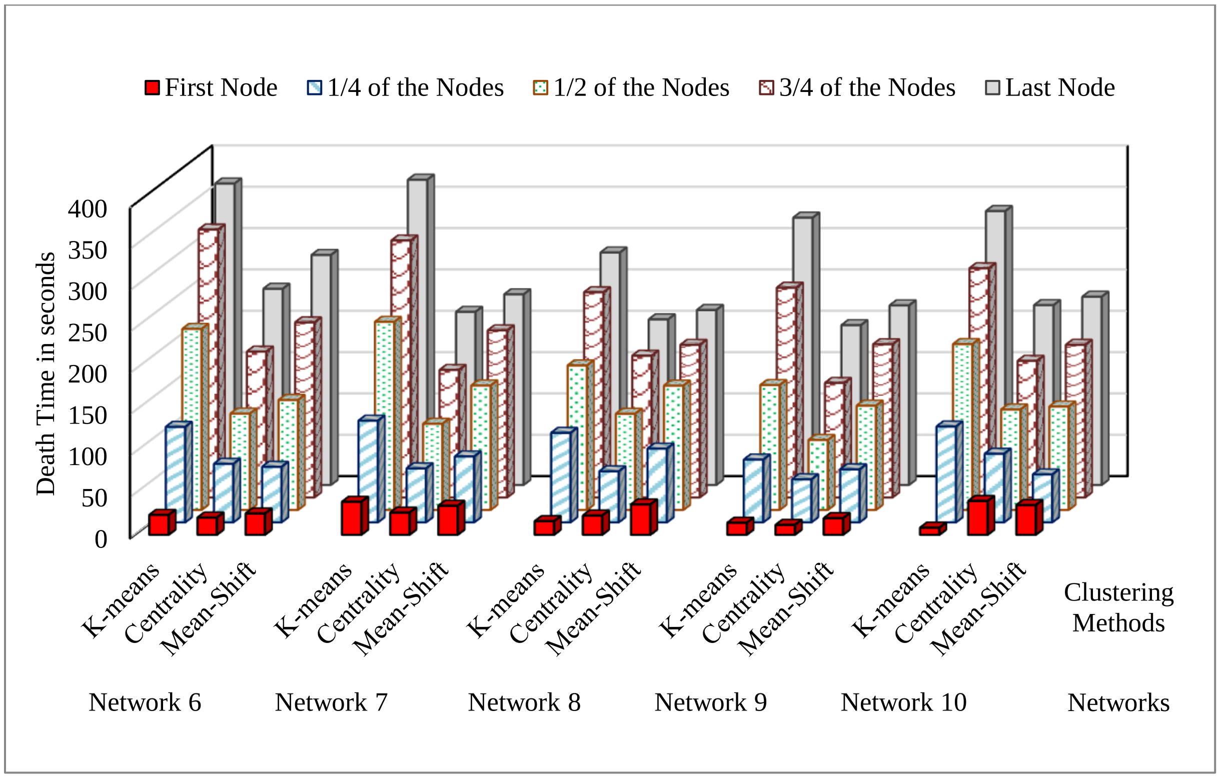


Figure 16: Node Death Times (First Node, 25%, 50%, 75%, Last Node) for the Integrated LEACH-C Protocol Using the Three Clustering Methods under Different Non-Uniformly Distributed Network Topologies

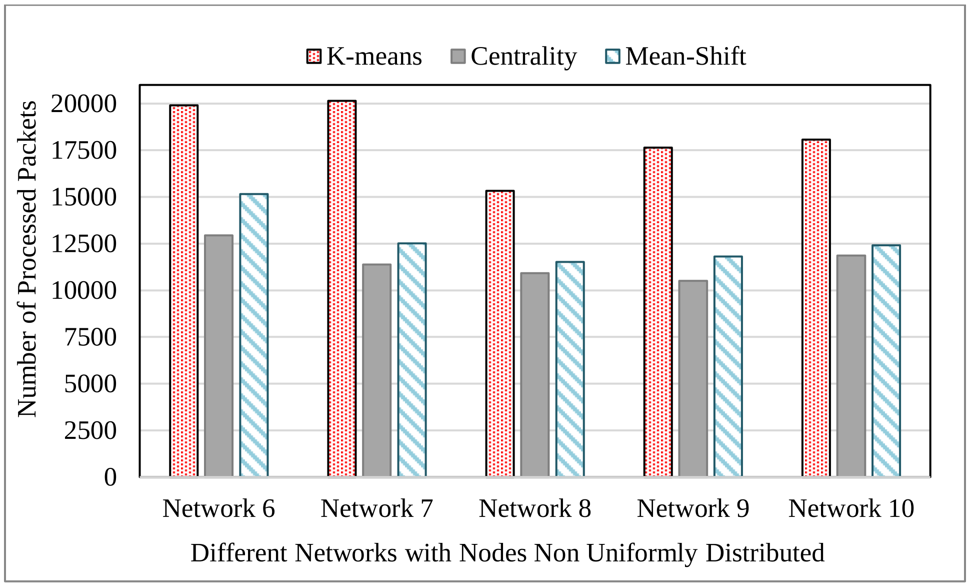


Figure 17: Number of Data-Packets processed

for the Integrated LEACH-C Protocol Using the Three Clustering Methods under Different Non-Uniformly Distributed Network Topologies

6. Conclusion and Future work

Clustering in Wireless Sensor Networks (WSNs) is essential for optimizing the energy consumption, extending the network lifetime and ensuring efficient data communication. By organizing sensor nodes into clusters, clustering reduces redundant transmissions, balances the workloads and minimizes the communication overhead. LEACH is one of the most well-known hierarchical clustering-based distributed routing protocols developed for WSNs. LEACH-C is a centralized improvement of LEACH. It utilizes centralized control by allowing the base station to form clusters based on nodes' energy levels and locations.

This paper proposed integrating the LEACH-C protocol with three widely recognized clustering methods: K-means, Mean-Shift and Closeness Centrality. For this purpose; a simulation program was developed and tested across various data rates and various network topologies, with uniformly and non-uniformly distributed nodes.

The results indicate that integrating LEACH-C with K-means outperforms the other two methods in terms of extending the network lifetime and maximizing the number of processed packets. Specifically, K-means demonstrated a percentage improvement in the death time, ranging from 46.7% to 50% over Closeness Centrality and 12.7% to 20% over Mean-Shift when the data rate was varied. Similarly, when changing the network topology, the improvement ranged from 63.6% to 70% over Closeness Centrality and 28.6% to 30.8% over Mean-Shift.

For the number of processed packets; K-means also demonstrated a percentage improvement ranging from 30.8% to 40% over Closeness Centrality and 16.7% to 17.9% over Mean-Shift while changing the data rate. When the network topology was changed, K-means achieved an improvement of 44.4% to 48% over Closeness Centrality and 21.9% to 23.3% over Mean-Shift.

For future work, we suggest exploring the impact of node mobility, so integrating K-means with machine learning techniques to optimize cluster head selection, improve scalability and achieve better energy balance in large-scale heterogeneous WSNs.

Additionally; we noticed that closeness Centrality and Mean-Shift sometimes perform better for the performance for the first nodes' death. As a suggestion for future work, a hybrid method that combines the strengths of these approaches with K-means could be explored to further optimize the network performance.

References

1. I.F. Akyildiz, W. Su, Y. Sankarasubramaniam & E. Cayirci, "Wireless sensor networks: a survey, Computer Networks Journal, vol.38, pp. 393 – 422, 2002.

2. Shweta Sharma & Amandeep Kaur, "Survey on Wireless Sensor Network, Its Applications and Issues", Journal of Physics: Conference Series, vol. 1969, 2021.

3. T. Soni Madhulatha, "An Overview on Clustering Methods", IOSR Journal of Engineering, vol. 2, no. 4, pp. 719-725, April 2012.

4. Dongkuan Xu & Yingjie Tian, "A Comprehensive Survey of Clustering Algorithms", Annals of Data Science Journal, vol.2, no.2, pp.165 -193, 2015.

5. Amin Shahraki, Amir Taherkordi, Øystein Haugen & Frank Eliassen, "Clustering objectives in wireless sensor networks: A survey and research direction analysis", Computer Networks journal, vol.180, 2020.

6. Wendi B. Heinzelman, Anantha P. Chandrakasan, & Hari Balakrishnan, "An Application-Specific Protocol Architecture for Wireless Microsensor Networks", IEEE Transactions on Wireless Communications, vol.1, no 4, pp. 660-670, October 2002.

7. Noha MM. Abdelnapi, Nahla F. Omran & Eman Mousa Mohamed, "Gateway based Multi-hop Enhanced Stable Election Protocol for WSN-based IoT", Research Square, October 2022. https://www.researchgate.net/publication/366487425_Gateway_based_Multi-hop_Enhanced_Stable_Election_Protocol_for_WSN-based_IoT, Last Accessed 22 March 2025

8. Christian Bauckhage, "Lecture Notes on k-Means Clustering", October 2013, https://www.researchgate.net/publication/262800457_Lecture_Notes_on_k-Means.Clustering_I, Last Accessed 22 March 2025.

9. Eric U. Oti1, Michael O. Olusola, Francis C. Eze & Samuel U. Enogwe, “Comprehensive Review of K-Means Clustering Algorithms”, International Journal of Advances in Scientific Research and Engineering (ijasre), vol. 7, pp. 64 -68, no. 8, pp. 790- 799, August 2021.

10. Dorin Comaniciu & Peter Meer, “Mean Shift: A Robust Approach Toward Feature Space Analysis”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.2, no. 5, pp.603-619, May 2002.

11. Yizong Cheng, “Mean Shift, Mode Seeking, and Clustering”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 17, no. 8, August 1995.

12. Stephen P. Borgatti & Martin G. Everett, “A Graph-theoretic perspective on centrality”, Social Networks Journal, vol.28, no.4, pp. 466-484, October 2006.

13. Péter Marjai, Bence Szabari & Attila Kiss, “An Experimental Study on Centrality Measures Using Clustering”, Computers Journal, vol.10, no.9, 2021.

14. “Wireless Sensor Network (WSN)”, last updated 18 June 2024, https://www.geeksforgeeks.org/wireless-sensor-network-wsn/, Last Accessed 22 March 2025.

15. Wendi B. Heinzelman, Anantha P. Chandrakasan & Hari Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Microsensor Networks", the 33rd Annual Hawaii International Conference on System Sciences (HICSS), 2000.

16. Khushboo Manohar & A.I. Darvadiya, “Study of Leach Protocol- A Review”, International Journal of Modern Trends in Engineering and Research”, pp. 401-407, 2014.

17. Wendi B. Heinzelman, Anantha P. Chandrakasan & Hari Balakrishnan, “An Application-Specific Protocol Architecture for Wireless Microsensor Networks”, IEEE Transactions on Wireless Communications, vol. 1, no. 4, pp.660-670, October 2002

18. Hongwei Chen, Chunhua Zhang, Xinlu Zong & Chunzhi Wang, “LEACH-G: An optimal cluster-heads selection algorithm based on LEACH”, Journal of Software, vol. 8, no. 10, pp.2660-2667, October 2013.

19. Mu Tong & Minghao Tang, “LEACH-B: An Improved LEACH Protocol for Wireless Sensor Network”, International Conference on Wireless Communications Networking and Mobile Computing (WiCOM), September 2010.

20. Maha Zayoud, H. M. Abdulsalam, A. Al-Yatama, and Seifedine Kadry, "Split and Merge LEACH Based Routing Algorithm for Wireless Sensor Networks," International Journal of Communication Networks and Information Security (IJCNIS), vol. 10, no. 1, pp. 155–162, 2018.

21. Heena Dhawan & Sandeep Waraich, "A Comparative Study on LEACH Routing Protocol and its Variants in Wireless Sensor Networks: A Survey", International Journal of Computer Applications, vol.95, no.8, pp.21-27, June 2014.

22. Asha Ahlawat & Vineeta Kumari, "An Extended Vice-Cluster Selection Approach to Improve V Leach Protocol in WSN", International Conference on Advanced Computing and Communication Technologies (ACCT), 2013.

23. Surender Kumar, M.Prateek, N.J.Ahuja & Bharat Bhushan, "DE-LEACH: Distance and Energy Aware LEACH", International Journal of Computer Applications, vol. 88, no.9, pp.36-42, 2014.

24. Divya Prabha & Vishal Kumar Arora, "A Survey on LEACH and its Descendant Protocols in Wireless Sensor Network", International Conference on Communication, Computing & Systems (ICCCS) - August 2014.

25. Sunil Kumar Singh, Prabhat Kumar & Jyoti Prakash Singh, "A Survey on Successors of LEACH Protocol", IEEE Access, vol.5, pp.4298-4328, February 2017.

26. Akanksha Vyas & Sachin Puntambekar, "Cluster Based Leach Routing Protocol and Its Successor: A Review", Journal of Scientific Research of The Banaras Hindu University, vol. 6, no.1, pp.326-341, 2022.

27. Chunhui Yuan & Haitao Yang, "Research on K-Value Selection Method of K-Means Clustering Algorithm", Multi-disciplinary Scientific Journal, vol. 2, no. 2, pp. 226-235, June 2019.

28. Rayan Yassminh, "Covering the Optimal Number of Clusters: Part 1 "Introduction & Background", February 2023, https://medium.com/@ryassminh/uncovering-the-optimal-number-of-clusters-part-1-introduction-background-79862df1d313, Last Accessed 22 March 2025.

29. Stanisław Węglarczyk, “Kernel density estimation and its application”, ITM Web of Conferences, vol.23, 2018.

30. Francis Bloch, M. Jackson & Pietro Tebaldi, “Centrality measures in networks”, Social Choice and Welfare Journal, vol.61, pp. 413-453, 2023.

31. Akrati Saxena & Sudarshan Iyengar, “Centrality Measures in Complex Networks: A Survey”, arXiv repository for electronic preprints of scientific papers, November 2020, https://arxiv.org/abs/2011.07190?utm_source=chatgpt.com, Last Accessed 22 March 2025.

32. Mahdi Shahbaba & Soosan Beheshti, “MACE-means clustering”, Signal Processing Journal , vol. 105, pp. 216-225, December 2014.

33. E.W. Nidoy. “k-MACE Clustering for Gaussian Clusters”, M.A.Sc Thesis, Ryerson University, 2016.

34. Faizan Rahman & Soosan Beheshti, “Kernel K-Mace Clustering”, 52nd Asilomar Conference on Signals, Systems and Computers, 2018.

35. Soosan Beheshti, Edward Nidoy & Faizan Rahman, “K-MACE and Kernel K-MACE Clustering”, IEEE Access, vol.8, pp.17390-17403, January 2020.

36. Fariba Aznoli & Nima Jafari Navimipour, “Deployment Strategies in the Wireless Sensor Networks: Systematic Literature Review, Classification, and Current Trends”, Wireless Personal Communications Journal, vol. 95, no. 2, pp. 819 – 846, July 2017.

37. Mohsen Guizani, “Wireless Communications Systems and Networks”, Kluwer Academic/Plenum Publishers, 2004.

38. “Poisson Distribution | Definition, Formula, Table and Examples”, Last Updated: Dec, 2024, http://geeksforgeeks.org/poisson-distribution/#poisson-distribution-formula, Last Accessed- 23 March 2025.

39. Aleksejus Kononovicius, “Poisson process: Interarrival times”, Physics of Risk, May 2023, https://rf.mokslasplius.lt/poisson-process-interarrival-times/.

40. Jing Zhao, Fan Zhang, Chao Zhao, Gang Wu, Haitao Wang & Xinyu Cao, “The Properties and Application of Poisson Distribution”, Journal of Physics: Conference Series, June 2020.

41. Saeid Pourroostaei Ardakani, “Data Aggregation Routing Protocols in Wireless Sensor Networks: A Taxonomy”, International Journal of Computer Networks & Communications (IJCNC) vol.9, no.2, pp.89-107, March 2017.

42. Arshpreet Kaur & Simarjeet Kaur, “A Review on Data Aggregation Techniques In Wireless Sensor Networks”, International Journal of Innovative Research in Science and Engineering, vol.2, no.5, pp.329-335, May 2016.

43. Emad Alnawafa and Ion Marghescu, “New Energy Efficient Multi-Hop Routing Techniques for Wireless Sensor Networks: Static and Dynamic Techniques”, Sensors Journal, no.6, vol.18, 2018.