BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **23**, No 4
Sofia • 2023 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.2478/cait-2023-0041

Exploring the Performance and Characteristics of Single Linkage and Complete Linkage Hierarchical Clustering Methods for IoT Sensor Networks

Fuad Bajaber

Department of Information Technology, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia
E-mail: fbajaber@kau.edu.sa

***Abstract*:** *The research explores applying hierarchical clustering methods, namely single linkage and complete linkage, in IoT Sensor Networks* (*ISNs*). *ISNs are distributed systems comprising numerous sensor nodes that collect data from the environment and communicate with each other to transmit the data to a base station. Hierarchical clustering is a technique that groups nodes into clusters based on proximity and similarity. This paper implements and compares the performance of single linkage and complete linkage methods in terms of cluster size, network lifetime, and cluster quality. The study's findings provide guidance for ISN researchers and designers in selecting the appropriate clustering method that meets their specific requirements.*

***Keywords*:** *Internet of Things* (*IoT*), *Hierarchical Clustering, Energy Efficiency, Single Linkage, Complete Linkage, Cluster Quality*

1. Introduction

Sensor and Internet of Things (IoT) sensor networks share similarities but differ in scope and objectives. Sensor networks are typically designed for specific, dedicated applications, often with localized data collection. These networks consist of sensor nodes that monitor and gather data from a particular environment, with the primary goal of data collection and transmission for a defined purpose. In contrast, IoT sensor networks are a subset of the broader IoT ecosystem. IoT sensor networks also use sensors for data collection but are part of a more extensive, globally connected network. IoT sensor networks are characterized by their ability to share data across vast geographical areas and communicate with cloud-based services, allowing for more extensive data processing and analytics on a global scale.

IoT Sensor Networks (ISNs) have emerged as a vital technology in various fields, including smart cities, environmental monitoring, surveillance, and healthcare. ISNs consist of numerous small, low-cost sensor nodes deployed in a specific area to collect and transmit data wirelessly [1-5]. Efficiently organizing the sensor data is

crucial to extract valuable insights and make informed decisions. Hierarchical clustering is a powerful technique used in machine learning to group similar data points into clusters [6-11].

One primary purpose of clustering is to ensure load balancing within the network. The network load is distributed more evenly by forming clusters, each with its Cluster Head (CH) responsible for data aggregation and communication. This allocation of responsibilities prevents specific sensor nodes from becoming overwhelmed with data processing and communication tasks while others remain underutilized. Consequently, load balancing enhances network performance and data reliability.

Furthermore, clustering in ISNs seeks to minimize energy consumption, which is of paramount importance due to the limited battery power available to sensor nodes. By organizing the network into clusters and employing cluster heads for data aggregation and forwarding, nodes can engage in energy-efficient data communication. This optimization results in significant energy savings, extending the operational lifespan of individual sensor nodes and the entire network.

Hierarchical clustering methods provide a hierarchical structure of clusters, allowing for a comprehensive exploration of the data at different levels of granularity. Two commonly used approaches in hierarchical clustering are single linkage and complete linkage. Single linkage clustering, also known as minimum distance clustering, merges clusters based on the minimum distance between any two points within each cluster. On the other hand, complete linkage clustering, also known as maximum distance clustering, merges clusters based on the maximum distance between any two points within each cluster. Exploring the performance and characteristics of single linkage and complete linkage hierarchical clustering methods in ISNs is crucial to understanding their effectiveness and applicability in this domain. By analyzing these methods, researchers can gain insights into their strengths and limitations, enabling them to make informed decisions when choosing the appropriate clustering technique for specific ISN applications [12-17].

Evaluating the performance and characteristics of single linkage and complete linkage clustering involves several aspects. The quality of the resulting clusters, such as their coherence and separation, is essential. These help assess the compactness, separation, and stability of the clusters formed by each method. ISNs often operate under resource-constrained environments, and the scalability and efficiency of clustering algorithms play a significant role. Understanding the requirements of single linkage and complete linkage clustering methods enables researchers to select the most suitable approach based on the available resources and the size of the ISN [18-23].

By exploring the performance and characteristics of single linkage and complete linkage hierarchical clustering methods in ISNs, researchers can enhance their understanding of each method's clustering behavior, advantages, and limitations. In this paper, we implement and investigate using single linkage and complete linkage hierarchical clustering methods in ISNs. We compare the performance of these methods in terms of cluster size, network lifetime, and cluster quality.

The primary purpose of this study is to provide a comprehensive understanding of the strengths and limitations inherent in single linkage and complete linkage hierarchical clustering methods as applied to ISNs. By offering insights into these clustering techniques, we aim to empower researchers and practitioners with the knowledge to make informed decisions when selecting clustering methods for their unique ISN applications. This knowledge will optimize data analysis, enhance decision-making processes, and maximize the utility of ISNs across diverse domains.

2. Literature review

In [24], authors propose a hybrid clustering algorithm to enhance sensor network energy efficiency. The algorithm incorporates two clustering techniques, one for initial network clustering and the other for identifying cluster leaders. Additionally, the authors present a novel hierarchical routing scheme that includes both flat and hierarchical routing. The algorithm modifies the cluster head selection and routing procedure dynamically based on the remaining energy of the nodes. This adaptive strategy aims to distribute energy consumption among nodes equitably, increasing the network's lifespan. The authors assess the algorithm's efficacy through exhaustive simulations and comparisons with existing protocols. The outcomes demonstrate considerable improvements in network lifetime, energy consumption, and data transmission effectiveness. The algorithm effectively distributes the energy load among nodes and reduces the number of transmissions, thereby enhancing the sensor network's overall performance. The authors could consider conducting additional experiments on more extensive networks to understand algorithm performance comprehensively.

The authors in [25] have presented a method for dynamic clustering in intelligent and eco-friendly Internet of Things applications. Their algorithm creates dynamic clusters by considering power demand and information volume. To form clusters, it follows a two-step process. First, it identifies and groups nodes that consume high amounts of energy. Second, it clusters nodes with low energy consumption based on their information volume. The simulation results demonstrate that this algorithm performs better than existing clustering methods in terms of energy efficiency, network lifetime, and network stability.

In their study, A n u r a g and T r i p a t h i [26] have proposed a multi-tier based clustering framework for Wireless Sensor Network-assisted Internet of Things (IoT) networks. This algorithm addresses the challenges of scalability and energy efficiency in large-scale IoT deployments. The framework comprises three tiers: central, intermediate, and peripheral. In the central tier, a master cluster head is selected to oversee the entire network and facilitate communication between the tiers. The intermediate tier consists of cluster heads responsible for data aggregation and transmission within their respective clusters. The peripheral tier consists of ordinary sensor nodes that collect and transmit data to the cluster heads. The authors have evaluated the framework's performance using simulations and compared it with other clustering methods. The results demonstrate that the proposed framework achieves better scalability, energy efficiency, and network lifetime than traditional clustering

methods. The authors should elaborate on factors like network size affecting cluster size decisions.

A Fault-tolerant Algorithm is presented in [27] to address the issues related to maintaining fault tolerance and energy efficiency. The algorithm concentrates on extending the network's lifecycle by organizing sensor nodes into clusters and utilizing a hierarchical data transmission structure. The algorithm employs a two-level clustering mechanism to cluster nodes efficiently. Nodes at the first level are organized into primary clusters, each led by a primary cluster chief. These primary clusters are subsequently combined into superclusters at a secondary level. The chiefs of the supercluster are responsible for aggregating and transmitting data to the base station. To ensure defect tolerance, the algorithm employs a dynamic cluster head selection mechanism that considers both the nodes' remaining energy and communication quality. This adaptive strategy enables the algorithm to modify the cluster head selection procedure in response to node failures or energy depletion, thereby ensuring uninterrupted network operation even in the presence of defects. The algorithm should be designed to handle node failures and changes in network topology.

In [28], authors have proposed a routing protocol that addresses the challenges of link lifetime and energy consumption. The protocol incorporates link lifetime and energy consumption prediction models to estimate network links' remaining lifetime and energy availability. It utilizes multi-path routing to enhance network reliability and load balancing by enabling simultaneous data transmission through multiple paths. By considering both link lifetime and energy consumption, the protocol dynamically selects the optimal path for data transmission based on these predictions. This approach aims to optimize resource utilization and improve the system's overall performance.

3. Methodology

Hierarchical clustering is a popular unsupervised learning technique that can be used in IoT Sensor Networks for clustering similar sensor nodes together. Two widely used methods of hierarchical clustering are single linkage and complete linkage. Comparing these two methods can help us understand their strengths and weaknesses and choose the best method for a particular application. In this methodology, we will discuss the steps involved in implementing and comparing single linkage and complete linkage hierarchical clustering methods in ISNs.

3.1. Data collection

In IoT sensor networks, data collection is essential in performing hierarchical clustering. ISNs consist of many sensor nodes distributed over a geographical area to monitor environmental conditions in smart cities, such as temperature, humidity, and light intensity. The data collected by the sensor nodes is used for various applications such as environmental monitoring, traffic control, and security surveillance. The data collected by the sensor nodes is usually in the form of sensor readings, which are transmitted to a base station for further processing.

In addition to sensor readings, other information such as node location, time stamp, and battery level can also be collected. To collect data from sensor nodes, a Time Division Multiple Access Protocol (TDMA) has been implemented. This protocol defines how sensor nodes communicate with each other and how data is transmitted and received between nodes. Also, Carrier Sense Multiple Access (CSMA) has been implemented. This protocol defines how sensor nodes communicate with the base station. Data collection is a critical step in performing hierarchical clustering in ISNs. It involves collecting sensor readings and other relevant information from sensor nodes.

3.2. Preprocessing

In IoT sensor networks, preprocessing is crucial in preparing the data for hierarchical clustering. Including location information in the preprocessing stage can enhance the clustering performance and provide insights into the spatial distribution of the sensor nodes.

When the sensor nodes are deployed in a specific area, location information becomes essential to the data. Including location information in the preprocessing stage can help to group sensor nodes that are geographically close to each other and identify the spatial patterns of the sensor readings. This information can be used to identify the regions where the sensor readings are similar or different, and it can provide insights into the physical processes that are taking place in the area.

One common preprocessing technique that includes location information is spatial clustering. Spatial clustering involves grouping sensor nodes based on their geographical proximity. This technique can be used to identify clusters of sensor nodes located in the same region and to group sensor nodes with similar spatial patterns in their readings.

Moreover, in ISNs, energy efficiency is a critical concern, as the sensor nodes have limited resources such as battery life and processing power. Therefore, preprocessing techniques that minimize the energy consumption of the sensor nodes should be used. The data aggregation technique has been used to reduce the amount of data transmitted, thereby reducing the energy consumption of the sensor nodes. Including location information in the preprocessing stage can enhance the clustering performance and provide insights into the spatial distribution of the sensor nodes.

3.3. Clustering

In this step, we will apply single linkage and complete linkage hierarchical clustering methods to cluster the sensor nodes based on their location information.

To apply both clustering methods, we first need to calculate the pairwise distances between each pair of sensor nodes using the Euclidean distance measure. The distance matrix can then be used as input to the clustering algorithm.

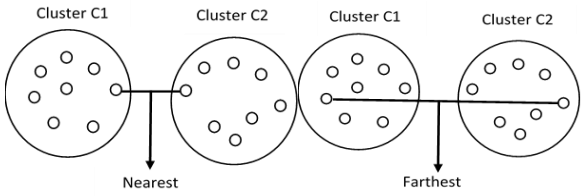
For single linkage clustering, Algorithm 1 starts by treating each sensor node as an individual cluster and then iteratively merges the two clusters with the closest pair of points, as shown in Fig. 1 (a). The next equation is responsible for handling this process. The process continues until all the sensor nodes belong to a single cluster,

$$D_s(C_1, C_2) = \min\{d(i \in C_1, j \in C_2)\}, \tag{1}$$

where C_1 represents the first cluster, and C_2 represents the second cluster.

For complete linkage clustering, Algorithm 2 starts by treating each sensor node as an individual cluster and then iteratively merges the two clusters with the farthest pair of points, as shown in Fig. 1 (b). The next equation is responsible for handling this process. The process continues until all the sensor nodes belong to a single cluster,

(2) $$D_c(C_1, C_2) = \max\{d(i \in C_1, j \in C_2)\}.$$



(a) (b)

Fig. 1. Single linkage (a), complete linkage (b)

Clustering the sensor nodes using single linkage and complete linkage hierarchical clustering methods can provide insights into the spatial patterns of the data. We can determine which method works better by comparing the results of both clustering methods.

3.4. Single linkage hierarchical clustering

The steps for performing single linkage hierarchical clustering on sensor nodes using Euclidean distance:

Step 1. Calculate the distance matrix between all pairs of sensor nodes based on Euclidean distance.

Step 2. Initialize each sensor node as a separate cluster.

Step 3. Find the two clusters with the shortest distance and merge them into a new cluster.

Step 4. Update the distance matrix to reflect the distances between the new cluster and the remaining clusters.

Step 5. Repeat Steps 3-4 until all sensor nodes are in a single cluster or the desired number of clusters is reached.

Step 6. Output the final clustering results.

Note that Step 3 uses the single linkage criterion, which merges clusters based on the shortest distance between any two points in the clusters, as calculated by the Euclidean distance. Also, in Step 5, the desired number of clusters can be set as a parameter.

Algorithm 1. Single Linkage Hierarchical Clustering

Input: Sensor data from nodes

Output: Clustered sensor nodes

Step 1. Calculate the distance matrix between all pairs of sensor nodes based on Euclidean distance

distance_matrix = calculate_distance_matrix(sensor_data)

Step 2. Initialize each sensor node as a separate cluster

clusters = initialize_clusters(sensor_data)

Step 3-5. Merge clusters based on the shortest distance until the desired number of clusters is reached

while len(clusters) > 1:

Find the two clusters with the shortest distance

C_1, C_2 = find_shortest_distance(distance_matrix)

Merge the two clusters into a new cluster

new_cluster = merge_clusters(clusters[C_1], clusters[C_2])

Update the distance matrix to reflect the distances between the new cluster and the remaining clusters

distance_matrix = update_distance_matrix(distance_matrix, clusters,C_1, C_2)

Remove the merged clusters from the list of clusters and add the new cluster

clusters.pop(C_1, C_2)

clusters.push(new-cluster)

Step 6. Output the final clustering results

output_clusters(clusters)

In Step 1, calculating the distance matrix between all pairs of sensor nodes based on Euclidean distance involves computing the pairwise distances between all sensor nodes based on their location. The Euclidean distance is a commonly used distance metric in machine learning and is defined as the straight-line distance between two points in Euclidean space, as shown in Fig. 2.

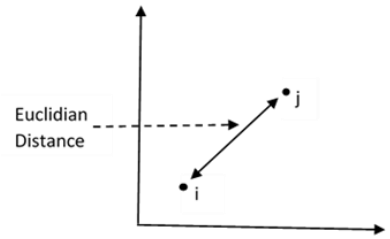


Fig. 2. Euclidian distance

To compute the distance matrix using Euclidean distance, the distance between *i* and *j* is denoted by

(3) $$d(i,j) = \sqrt{|x_1 - x_2|^2 + |y_1 - y_2|^2}\ .$$

In Step 2, each sensor node is initially treated as its own cluster. This means that we start with as many clusters as sensor nodes in the network. This step is called *initializing the clusters*. By initializing each sensor node as a separate cluster, we are essentially creating a set of singleton clusters, where each cluster contains only one sensor node. The rationale behind this approach is to start with the simplest possible clustering, where each sensor node is considered unique and has no relationship to any other sensor node.

Step 3, the next step after initializing the clusters, is to iteratively merge the two clusters that are closest to each other. This step involves finding the two clusters with the shortest distance and merging them into a new cluster. To find the two clusters with the shortest distance, we use the distance matrix that was calculated in the previous step. The distance matrix is a square matrix that contains the pairwise

distances between all the sensor nodes. Each row and column of the distance matrix corresponds to a single sensor node, and the value at the intersection of row and column represents the distance between these nodes. To find the two clusters with the shortest distance, we simply search the distance matrix for the smallest value. The indices of the smallest value correspond to the two clusters that are closest to each other. Once we have identified these two clusters, we merge them into a new cluster. The new cluster is created by combining the sensor nodes from the two old clusters.

Merging two clusters involves defining a criterion for how to combine the sensor nodes from the two clusters. In single linkage clustering, the criterion is to simply take all the sensor nodes from both clusters and form a new cluster. This new cluster will be represented by a new row and column in the distance matrix, and the distances between this new cluster and all the other clusters need to be calculated and added to the distance matrix.

Step 4, after combining two clusters into a new cluster, the distance matrix is updated to reflect the distances between the new cluster and the remaining clusters. When two clusters are integrated, the distance between them becomes irrelevant because they now belong to the same cluster. However, the distances between the new cluster and the other clusters must be updated to reflect that the new cluster now represents a group of sensor nodes instead of a single sensor node.

The distance between the new cluster and each remaining cluster must be calculated to update the distance matrix. This can be accomplished through the use of a linkage criterion that defines the distance between two clusters based on the distances of their individual sensor nodes. Single linkage clustering defines the distance between two clusters as the minimal distance between any two sensor nodes, one from each cluster.

To calculate the distance between the new cluster and each of the other clusters, we first determine the distance between each sensor node in the new cluster and each sensor node in the other clusters. The minimum of these distances is then used to determine the distance between the new cluster and the remaining cluster. These new distances are substituted for the distances previously associated with the merged clusters in the distance matrix. This process of updating the distance matrix is crucial because it guarantees that the distances between all clusters remain accurate throughout the clustering procedure. By updating the distance matrix appropriately, we can ensure that the two clusters with the shortest distance are always identified and merged at each iteration, resulting in a valid hierarchical clustering of the data.

Step 5 involves iterating over steps 3 and 4 until a stopping criterion is met. The stopping criterion can be either of the following:

- All sensor nodes are in a single cluster, indicating that the clustering process is complete.
- The desired number of clusters is reached, indicating that the algorithm should stop clustering and output the results.

By repeating Steps 3-4 until all sensor nodes are in a single cluster or the desired number of clusters is reached, the algorithm ensures that it has found the best possible hierarchical clustering based on the given distance metric and stopping criterion.

Step 6, the final output of the hierarchical clustering algorithm is the set of clusters formed as a result of merging individual sensor nodes or sub-clusters. The number of clusters formed will depend on the stopping criterion that was used, whether it was a pre-defined number of clusters or a hierarchical structure with a specific level of similarity.

3.5. Complete linkage hierarchical clustering

Complete linkage hierarchical clustering is another method for clustering sensor nodes, which is similar to single linkage clustering but uses a different distance metric to measure the similarity between clusters. The steps involved in complete linkage hierarchical clustering are as follows:

Step 1. Calculate the distance matrix between all pairs of sensor nodes based on Euclidean distance.

Step 2. Initialize each sensor node as a separate cluster.

Step 3. Find the two clusters with the maximum distance, and merge them into a new cluster.

Step 4. Update the distance matrix to reflect the distances between the new cluster and the remaining clusters.

Step 5. Repeat Steps 3 and 4 until all sensor nodes are in a single cluster or the desired number of clusters is reached.

Step 3 is different from single linkage clustering, as it calculates the maximum distance between any two sensor nodes from each cluster and merges the clusters with the maximum distance. This method tends to form more compact and spherical clusters than single linkage clustering.

As with single linkage clustering, Step 5 involves iterating over Steps 3 and 4 until a stopping criterion is met. The stopping criterion can be either of the following:

- All sensor nodes are in a single cluster, indicating that the clustering process is complete.
- The desired number of clusters is reached, indicating that the algorithm should stop clustering and output the results.

The clusters formed using complete linkage clustering may differ from those formed using single linkage clustering, depending on the distance metric used.

Single linkage and complete linkage hierarchical clustering are widely used methods for grouping nodes into clusters based on their proximity. These methods differ primarily in how they measure the distance between clusters. Single linkage clustering calculates the distance between clusters as the shortest distance between any two nodes in the respective clusters. This method tends to produce clusters connected by a chain of nodes. It may also merge close clusters.

In contrast, complete linkage clustering measures the distance between clusters as the maximum distance between any two nodes in the clusters. It results in more compact clusters. Complete linkage tends to produce well-separated clusters. However, it can be computationally more intensive due to the need to calculate maximum distances. The choice between single linkage and complete linkage hierarchical clustering depends on the application.

Algorithm 2. Complete Linkage Hierarchical Clustering

Input: Sensor data from nodes

Output: Clustered sensor nodes

Step 1. Calculate the distance matrix between all pairs of sensor nodes based on Euclidean distance

distance_matrix = calculate_distance_matrix(sensor_data)

Step 2. Initialize each sensor node as a separate cluster

clusters = initialize_clusters(sensor_data)

Step 3-5. Merge clusters based on the farthest distance until the desired number of clusters is reached

while len(clusters) > 1:

Find the two clusters with the farthest distance

C_1, C_2 = find_ farthest _distance(distance_matrix)

Merge the two clusters into a new cluster

new_cluster = merge_clusters(clusters[C_1], clusters[C_2])

Update the distance matrix to reflect the distances between the new cluster and the remaining clusters

distance_matrix = update_distance_matrix(distance_matrix, clusters, C_1, C_2)

Remove the merged clusters from the list of clusters and add the new cluster

clusters.pop(C_1, C_2)

clusters.push(new_cluster)

Step 6. Output the final clustering results

output_clusters(clusters)

3.6. Determining the optimal number of clusters

The optimal number of clusters must be determined after aggregating the sensor nodes using both single linkage and complete linkage hierarchical clustering algorithms. The optimal number of clusters can be determined using the elbow method, which is a method for determining the number of clusters based on the Within-Cluster Sum of Squares (WCSS) values. The WCSS value is the sum of the squared distances between each sensor node and its cluster centroid,

$$\mathrm{WCSS}(C_k) = \sum_{i \in C_k} (i - m_k)^2, \tag{4}$$

where i represents a sensor node belonging to the cluster C_k, and m_k is the mean value of the nodes assigned to the cluster C_k.

The elbow method involves selecting the number of clusters at which the change in WCSS value begins to level off. To apply the elbow method, we first determine the WCSS value for cluster sizes ranging from one to the total number of sensor nodes. The WCSS values can then be plotted against the number of clusters to locate the elbow point.

We can employ the elbow method independently to both single linkage and complete linkage clustering algorithms and then compare the results. As shown in Fig. 3, the optimal number of clusters can be determined by selecting the point where the curve begins to level off, and the improvement in the WCSS value is insignificant. Once the optimal number of clusters has been determined, the clustering algorithms can be re-executed using that number of clusters to produce the final clustering result.

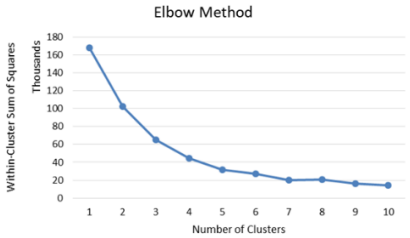


Fig. 3. Determining the optimal number of clusters

The steps for the elbow method in hierarchical clustering for sensor nodes are as Algorithm 3.

Algorithm 3. Elbow Method in Hierarchical Clustering for Sensor Nodes

Input: Sensor data from nodes

Output: Clustered sensor nodes

Step 1. Calculate the distance matrix

distance_matrix = calculate_distance_matrix(sensor_nodes)

Step 2. Initialize each sensor node as a separate cluster

clusters = initialize_clusters(sensor_nodes)

Step 3. Perform clustering until the maximum number of clusters is reached

max_clusters_reached = False

while not max_clusters_reached:

Step 3a. Merge the two clusters with the smallest distance

C_1, C_2= find_clusters_with_smallest_distance(clusters, distance_matrix)

new_cluster = merge_clusters(clusters[C_1], clusters[C_2])

Step 3b. Update the distance matrix

distance_matrix = update_distance_matrix(distance_matrix, clusters, C_1, C_2)

Step 3c. Calculate the sum of squared distances

sum_of_squared_distances = calculate_sum_of_squared_distances(sensor_nodes, new_cluster)

// Check if the maximum number of clusters has been reached

if len(clusters) == max_clusters:

max_clusters_reached = True

Step 4. Plot the sum of squared distances against the number of clusters

plot_sum_of_squared_distances(sum_of_squared_distances)

Step 5. Identify the elbow point

elbow_point = find_elbow_point(sum_of_squared_distances)

Step 6. Choose the number of clusters at the elbow as the optimal number of clusters

optimal_clusters = elbow_point

Step 7. Perform clustering again using the optimal number of clusters

final_clusters = perform_clustering(sensor_nodes, optimal_clusters)

Step 8. Output the final clustering results

output_clusters(final_clusters)

In Step 3, the algorithm keeps merging the two clusters with the smallest distance until the maximum number of clusters is reached. The distance between clusters is calculated using complete linkage and Euclidean distance.

In Step 3c, we calculate the sum of squared distances between each data point and its corresponding cluster center. This is done to plot the sum of squared distances against the number of clusters in Step 4.

In Step 4, we plot the sum of squared distances against the number of clusters. This plot is used to identify the elbow, which is the number of clusters where the rate of decrease in the sum of squared distances starts to level off.

In Step 5, we identify the number of clusters at the elbow as optimal. This is where adding more clusters does not lead to a significant reduction in the sum of squared distances.

Finally, in Step 7, we output the final clustering results based on the optimal number of clusters identified in Step 6.

Looking at Fig 3, the dataset provided for determining the optimal number of clusters via the elbow method has been generated using a simulation process, the specifics of which will be comprehensively described in the subsequent section of our research paper. To determine the optimal number of clusters, we have applied the elbow method. This technique involves plotting the WCSS values against the number of clusters and identifying where the reduction in WCSS begins to slow significantly.

Upon visual analysis of the data, we observed the following trend: As the number of clusters increases from 1 to 2, there is a significant reduction in WCSS. This reduction continues as we move from 2 to 3 clusters and from 3 to 4 clusters. However, the rate of decrease in WCSS starts to slow notably when transitioning from 4 to 5 clusters, and it becomes even less pronounced as we proceed to 6 clusters and beyond. Therefore, based on the elbow method, the optimal number of clusters for this dataset appears to be 4. This selection is made because it is at this point that the reduction in WCSS starts to level off significantly, indicating that adding more clusters does not yield a proportionate reduction in WCSS.

4. Discussion

In this section, we assess the effectiveness of both single linkage and complete linkage. The ISN and its sensor nodes are situated in a 100×100 m area. Therefore, we have conducted experiments to evaluate parameters such as cluster size, simulation time, network lifetime, and cluster quality. Table 1 outlines the parameters used during the simulation, and all experiments were carried out using the OMNET simulator [29].

Table 1. Simulation parameters

Parameter	Value
Size of sensing field	100×100 m
Number of sensor nodes	From 50 up to 100 nodes
The initial energy of each node	0.9 J
Base station location	50×175
E_{el}	50 nJ per 1 bit
εfs	10 pJ per 1 bit per $1m^2$
Size of a data packet	500 bytes
Size of info packet	25 bytes

4.1. Radio model

Our radio model is similar to the one presented in [30]. We have calculated the power consumption and transmission for *n*-bit data over a distance of *d* using the following equations. In these equations, E_{Tx} refers to the power used for Transmitting circuits, while E_{Rx} refers to the power used for Receiving circuits. E_{DA} is the energy required for gathering Data. Each sensor node has a primary energy of 0.9 J. Based on the next equation, we have determined the amount of energy needed for each sensor node to send *n*-bit data over a distance of *d*,

(5) $$E_{\mathrm{Tx}}(n,d) = nE_{\mathrm{elec}} + n\,\varepsilon_{\mathrm{fs}}\,d^2,$$

where the term E_{elec} refers to electronic energy, while $\varepsilon_{\mathrm{fs}}$ stands for free space power loss.

To receive *n*-bit data from a sensor node over a distance of *d*, a certain amount of energy is required,

(6) $$E_{\mathrm{Rx}}(n) = nE_{\mathrm{elec}}.$$

The energy needed for data aggregation is

(7) $$E_{\mathrm{DA}} = 5\ \mathrm{nJ\ per\ 1\ bit\ per\ 1\ signal}.$$

4.2. Cluster size

One simple method to evaluate the number of nodes in clusters is to examine the size of each cluster. A cluster that is too small may not be representative of the underlying structure of the network, while a cluster that is too large may be too general and obscure important patterns. By examining the size of each cluster, one can determine if the number of nodes in each cluster is appropriate.

Fig. 4 shows the cluster sizes produced by the single linkage and complete linkage hierarchical clustering methods for different numbers of nodes in the sensor network. In the case of single linkage, the clusters produced vary in size with the number of nodes, with some clusters having significantly more nodes than others. For example, in the case of 50 nodes, the largest cluster has 31 nodes, while the smallest cluster has only 2 nodes. This suggests that single linkage may not be as effective at producing evenly-sized clusters.

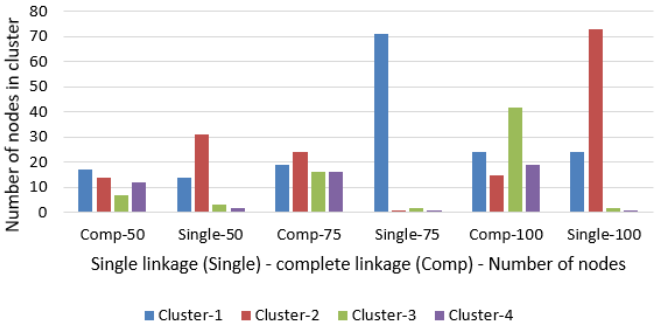


Fig. 4. Cluster size

On the other hand, the clusters produced by complete linkage tend to be more evenly sized, with the number of nodes in each cluster being more evenly distributed.

For example, in the case of 75 nodes, each cluster has between 16 and 24 nodes. This suggests that complete linkage may be a better method for producing clusters of similar sizes.

4.3. Simulation time

Simulation time is the duration required to execute the entire network operation. It is a metric for measuring the time complexity of a recently developed and simulated model. Additionally, it indicates how quickly the proposed algorithm converges.

We have evaluated the simulation time for single linkage and complete linkage hierarchical clustering methods for different sensor node configurations. Fig. 5 shows that the simulation time increases as the number of sensor nodes increases for both single linkage and complete linkage.

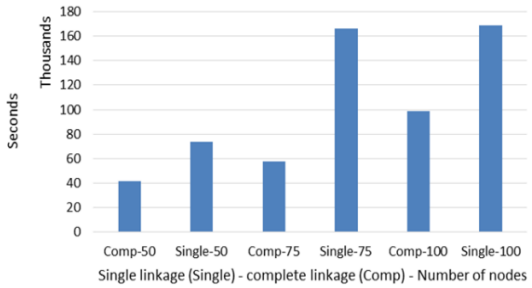


Fig. 5. Simulation time

For single linkage, the simulation time for 50 nodes was 73,665 ms; for 75 nodes, it was 166,305 ms; and for 100 nodes, it was 168,429 ms. On the other hand, for complete linkage, the simulation time for 50 nodes was 41,867 ms; for 75 nodes, it was 57,445 ms; and for 100 nodes, it was 98,467 ms.

These results indicate that complete linkage has a lower simulation time than single linkage for all sensor node configurations. This is because the single linkage considers the minimum distance between clusters. However, it is important to note that the difference in simulation time between the two methods is relatively small for smaller node configurations but becomes more significant as the number of nodes increases. Overall, the simulation time results provide valuable insights into the time complexity of the clustering algorithms and can help optimize the network performance.

4.4. Network lifetime

The network lifetime of a sensor network refers to the time duration until the first node dies or the time duration until most nodes die. This is an important metric to evaluate the performance of clustering algorithms since it determines how long the network can operate efficiently.

Fig. 6 shows the network lifetime of sensor nodes using single linkage and complete linkage hierarchical clustering methods. For the single linkage method, the network lifetime decreases as the number of nodes increases. In the case of 50 nodes,

the network lifetime was 234 rounds before all the nodes died, while for 75 nodes and 100 nodes, the network lifetime decreased to 233 and 228 rounds, respectively. On the other hand, the complete linkage method showed a better network lifetime than the single linkage method. For instance, with 50 nodes, the network lifetime was 236 rounds, which was better than the 234 rounds of single linkage. Similarly, for 75 and 100 nodes, the network lifetime was 234 and 236 rounds, respectively, which were better than the corresponding values of single linkage. Overall, the complete linkage method showed a better network lifetime than the single linkage method.

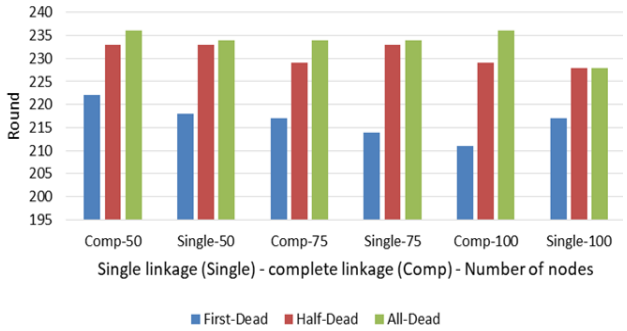


Fig. 6. Network lifetime

4.5. Cluster quality

The Silhouette coefficient is used to evaluate the quality of the clusters produced by hierarchical clustering methods for sensor nodes. The Silhouette coefficient measures how well each sensor node fits into its assigned cluster.

To calculate the Silhouette coefficient for each sensor node, the distance between the node and all other nodes in its assigned cluster is calculated, as well as the distance between the node and all other nodes in the next nearest cluster. The Silhouette coefficient is then calculated as the difference between these two distances divided by the maximum of the two distances.

For each node in the network, compute its distance to all other nodes in the same cluster. This results in a set of average distances, which are denoted as a for each node. Cohesion measures how closely related nodes are in a cluster. For each node, compute its distance to all nodes in the nearest neighboring cluster. This results in another set of average distances, which are denoted as b for each node. Separation measures how distinct or well-separated a cluster is from other clusters.

Calculate the silhouette coefficient for each node using the formula:

$$S_i = \frac{b_i - a_i}{\max(a_i, b_i)}, \tag{8}$$

where S_i is the silhouette coefficient for node i, a_i is the average distance between node i and all other nodes in the same cluster, and b_i is the average distance between node i and all nodes in the nearest neighboring cluster.

Compute the average silhouette coefficient across all nodes in the network to obtain a measure of the overall quality of the clustering result using the formula:

$$\bar{S} = \frac{1}{n} \sum_{i=1}^{n} S_i, \tag{9}$$

where n is the total number of nodes in the network.

A high Silhouette coefficient indicates that the node is well-matched to its assigned cluster, while a low Silhouette coefficient indicates that the node may be better matched to another cluster.

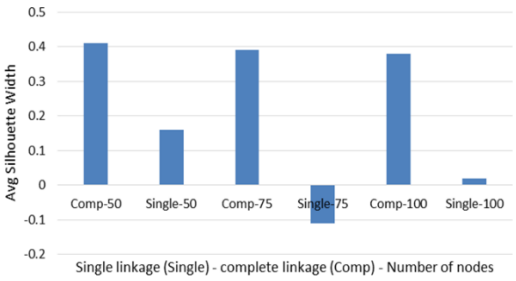


Fig. 7. Cluster quality

A value closer to 1 indicates that the node is well-matched to its cluster, while a value closer to –1 indicates that the node may belong to a different cluster. In this study, we used the Silhouette coefficient to evaluate the quality of the clusters produced by single linkage and complete linkage hierarchical clustering methods for sensor nodes. Fig. 7 shows that the average silhouette widths for the different methods and the number of nodes vary significantly.

For the single linkage method, the average silhouette width is positive for 50 and 100 nodes but negative for 75 nodes. This suggests that the clustering quality is relatively good for 50 and 100 nodes but not as good for 75 nodes. On the other hand, the complete linkage method shows consistently good clustering quality with positive average silhouette widths for all three number of nodes. The values for complete linkage are also consistently higher than those for single linkage, suggesting that it may be a more effective method for clustering sensor nodes.

Overall, the results indicate that the complete linkage hierarchical clustering method produces better quality clusters for sensor nodes than the single linkage method, as indicated by the consistently higher Silhouette coefficients.

5. Conclusion

In conclusion, this study compares the performance of single linkage and complete linkage hierarchical clustering methods in ISN applications. The results show that both methods have their own strengths and limitations.

Single linkage clustering has been found to be more suitable for ISN applications where the network topology is highly connected and densely populated, as it tends to produce smaller and more compact clusters. On the other hand, complete linkage clustering has been found to be more suitable for applications with sparser network topology, as it tends to produce larger and more distinct clusters.

The study also shows that the performance of the clustering methods can be affected by the number of clusters, the size of the network, and the specific clustering algorithm used. Therefore, it is important to select the clustering method and algorithm based on the specific requirements and characteristics of the application.

Overall, the results of this study provide valuable insights into the strengths and limitations of different clustering methods for ISN applications. The future of ISNs presents opportunities for exploration. Researchers could develop energy-efficient routing protocols with machine learning. Improving the scalability and heterogeneity management of ISNs is vital for their evolution.

Acknowledgments: This work was funded by the Deanship of Scientific Research (DSR), King Abdulaziz University, Jeddah, Saudi Arabia, under Grant No D-096-611-1439. The authors, therefore, gratefully acknowledge the DSR technical and financial support.

References

1. Bilal, J., H. Farman, H. Javed, B. Montrucchio, M. Khan, S. Ali. Energy Efficient Hierarchical Clustering Approaches in Wireless Sensor Networks: A Survey. – Wireless Communications and Mobile Computing, 2017, pp. 1-14. **https://doi.org/10.1155/2017/6457942**
2. Gopika, D., R. Panjanathan. Energy Efficient Routing Protocols for WSN Based IoT Applications: A Review. – Materials Today Proceedings, November 2020. **https://doi.org/10.1016/j.matpr.2020.10.137**
3. De-gan, Z., J. Qiu, T. Zhang, H. Wu. New Energy-Efficient Hierarchical Clustering Approach Based on Neighbor Rotation for Edge Computing of IOT. – In: Proc. of 28th International Conference on Computer Communication and Networks (ICCCN'19), IEEE, 2019, pp. 1-2.
4. Gazi, R. M. E., K. Wahid. LDCA: Lightweight Dynamic Clustering Algorithm for IoT-Connected Wide-Area WSN and Mobile Data Sink Using LoRa. – IEEE Internet of Things Journal, Vol. **9**, 2021, No 2, pp. 1313-1325.
5. Mohammad, M., Y. Jaradat, D. Zaidan, I. Jannoud. To Cluster or Not to Cluster: A Hybrid Clustering Protocol for WSN. – In: Proc. of IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT'19), IEEE, 2019, pp. 678-682.
6. Shemim, F., U. Witkowski. Energy Efficient Clustering Protocols in WSNs: Performance Analysis and Comparison of EEAHP Protocol with LEACH and EAMMH Using MATLAB. – In: Proc. of Advances in Science and Engineering Technology International Conferences (ASET'20), IEEE, 2020, pp. 1-5.
7. Dhiviya, S., A. Sariga, P. Sujatha. Survey on WSN Using Clustering. – In: Proc. of 2nd International Conference on Recent Trends and Challenges in Computational Models (ICRTCCM'17), IEEE, 2017, pp. 121-125.
8. Santosh, A., K. C. Manoj. A Survey on Clustering Approaches to Strengthen the Performance of Wireless Sensor Network. – In: Proc. of 2nd International Conference on Inventive Research in Computing Applications (ICIRCA'20), IEEE, 2020, pp. 814-820.
9. Fionn, M., P. Contreras. Algorithms for Hierarchical Clustering: An Overview. – Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, Vol. **2**, 2012, No 1, pp. 86-97.
10. Odilia, Y., K. T. Ramdeen. Hierarchical Cluster Analysis: Comparison of Three Linkage Measures and Application to Psychological Data. – The Quantitative Methods for Psychology, Vol. **11**, 2015, No 1, pp. 8-21.
11. Sambo, W., B. O. Yenke, A. Förster, P. Dayang. Optimized Clustering Algorithms for Large Wireless Sensor Networks: A Review. – Sensors, Vol. **19**, 2019, No 2, 322.
12. Amin, S., A. Taherkordi, Y. Haugen, F. Eliassen. Clustering Objectives in Wireless Sensor Networks: A Survey and Research Direction Analysis. – Computer Networks, Vol. **180**, 2020, 107376.
13. Himanshu, S., A. Haque, F. Blaabjerg. Machine Learning in Wireless Sensor Networks for Smart Cities: A Survey. – Electronics, Vol. **10**, 2021, No 9, 1012.

14. W e n l i a n g, W., N. X i o n g, C. W u. Improved Clustering Algorithm Based on Energy Consumption in Wireless Sensor Networks. – Iet Networks, Vol. **6**, 2017, No 3, pp. 47-53.
15. S y e d B i l a l, S., Z. C h e n, F. Y i n, I. U l l a h K h a n, N. A h m a d. Energy and Interoperable Aware Routing for Throughput Optimization in Clustered IoT-Wireless Sensor Networks. – Future Generation Computer Systems, Vol. **81**, 2018, pp. 372-381.
16. S o b i n, C. C. A Survey on Architecture, Protocols and Challenges in IoT. – Wireless Personal Communications, Vol. **112**, 2020, No 3, pp. 1383-1429.
17. T r u p t i M a y e e, B., U. C h a n d r a S a m a l, S. K. M o h a p a t r a. Energy-Efficient Modified LEACH Protocol for IoT Application. – IET Wireless Sensor Systems, Vol. **8**, 2018, No 5, pp. 223-228.
18. M e h d i, H., A. H e m m a t i, A. M. R a h m a n i. Clustering for Smart Cities in the Internet of Things: A Review. – Cluster Computing, Vol. **25**, 2022, No 6, pp. 4097-4127. **https://doi.org/10.1007/s10586-022-03646-8**
19. A b b a s S h a h, S., D. S i e r r a-S o s a, A. K u m a r, A. E l m a g h r a b y. IoT in Smart Cities: A Survey of Technologies, Practices and Challenges. – Smart Cities, Vol. **4**, 2021, No 2, pp. 429-75. **https://doi.org/10.3390/smartcities4020024**
20. M e h r a, P. S. Lbecr: Load Balanced, Efficient Clustering and Routing Protocol for Sustainable Internet of Things in Smart Cities. – Journal of Ambient Intelligence and Humanized Computing, 2022, pp. 1-23.
21. H a s s a n, E., A. N a j i d. ECH: An Enhanced Clustering Hierarchy Approach to Maximize Lifetime of Wireless Sensor Networks. – IEEE Access, Vol. **7**, 2019, pp. 107142-107153.
22. P r e m k u m a r, C., F. A l-T u r j m a n, M. K u m a r, T. S t e p h a n. I-AREOR: An Energy-Balanced Clustering Protocol for Implementing Green IoT in Smart Cities. – Sustainable Cities and Society, Vol. **61**, 2020, 102254.
23. V i m a l, V., K. U. S i n g h, A. K u m a r, S. K. G u p t a, M. R a s h i d, R. K. S a k e t, S. P a d m a n a b a n. Clustering Isolated Nodes to Enhance Network's Life Time of WSNs for IoT Applications. – IEEE Systems Journal, Vol. **15**, 2021, No 4, pp. 5654-5663.
24. A k h i l e s h, P. A., R. K. S i n g h. EEHCHR: Energy Efficient Hybrid Clustering and Hierarchical Routing for Wireless Sensor Networks. – Ad Hoc Networks, Vol. **123**, 2021, 102692.
25. A m r i t, M., A. P. G o s w a m i, L. Y a n g, Z. Y a n, M. D a n e s h m a n d. Dynamic Clustering Method Based on Power Demand and Information Volume for Intelligent and Green IoT. – Computer Communications, Vol. **152**, 2020, pp. 119-125.
26. A n u r a g, S., S. T r i p a t h i. A Multi-Tier Based Clustering Framework for Scalable and Energy Efficient WSN-Assisted IoT Network. – Wireless Networks, Vol. **26**, 2020, pp. 3471-3493.
27. A n k u r, C., S. K u m a r, S. G u p t a, M. G o n g, A. M a h a n t i. FEHCA: A Fault-Tolerant Energy-Efficient Hierarchical Clustering Algorithm for Wireless Sensor Networks. – Energies, Vol. **14**, 2021, No 13, 3935.
28. Z h a n g, D., L. C h e n, J. Z h a n g, J. C h e n, T. Z h a n g, Y. T a n g, J. Q i u. A Multi-Path Routing Protocol Based on Link Lifetime and Energy Consumption Prediction for Mobile Edge Computing. – IEEE Access, Vol. **8**, 2020, pp.69058-69071.
29. OMNET++ Simulation Environment. **http://www.omnetpp.org**
30. H e i n z e l m a n, W. B., A. P. C h a n d r a k a s a n, H. B a l a k r i s h n a n. An Application-Specific Protocol Architecture for Wireless Microsensor Networks. – IEEE Transactions on Wireless Communications, Vol. **1**, 2002, No 4, pp. 660-670. **https://doi.org/10.1109/twc.2002.804190**

Received: 07.09.2023; Second Version: 04.10.2023; Accepted: 16.10.2023 (fast track)