HAC: Hierarchical Agglomerative Clustering With Linear Programming for Wireless Sensor Networks

Mohi Dine, Sidi Mohamed
Finnerty, Patrick
Ohta, Chikara

(Citation)
IEEE Access,12:8110-8122

(Issue Date)
2024-01-12

(Resource Type)
journal article

(Version)
Version of Record

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 17 December 2023, accepted 30 December 2023, date of publication 12 January 2024,
date of current version 19 January 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3353318

RESEARCH ARTICLE

HAC: Hierarchical Agglomerative Clustering With Linear Programming for Wireless Sensor Networks

SIDI MOHAMED MOHI DINE, PATRICK FINNERTY, AND CHIKARA OHTA, (Member, IEEE)
Graduate School of System Informatics, Kobe University, Kobe 657-8501, Japan

Corresponding author: Sidi Mohamed Mohi Dine (sidimeddin@fine.cs.kobe-u.ac.jp)

This work was supported by JSPS KAKENHI under Grant JP22H03585.

ABSTRACT Wireless sensor networks are a collection of tiny sensor devices powered by energy-scarce batteries. Depending on the nature of the application, those tiny devices might be deployed in areas that make it difficult to recharge or change their battery. Therefore, it is important to optimize the energy consumption of these devices to prolong their lifetime for as long as possible. Clustering is regarded as the most efficient approach to extend the lifetime of wireless sensor networks. This study proposes a novel method integrating cluster head selection and cluster formation to obtain as long a network lifetime as is possible. We employ a linear programming model to resolve the issue of cluster head selection and a Hierarchical Agglomerative Clustering (HAC) method to form clusters in the network. We evaluated HAC on different sets of inputs where we varied the position of the sink from the corner of the network to the center. HAC outperformed other existing clustering techniques for the first node death (FND) and half node death (HND) lifetime definitions in the cases where the sink is at the corner of the network. In the cases where the sink is located at the center of the network, HAC outperformed the other existing techniques for all definitions of network lifetime.

INDEX TERMS Clustering, linear programming, network lifetime, power optimization, wireless sensor network.

I. INTRODUCTION

A wireless sensor network (WSN) consists of many sensors deployed to collect data from the sensed area [1], [2]. The miniature nature of these devices makes them useful in applications where only tiny sensors can fit [3]. However, these tiny devices operate with small batteries, raising the problem of energy scarcity as they are prone to discharge quickly and are cumbersome to recharge. The entire network is affected when these batteries run out of power as the sensors can no longer communicate.

Clustering has been proposed as an efficient way to optimize the energy performance of wireless sensor networks [4]. In clustering techniques, the network members are divided into groups. A special node in these clusters, the cluster head (CHs), transmits data to the sink [5]. To transmit data from a cluster to the sink, each member node sends its data to the cluster head. Then, the cluster head concatenates and sends all the received data to the sink. Since only cluster heads are allowed to transmit to the sink, the volume of overhead and control messages with the sink is reduced [6]. Thus, data aggregation in clustering techniques improves network efficiency by managing the resources of the network. Moreover, nodes that cannot reach the sink directly can relay their data through their cluster head [7].

The clustering procedure usually comprises two main phases: cluster head selection and cluster formation. The energy consumption in wireless sensor networks is very sensitive to how cluster heads are selected [8]. As the cluster head transmits the data of all members to the sink, it consumes more energy than the member nodes consume. Thus, selecting the same cluster head for many successive

The associate editor coordinating the review of this manuscript and approving it for publication was Yiming Tang.

rounds of transmission causes its battery to discharge quickly. To resolve this issue, cluster head rotation: a technique that many recent clustering and routing protocols use to share and balance the load between different members within a cluster [9], was introduced [10]. Besides the cluster head selection phase, the cluster's formation phase determines which nodes join which cluster. However, ensuring that each node is assigned to the most appropriate cluster is not trivial. Often, the number of clusters to form is decided a priori. Then, deciding which node belongs to which cluster involves testing all possible cluster combinations, which is not realistic due to the combinatorial explosion. Indeed, the number of ways we can partition a set of n nodes into k nonempty groups, i.e., k subsets is calculated as a Stirling set number of the second kind, which noted $S(n, k)$ [11]. There are several ways to calculate the number, one of them is to utilize the following recursive relationship:

$$S(n, n) = S(n, 1) = 1, \tag{1}$$

$$S(n, k) = S(n-1, k-1) + k\, S(n-1, k). \tag{2}$$

Given a network of 100 nodes to be clustered into five clusters, there are approximately 6.57×10^{67} possible ways to partition the nodes. To avoid the complexity of this problem, we propose a heuristic approach to reduce the search space.

In general, the nature of the network application and the accuracy and sensitivity of sensed data are the main factors that define the network lifetime. For instance, in applications where the readings obtained from each node are vital to the system, it is necessary to follow a conservative approach and define the network lifetime as the death of the first node. On the contrary, in some networks, it is possible to predict the readings of some nodes based on the nature of the system and from the history of the past transmission between those nodes and the sink [12]. In such a situation, the network can remain functional despite some of its member nodes not transmitting any more.

The concept of network lifetime is given multiple definitions in the literature. The death of a node or a group of nodes is always at the heart of the definitions of network lifetime. A node is dead when its battery capacity is insufficient to send useful information either directly to the sink or through a relay node [13]. In certain applications, network lifetime is determined by the death of the first node in the network [14]. In others, it is defined as the death of the last node in the network [15], or the death of a certain percentage of the nodes in the network [15].

In this study, we define the lifetime of the network by the death of the first node. Then, we propose a clustering technique that makes use of the network topology to delay the occurrence of the first node death, thus extending the network lifetime. This clustering mechanism is based on a hierarchical agglomerative clustering technique. This study's cluster head selection techniques are derived from the linear programming-based cluster head selection (LPCHS) method that we previously introduced in [16].

The rest of this paper is divided as follows: in Section II, we review some techniques used in the clustering of WSN. Section III introduces the method we propose, and Section IV presents the parameters and assumptions made to simulate the proposed method. Section V presents and compares the achieved results through HAC with existing clustering techniques. We discuss the results obtained in this study in Section VI, and we conclude in Section VII.

II. RELATED WORK

Routing protocols and clustering are essential topics in energy optimization in WSNs. In this section, we present various techniques that have been proposed.

A. LEACH

Low-energy adaptive clustering hierarchy (LEACH) is a self-organizing protocol in which nodes organize themselves into clusters [17]. Each cluster has a unique special cluster head node that allows communicating directly to the sink. Cluster head nodes spend a lot of energy collecting data from the member nodes in their cluster, compressing it, and transmitting it to the sink. The randomness, the fact that it does not consider the residual energy, and the distance between nodes and the sink in LEACH were among the points addressed by Mao et al. in [18]. In this paper, they proposed a method that gives weight to the residual energy and distance between nodes in the selection of cluster heads. This method also selects the data route between cluster heads based on the residual energy of the cluster heads involved in multi-hop data transmission to the sink, the distance between these cluster heads, and the angle formed by the current cluster head, the next hop cluster head, and the base station. In [19], Bhih et al. proposed an improved LEACH-based algorithm, LEACH-Kmeans, that utilizes the K-means clustering technique to select the cluster heads. Similarly to traditional LEACH, LEACH-Kmeans starts with setting the number of cluster heads and then uses the K-means technique to find the cluster center (closer to all nodes in the cluster) and assign it as a cluster head.

B. FUZZY LOGIC BASED CLUSTERING

A fuzzy logic approach has been proposed to select the cluster heads and determine the radius of the clusters to be formed around the selected cluster heads. In [20], a set of provisional cluster heads is chosen using a probabilistic method similar to the one used in LEACH. Subsequently, the residual energy, distance to the base station, and the concentration of each of these provisional cluster heads are input into a fuzzy logic controller. Then, the fuzzy logic controller selects the chance of each provisional cluster head to be selected as a prime (true) cluster head. The remaining provisional cluster heads which were not selected as prime cluster heads return to being normal member nodes. The fuzzy logic controller also outputs the radius of each cluster formed around each provisional cluster head. Cluster heads relay their data to the sink using a multi-hop transmission method. In this method,

clusters that are near the sink have to relay the data of all the remaining clusters, which exhausts their energy capacity faster. This problem is referred to as the hot spot problem. To alleviate this issue, this technique limits the size of these clusters.

Quanbiao et al. [21] proposed an uneven clustering approach based on fuzzy logic and the entropy weight method. The fuzzy logic controller receives the distance from a node to the base station, the node's remaining energy, and the node's neighborhood value as inputs. The controller outputs the probability of each node to be selected as a cluster head and the radius of each cluster. For the same reasons in [20], this approach reduces the size of clusters that are near the sink.

Nidhi et al. in [22] proposed a fuzzy logic-based clustering technique for cluster head selection in wireless sensor networks. In this technique, the fuzzy controller receives three membership functions: node centrality, distance from a node to the base station, and node energy. The fuzzy controller outputs a crisp value of the chance of a node to be selected as cluster head.

C. CLUSTERING BASED ON THE SIZE OF CLUSTERS

Singh et al. [23] proposed a method to improve the network lifetime through the formation of uniform cluster sizes. Initially, this method selects cluster heads and forms clusters in the same manner as LEACH does. Member nodes at the boundaries of large clusters have a large communication distance to their cluster heads which affects the energy consumed in inter-cluster communications. To curb this predicament, each of these member nodes is assigned to another cluster whose cluster head is closer than its initial cluster head.

To solve the hot spot problem, as in [20], Sivakumar et al. in [24], proposed an uneven clustering protocol. This approach involves the use of an oppositional group search optimizer (OGSO) executed by the base station. The cluster heads are selected based on their maximum residual energy, close distance to the base station, and the minimum number of neighboring nodes. This approach selects the node with the maximum residual energy, the closest to the base station, and with the minimum number of neighbors to be a cluster head.

D. BIO-INSPIRED CLUSTERING TECHNIQUES

In [25], Jan and Masood proposed a cluster head selection technique titled multiple solutions based particle swarm optimization for cluster-head-selection in wireless-sensor networks (MPSO). This method selects the best three cluster head candidates in every transmission round, where these candidates will serve as the cluster heads for the next three rounds. The selection of three cluster heads for the upcoming three rounds serves to reduce the number of calls for the cluster head selection algorithm, which then, reduces the amount of energy consumed. The cluster heads are selected based on their fitness function, which depends on three factors: Maximization of the residual energy, minimization of the distance between the cluster head and the base station, and minimization of the intra-cluster distance.

In [26], Jin and Muqing proposed a Genetic-based cluster routing algorithm for wireless sensor networks. One of the most common shortcomings of LEACH is the possibility of selecting a node with low energy or a node that is far from the base station as a cluster head. To tackle this issue, this approach takes the remaining energy and the distance to the base station into consideration when selecting the cluster head. A Genetic algorithm is applied to select the best path to relay the data from each cluster head to the base station. The fitness function of this Genetic algorithm takes into consideration the balance between energy consumed to transmit to the base station and the load of nodes.

E. LINEAR PROGRAMMING-BASED CLUSTERING TECHNIQUES

Related to the problem of energy optimization in WSN using linear programming, most of the existing studies invoke only cluster head selection, not cluster formation. The process of selecting the optimal cluster head is modeled as an optimization problem that aims to maximize the network lifetime and guarantees that all nodes are covered. Eskandari et al. used two models, an integer linear programming model and a linear programming model [27]. The integer linear programming modeled the optimization problem by placing emphasis only on the coverage of all nodes in the clusters. The linear programming model takes into consideration the optimization of energy consumption by ensuring that the distance between the cluster head and member nodes is as minimal as it can be. However, this method requires the number of cluster heads in the network to be decided beforehand.

Jyothi and Sakthivel presented CFCLP, a novel clustering framework based on the combinatorial approach and linear programming in wireless sensor networks [28]. CFCLP applies a linear programming model to solve the problem of cluster head selection in wireless sensor networks. To estimate the residual energy of nodes, CFCLP takes into consideration all possible causes of energy consumption such as traffic flow, time rate of transmission, and distance of transmission. The model used in CFCLP uses an optimization function that combines the amount of energy consumed due to each factor to select the most appropriate cluster head that guarantees minimum energy consumption.

Li et al. proposed an integer linear programming formulation for energy optimization in wireless sensor networks in [29]. In this study, they proposed three different ILP formulations that optimize the problem by selecting the minimum number of cluster heads that guarantee network coverage. After they formulate the problem such that the maximum lifetime is obtained, they then select the optimal CHs.

F. CONTRIBUTIONS OF THIS WORK

In most of the techniques mentioned above, the clusters are formed based on the concept of the radius of the cluster head as in [20] or the received signal strength (RSS) as in the LEACH technique. Both concepts are similar in how clusters are formed around the cluster heads. In both approaches, the network area is geographically divided into a group of sectors based on the cluster head radius or the RSS. Then, all the member nodes in the vicinity of the radius or RSS are assumed to belong to the same cluster. In other words, all members of a cluster are enclosed in a geographically contiguous area, and no members of other clusters exist in this area.

In HAC, we propose a more flexible cluster formation method, where clusters are not formed based on the location of the cluster heads. Members of a cluster could be scattered all around the network area. For instance, some cluster members could be at the network edge, some at the center, and others near the sink. HAC selects cluster members based on minimizing the amount of energy these members spend to transmit to the sink as well as the amount of energy spent to transmit inside the cluster (to the CHs). As far as we know, HAC is the only clustering technique that scatters the clusters all over the network area.

The contributions of this paper are as follows:

- We proposed a novel linear programming-based cluster head selection technique.
- We proposed a cluster formation heuristic tailored for the linear programming cluster head selection technique.
- The formation of flexible clusters with scattered members in the network area.
- Simulations reveal that HAC performs better than existing clustering protocols.

III. PROPOSED METHOD

In general, the nodes farthest from the sink discharge faster than the others. In Section I, we defined the network lifetime as the death of the first node in the network. Therefore, the nodes that discharge the fastest limit the lifetime of the whole network. Hence, our technique aims to prolong the lifetime of these nodes. In this context, we will refer to the nodes prone to discharge quickly as "*weak*" nodes and those less susceptible to discharge quickly as "*strong*" nodes.

The principle of our proposed clustering technique consists of merging weak nodes with strong nodes to create clusters with longer lifetimes. Subsequently, to ensure a balanced load distribution between strong and weak nodes within these newly formed clusters, we employ the linear programming cluster head selection method we introduced in [16] to equitably distribute the cluster head responsibilities among the cluster members.

However, deciding which weak node should be merged with which strong node to achieve optimal performance is not trivial. In our proposed method, we start by considering every single node as a cluster. Then, we repetitively merge the weakest cluster with the cluster that will best improve its lifetime.

TABLE 1. Parameters used in clustering.

Symbol	Definition
$\mathcal{N}$	the set of nodes in the network
$\mathcal{C}$	set of nodes in a cluster
R_i	the number of rounds the node i is selected as cluster head
$L_{\mathcal{C}}$	lifetime of cluster $\mathcal{C}$
$E_{i\to\text{sink}}$	the amount of energy the node i consumes to transmit to the sink if it is selected as the cluster head
$E_{i\leftarrow j}$	denotes the amount of energy that the node i consumes as the cluster head to receive data from the member node j
$E_{i\to j}$	denotes the amount of energy consumed by the member node i to transmit to the cluster node j
E_i^{init}	denotes the initial energy of the node i.
$\mathcal{F}_{\text{net}}$	the family of clusters in the network
$\mathcal{F}_{\text{net}}^{\text{opt}}$	the family of clusters which provisionally delivered the longest lifetime
L_{net}	network lifetime
$L_{\text{net}}^{\text{opt}}$	provisionally optimal lifetime the network

In Subsection III-A, we recall our cluster head selection method. Our hierarchical agglomerative cluster formation method is introduced in Subsection III-B. We then demonstrate our method with an example in Subsection III-C. We then briefly analyze the complexity of our method in Subsection III-D.

A. CLUSTER HEAD SELECTION

We adopt the LPCHS method we proposed in previous work [16]. This method uses a linear programming model to determine the number of rounds each node in the cluster should serve as the cluster head.

In each round, every member of the cluster is either a cluster head or a "simple" member node. The simple nodes transmit their data to the cluster head. The cluster head then transmits all the data it receives as well as its data to the sink. Therefore, the cluster head tends to consume more energy than the other member nodes in a given round as it performs two main tasks: gathering the data from all the member nodes and transmitting the gathered data to the sink.

We modeled the energy consumption of each node based on the number of rounds this node serves as cluster head. We formulated a linear programming problem to maximize the cluster's lifetime, i.e., the sum of the number of rounds each node serves as a cluster head, as shown in Equation (3). This optimization problem is constrained by the energy consumption of the nodes, which cannot exceed their battery capacity, as formulated in Equation (4). This constraint is composed of two terms: the energy consumed when the node is the cluster head (top half of Equation (4)) and the energy consumed when sending to the cluster head.

$$\text{Maximize} \quad \sum_{i\in\mathcal{C}} R_i, \tag{3}$$

subject to

$$E_i^{\text{init}} \geq \left(E_{i\to\text{sink}} + \sum_{j\in\mathcal{C}\setminus\{i\}} E_{i\leftarrow j} \right) R_i + \sum_{j\in\mathcal{C}\setminus\{i\}} E_{i\to j} R_j, \quad \text{for } i \in \mathcal{C}, \tag{4}$$

The list of symbols used in this section is listed in Table 1.

To compute the terms $E_{i\to\text{sink}}$, $E_{i\leftarrow j}$, and $E_{i\to j}$, we need to specify the energy consumption model. In this study, we use the first-order radio model, as we will explain later in Subsection IV-B.

The linear programming model above outputs the cluster head proportion between nodes. It also computes the lifetime of the cluster, which will play an important role in the cluster formation method we detail in the next subsection.

B. CLUSTER FORMATION

Results obtained from [16] confirmed that LPCHS extends the lifetime of wireless sensor networks. However, LPCHS does not propose a complete clustering technique as it lacks a method to divide the network into clusters. Instead, LPCHS borrows the clustering method from LEACH, which leaves some lifetime gains untapped. To tackle this issue, HAC proposes a systematic method to form clusters based on the results obtained from LPCHS.

As stated in Subsection III-A, the nodes that discharge faster limit the lifetime of the network, as we defined the network lifetime by the first node death. HAC uses a bottom-up approach to merge the nodes that discharge faster with other nodes to form new clusters with longer lifetimes.

At the beginning, each node is treated as a cluster of a single member. Cluster formation starts by selecting the ''weakest cluster'' i.e., the one with the shortest lifetime. We then calculate the lifetime of the clusters that would be formed if we merged this weakest cluster with another and choose the combination that produces the merged cluster with the longest lifetime. The cluster lifetime is determined using the model explained in Subsection III-A. We repeat this process, selecting the next weakest cluster until the entire network is consolidated into a single cluster. Finally, we identify the step at which the formed clusters achieved the longest lifetime, representing the finalized clustering configuration of the network.

Algorithm 1 illustrates the procedure for performing the clustering phase. $\mathcal{N}$ denotes the set of nodes in the network. The variable $\mathcal{C}$ represents a cluster of nodes, and $L_{\mathcal{C}}$ denotes the cluster's lifetime, calculated using Equation (3). We use the notation $\mathcal{F}_{\text{net}}$ to refer to the family of clusters (a cluster being a set of nodes) as the network configuration. Let L_{net} denotes the network lifetime, which is defined as the lifetime of the weakest cluster in terms of FDN. Therefore, L_{net} is given by

$$L_{\text{net}} = \min_{\mathcal{C}\in\mathcal{F}_c} L_{\mathcal{C}}. \tag{5}$$

At this initial stage, every cluster is regarded as a cluster with a single node, that is, $\mathcal{F}_{\text{net}} = \{\{i\} | i \in \mathcal{N}\}$. In such a case, the cluster lifetime depends only on transmitting energy to the sink. Therefore, the lifetime of the cluster with node i is determined as the number R_i of transmissions from node i to the sink. Consequently, Equation (4) is simplified as

$$E_i^{\text{init}} \geq E_{i\to\text{sink}} R_i. \tag{6}$$

The largest R_i that satisfies Equation (6) is the lifetime of that cluster like Equation (3). Although R_i is conceptually an integer, in Algorithm 1, it is treated as a floating-point number and calculated using

$$R_i = \frac{E_i^{\text{init}}}{E_{i\to\text{sink}}}. \tag{7}$$

The weakest cluster is identified as the single-node cluster with node $i^* = \text{argmin}_{i\in\mathcal{N}} R_i$. Consequently, the network lifetime is calculated as follows:

$$L_{\text{net}} = R_{i^*}. \tag{8}$$

Note that Equations (4) and (3) naturally result in Equations (7) and (8), respectively, given $\mathcal{F}_{\text{net}} = \{\{i\} | i \in \mathcal{N}\}$.

Once the parameters are initialized, the iterative phase of the algorithm starts. First, the weakest cluster $\mathcal{C}_{\text{weakest}}$ is selected among the clusters. Then, to search for a better cluster configuration, we merge the weakest cluster with another cluster, referred to as the ''partner'' cluster, to create a ''merged'' cluster with the longest possible lifetime compared to other combinations. As a result, the cluster $\mathcal{C}_{\text{weakest}}$ and the cluster $\mathcal{C}_{\text{partner}}$ no longer exist in the network as two distinct clusters, and the newly formed merged cluster $\mathcal{C}_{\text{merged}}$ appears. The former two clusters are removed from the family $\mathcal{F}_{\text{net}}$, and the latter is added. Subsequently, the network lifetime L_{net} is updated based on the modified $\mathcal{F}_{\text{net}}$. If the updated lifetime L_{net} is longer than $L_{\text{net}}^{\text{opt}}$, the latter is updated with the former, making the current configuration $\mathcal{F}_{\text{net}}$ provisionally the optimal cluster configuration $\mathcal{F}_{\text{net}}^{\text{opt}}$.

The clustering iterations repeat until the whole network is rendered into a single cluster that contains all the nodes in the network. Each iteration is not guaranteed to yield a higher network lifetime than obtained in previous steps. Through the iterations, however, there is a step at which the network lifetime reaches its maximum level. This step corresponds to the optimal network lifetime, denoted as $L_{\text{net}}^{\text{opt}}$, and the cluster formation at this step is considered the optimal cluster formation, denoted as $\mathcal{F}_{\text{net}}^{\text{opt}}$.

Fig. 1 summarizes the procedure of HAC clustering.

C. EXAMPLE

Let us consider the following example to visualize how clusters are formed in Algorithm 1. Fig. 2(a) shows the node distribution of a network consisting of six randomly deployed nodes. Initially, HAC treats each individual node as a cluster with a single node, as shown in Step 0 of Table 2.

To boost the lifetime of the cluster with the lowest lifetime, HAC merges this cluster with the best candidate cluster that

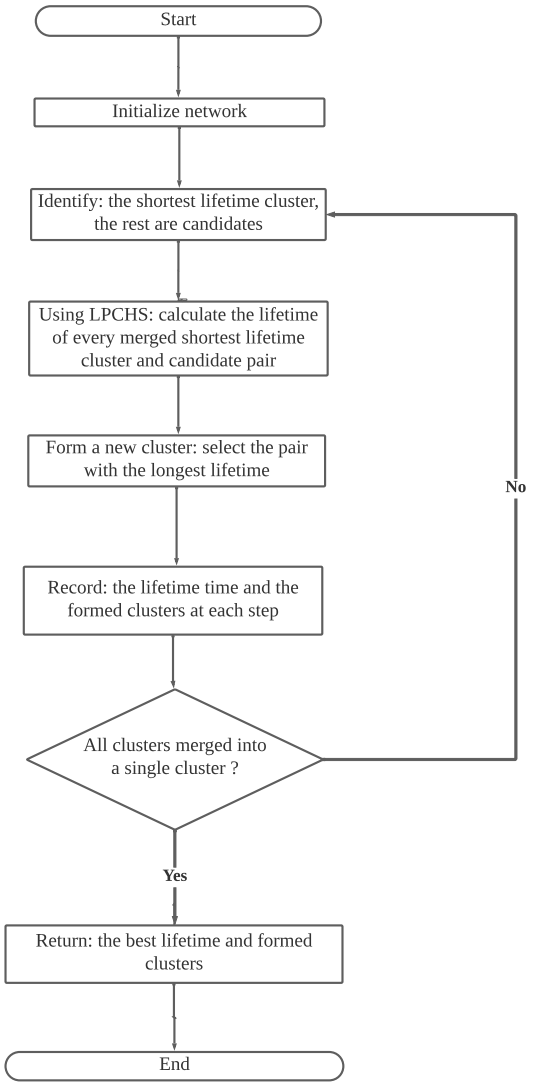


FIGURE 1. Flowchart of HAC clustering.

achieves the best lifetime of the newly formed cluster $\mathcal{C}_{\text{merged}}$. At Step 1 in Table 2, we can see that Cluster 1 has the lowest lifetime with 50 transmission rounds. We perform the first step by computing the expected lifetime of merging Cluster 1 with all possible cluster candidates in the network by calculating their expected lifetime if merged into a cluster. We found that the best partner for Cluster 1 is Cluster 2. Thus, Clusters 1 and 2 are merged into a new cluster as Cluster 6. The attributes of the formed clusters and their layout, after the execution of the first step, are presented at Step 1 of Table 2 and Fig. 2(b), respectively.

The procedure is then repeated as there are currently five clusters. Once again, the cluster with the shortest lifetime, Cluster 5, is merged with the best candidate cluster, in this situation Cluster 0, forming Cluster 7 as a new cluster. Steps 1 through 5 in Table 2 provide a summary of the properties of the current clusters, including the newly formed clusters, until the final step, where the entire network becomes a single cluster. Fig. 2(a) through Fig. 2(f) depict the layout

Algorithm 1 Clustering Process

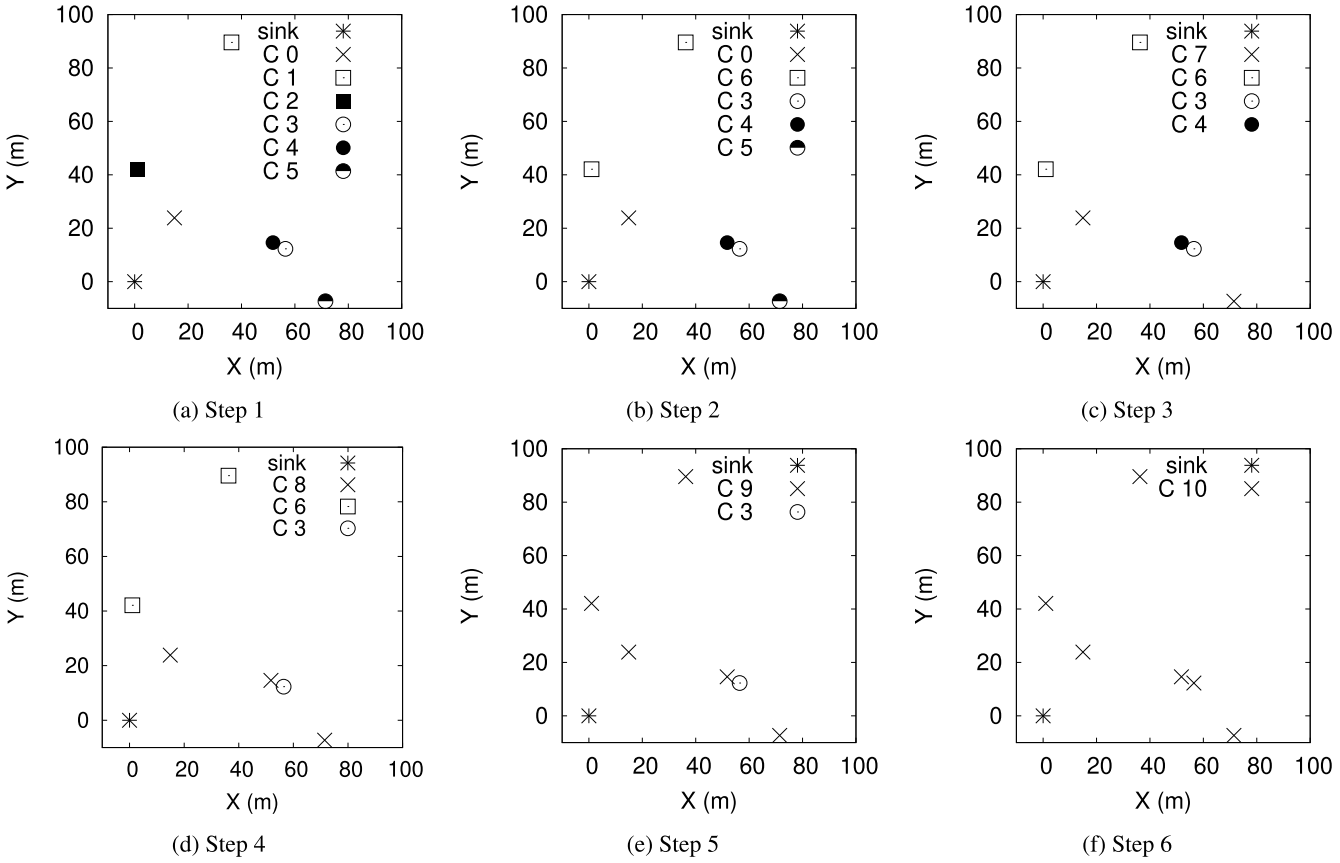
Require: the set $\mathcal{N}$ of nodes in the network
Ensure: formed clusters

$\mathcal{F}_{\text{net}} \leftarrow \{\{i\} | i \in \mathcal{N}\}$
$\mathcal{F}_{\text{net}}^{\text{opt}} \leftarrow \mathcal{F}_{\text{net}}$
$L_{\text{net}}^{\text{opt}} \leftarrow \min_{\mathcal{C} \in \mathcal{F}_{\text{net}}} L_{\mathcal{C}}$
function CLUSTERING($\mathcal{N}$)
  while $|\mathcal{F}_{\text{net}}| > 1$ **do**
    $\mathcal{C}_{\text{weakest}} \leftarrow \text{argmin}_{\mathcal{C} \in \mathcal{F}_{\text{net}}} L_{\mathcal{C}}$
    $\mathcal{C}_{\text{partner}} \leftarrow \text{argmax}_{\mathcal{C} \in \mathcal{F}_{\text{net}} \setminus \{\mathcal{C}_{\text{weakest}}\}} L_{\mathcal{C} \cup \mathcal{C}_{\text{weakest}}}$
    $\mathcal{C}_{\text{merged}} \leftarrow \mathcal{C}_{\text{weakest}} \cup \mathcal{C}_{\text{partner}}$
    $\mathcal{F}_{\text{net}} \leftarrow \mathcal{F}_{\text{net}} \setminus \{\mathcal{C}_{\text{weakest}}\}$
    $\mathcal{F}_{\text{net}} \leftarrow \mathcal{F}_{\text{net}} \setminus \{\mathcal{C}_{\text{partner}}\}$
    $\mathcal{F}_{\text{net}} \leftarrow \mathcal{F}_{\text{net}} \cup \{\mathcal{C}_{\text{merged}}\}$
    $L_{\text{net}} \leftarrow \min_{\mathcal{C} \in \mathcal{F}_{\text{net}}} L_{\mathcal{C}}$
    if $L_{\text{net}} > L_{\text{net}}^{\text{opt}}$ **then**
      $\mathcal{F}_{\text{net}}^{\text{opt}} \leftarrow \mathcal{F}_{\text{net}}$
      $L_{\text{net}}^{\text{opt}} \leftarrow L_{\text{net}}$
    end if
  end while
  return $\mathcal{F}_{\text{net}}^{\text{opt}}$
end function

TABLE 2. Formed members at each step.

Step	Cluster-ID	Size	Lifetime	Member nodes
0	0	1	386	0
	1	1	50	1
	2	1	219	2
	3	1	130	3
	4	1	147	4
	5	1	88	5
1	0	1	386	0
	3	1	130	3
	4	1	147	4
	5	1	88	5
	6	2	110	1 and 2
2	3	1	130	3
	4	1	147	4
	6	2	110	1 and 2
	7	2	107	0 and 5
3	3	1	130	3
	6	2	110	1 and 2
	8	2	128	4, 0, and 5
4	3	1	130	3
	9	5	98	1, 2, 4, 0, and 5
5	10	5	96	3, 1, 2, 4, 0, and 5

of the formed clusters throughout HAC clustering. From the whole clustering process displayed in Table 2, we identify the optimal cluster formation to be the configuration from Step 3. This is because the shortest lifetime observed at Step 3, which is 110 rounds, is the longest among the shortest lifetimes observed at each step. In all other steps, there is at least one cluster with a lifetime of less than 110 rounds. Therefore, the most favorable cluster formation is achieved at Step 3, where the network is divided into three clusters. The dendrogram in Fig. 3 illustrates the sequence of cluster formation and the optimal configuration.



(a) Step 1 (b) Step 2 (c) Step 3

(d) Step 4 (e) Step 5 (f) Step 6

FIGURE 2. Example of steps of HAC algorithm.

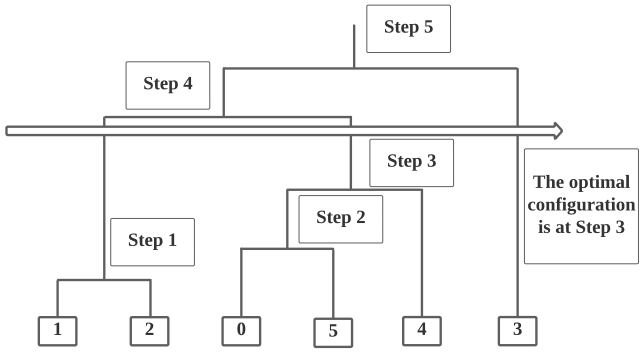


FIGURE 3. The dendrogram of cluster formation in HAC.

D. COMPLEXITY ANALYSIS

The linear programming model is responsible for selecting the cluster heads and the proportion of time they serve in a cluster.

Solving the linear programming problem involves assigning the proportion of cluster head duty within a cluster if performed with complexity $O(n)$, where n is the number of nodes participating in the cluster, i.e., $n = \mathcal{N}$. As seen from Fig. 1, cluster formation is an iterative process where the cluster with the shortest lifetime is merged with each cluster of the candidate nodes. Merging the cluster with the lowest lifetime with every candidate requires $(n-1)$ merge operations. This process is repeated for all the candidate clusters, resulting in a total of $(n-1) \times n$ merges, simplified to $O(n^2)$. Once we form a new cluster, the size of the network, n, is reduced by one, and a new cluster with the lowest lifetime is identified. The merging process of the cluster with the lowest lifetime with the candidates is repeated until the entire network is rendered into a single cluster. This adds an extra complexity to the order $O(n)$, which results in a final complexity of $O(n^3)$.

IV. SIMULATION SETUP

A. NETWORK LAYOUT

The proposed clustering approach was tested using four different node layouts: ''normal-center'', ''normal-corner'', ''uniform-center'' and ''uniform-corner''. The terms ''normal'' and ''uniform'' refer to how the nodes were randomly deployed within the sensed area. In the ''normal'' distribution, the nodes were deployed according to a normal distribution, while in the ''uniform'' distribution, the nodes were deployed uniformly across the area. The terms ''center'' and ''corner'' indicate the placement of the sink. In the ''center'' layout, the sink was positioned at the center of the sensed area, whereas in the ''corner'' layout, the sink was placed at one of the corners of the area. The network

comprises a hundred nodes deployed in an area of 100 m by 100 m.

B. ENERGY MODEL

The HAC technique is introduced to enhance the performance of LPCHS by incorporating a cluster formation method based on LPCHS's findings. To illustrate the impact of HAC on LPCHS, we utilized the same radio transmission/reception models as described in [16]. In [16], a first-order radio transmission model was used, where the energy required to transmit a k-bit data packet, denoted as E_{tx}, is modeled as:

$$E_{\text{tx}} = k\left(e_{\text{elec}} + \epsilon_{\text{amp}} d^p\right), \tag{9}$$

where the symbols ϵ_{amp}, e_{elec}, d, and p represent the transmit amplifier efficiency, the sensor hardware electronics energy consumption, the distance between the transmitter and the receiver, and the path loss exponential factor, respectively. On the other hand, the amount E_{rx} of energy spent to receive a k-bit packet is modeled by

$$E_{\text{rx}} = k e_{\text{elec}}. \tag{10}$$

For this study, it is assumed that the wireless environment resembles free space, and as a result, the value of the path loss exponential factor p is set to two. Additionally, the study assumes that each sensor node has a transmission range, allowing it to communicate directly with the sink. Therefore, any sensor node could serve as a cluster head if selected. Furthermore, each node is assumed to have an initial energy of E_i^{init} equal to 1 Joule. Table 3 presents the values of the transmission parameters used for simulating packet transmission.

Similar to the assumption made in [16], all member nodes are assumed to transmit packets of data with the same length k to the cluster head. Subsequently, the cluster head receives and concatenates all the data from the member nodes, adds its sensed data, and transmits the combined data to the sink. For example, in a cluster comprising n member nodes, the cluster head is required to send $n \times k$ bits of data to the sink. It should be noted that the data packet length includes both the information sensed by the node and the message header.

To select the cluster head, we rely on the linear programming model presented in Equation (4). In [16], the terms in this equation are modeled as follows:

- $E_{i\to\text{sink}}$ represents the energy required to transmit data from the cluster head to the sink. Since it is related to transmission energy, it can be calculated using Equation (9). The cluster head must concatenate the $n \times k$ packets of data received from the member nodes and then transmit these packets to the sink:

$$E_{i\to\text{sink}} = nk\left(e_{\text{elec}} + \epsilon_{\text{amp}} d^p\right). \tag{11}$$

- $E_{i\leftarrow j}$ represents the amount of energy that a cluster head i will consume to receive data from member node j. In this scenario, the cluster head receives a packet of data of length k from each member in the cluster, except itself.

TABLE 3. Radio transmission model's parameters.

Parameter	Value
e_{elec}	50 nJ/bit
ϵ_{amp}	100 pJ/bit/m^2
p	2
Packet length	2000 bits
Initial energy of each node	1 J/node

This energy consumption can be calculated using the receive data model presented in Equation (10):

$$E_{i\leftarrow j} = (n-1)k\, e_{\text{elec}}. \tag{12}$$

- $E_{i\to j}$ represents the amount of energy consumed by the member node i to transmit data to the current cluster head j. This energy expenditure is modeled using Equation (9) because it pertains to transmission energy. In this context, node i is transmitting a single packet of data to cluster head j:

$$E_{i\to j} = k\left(e_{\text{elec}} + \epsilon_{\text{amp}} d^p\right). \tag{13}$$

C. SINK-BASED COMPUTATION

In this study, we assume that the sink has a largely sufficient energy budget. To reduce the power consumption of the sensor nodes, all the computations required in HAC are performed at the sink. Node attributes and the network size and boundaries are given as input to the sink. The sink is also responsible for sending control messages to cluster heads and member nodes. IBM CPLEX optimizer was used to solve the linear programming model in this research.

V. RESULTS

To effectively demonstrate our proposed method's efficiency, we compare the results obtained from our approach (HAC) against a well-known clustering technique, LEACH, as well as our previous work, which combines the clustering method of LEACH with the cluster head selection using linear programming (LPCHS). We compare these three approaches using four distinct types of node layouts, as elaborated in Section IV. The simulation was executed across 25 randomly generated topologies for each layout to ensure sufficient confidence in the results.

The results are presented in Fig. 4. This whisker bar chart shows various statistics of network lifetime, including the first quartile, median, third quartile, and maximum values, obtained over the 25 network topologies we generated for each algorithm. Network lifetime is defined in terms of FND and points corresponding to 30 percent, 50 percent, and 70 percent of node death.

Fig. 4(a) depicts the performance of the three algorithms in the normal-center scenarios. This figure reveals that our proposed algorithm, HAC, outperforms the other two approaches in all the lifetime definitions. The median FND is at round 180 for HAC, followed by LPCHS at 130 rounds and 83 rounds for LEACH. HAC still secures the best

performance among the three algorithms with 247 rounds before the death of 30 percent of nodes in the network, while the other two algorithms attained almost the same number of rounds with 154 rounds and 156 rounds for LPCHS and LEACH. By the time the death of 50 percent death occurred, HAC would have secured 291 rounds of transmissions, and LEACH would surpass LPCHS with 173 rounds and 158 rounds, respectively. HAC still outperforms the other two approaches at the death of 70 percent of nodes with a lifetime of 397 rounds, 191 rounds for LEACH, and 158 rounds for LPCHS.

Fig. 4(b) shows the results in the uniform-center scenarios. The figure mentioned above reveals the supremacy of HAC over LPCHS and LEACH. The FND occurrence is at 197 rounds for HAC, followed by LPCHS with 110 rounds, then LEACH with no more than 65 rounds. At the death of 30 percent of nodes, LPCHS and LEACH achieve almost the same lifetime with 119 and 118 rounds, respectively. Despite the increase in the performance of LEACH at the death of 30 percent of nodes, the lifetime obtained by HAC still almost doubles that of LEACH with 212 rounds. At the death of 50 percent of nodes, LEACH outperformed LPCHS with 134 rounds and 124 rounds, respectively, and HAC is still on top of the three with 226 rounds. HAC guaranteed a lifetime of 239 rounds, 147 rounds for LEACH, and 140 rounds for LPCHS at the death of 70 percent of nodes.

Fig. 4(c) displays the results in the normal-corner scenarios. Results from the graph above are characterized by the steadiness of the lifetime obtained from HAC and LPCHS over the death of nodes in the network. On the contrary, the lifetime obtained by LEACH drastically increased over the course of node death. In the cases where FND defines the network lifetime, HAC was the approach that best extends the network's lifetime with 80 rounds, followed by LPCHS with 50 rounds and LEACH with no more than 11 rounds. At the death of 30 percent of nodes, the lifetime of LEACH rapidly reaches 77 rounds, surpassing LPCHS, which has a lifetime of 50 rounds, while HAC is still leading the performance with a lifetime of 83 rounds. At the death of 50 percent of the nodes, LEACH attained 85 rounds, HAC reached 93 rounds, and LPCHS did not surpass 62 rounds. LEACH secured 195 rounds, HAC guaranteed 87 rounds, 68 for LPCHS at the death of 70 percent of nodes.

Fig. 4(d) depicts the results of the uniform-corner scenario. Similar to the previous case of the normal-corner, the outputs obtained from the simulation of uniform-corner network layout reveal a remarkable amelioration of the performance of LEACH over the death of nodes in the network. Both HAC and LPCHS performance is characterized by monotony and slow progress. HAC guaranteed a lifetime of 78 rounds before FND, 40 rounds for LPCHS, and only 3 rounds for LEACH. At the death of 30 percent of nodes, LEACH quickly reached 52 rounds, which is only seven rounds behind LPCHS with 59 rounds. HAC has a lifetime of 82 rounds at the death of 30 percent of nodes. HAC is still on top with 84 rounds, LEACH with 71 rounds, and LPCHS with 59 rounds at the death of 50 percent of nodes. At the death of 70 percent of nodes, LEACH escalated to reach a lifetime of 224 rounds, followed by HAC with 86 rounds, then LPCHS with 62 rounds.

The simulation results reveal that HAC outperforms LPCHS under all network layout scenarios and for all network lifetime definitions. HAC also outperforms LEACH in all network scenarios where the sink is placed at the center of the network and for all definitions of the network's lifetime. In the network layout cases where the sink is placed at the corner of the network, HAC outperformed LEACH almost up to the death of 50 percent of nodes. Then, LEACH outperformed HAC from the death of 50 percent until the end of the last node. LPCHS outperformed LEACH for all cases where the network's lifetime was defined by the FND occurrence or the death of 30 percent of nodes in the network. LPCHS is an efficient cluster head selection method but does not propose a cluster formation method. In the study presented by [16], LPCHS used LEACH techniques to form clusters. In HAC, we proposed a cluster formation method customized for LPCHS, and this proposed method outperformed LPCHS and LEACH.

VI. DISCUSSION

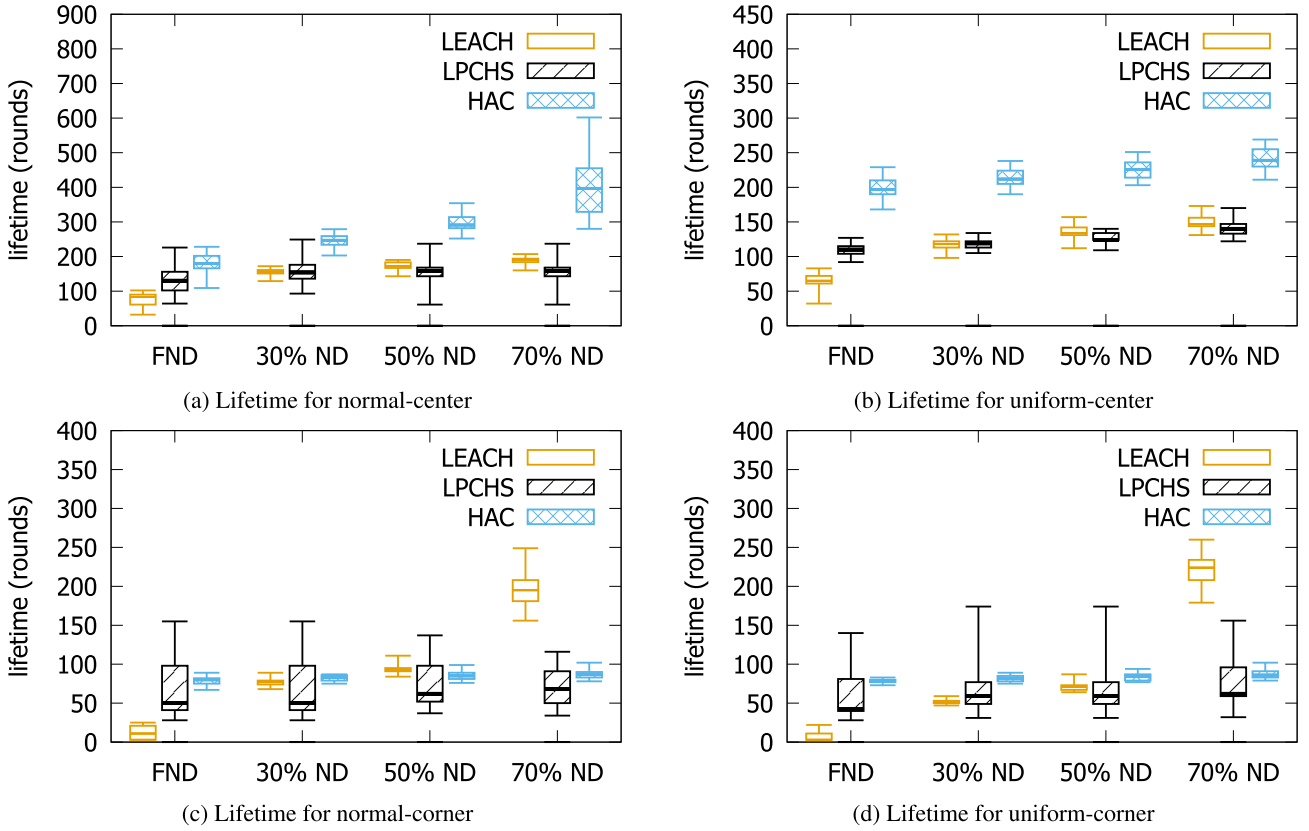
A. INFLUENCE OF THE SINK LOCATION

The cluster size and its distance to the sink are among the factors that should be tuned carefully to optimize energy consumption in WSNs. The systematic cluster head selection and cluster formation method enabled HAC to over-output LEACH and LPCHS. LEACH does not take preventive measures to limit the size of a cluster whose head is located geographically far from the sink. In such a situation, it is possible to have clusters of large size and with cluster heads far from the sink. This situation leads to a quick discharge of a large group of nodes after a few rounds of data transmission between the network and the sink.

We note that the location of the sink with regard to the entire network has a significant impact on its lifetime. Indeed, the energy consumption of nodes is affected by the distance of transmission. For instance, the nodes spend less energy to transmit to a sink in the center of the network than the amount of energy consumed to transmit to a sink at the network's edge. The impact of sink placement manifested in the performance of HAC versus the performance of the two other approaches. We can confirm with Fig. 4(c) and Fig. 4(d) that HAC outperforms LEACH almost up to the death of 50 percent of nodes, whereas Fig. 4(a) and Fig. 4(b) reveal that when the sink is located at the center of the network, HAC outperforms LEACH in all definitions of network lifetime.

B. SINGLE-NODE CLUSTERS

To understand the causes of the dominance of HAC over LEACH in the cases where the sink is at the center of the network, we need to examine the number and sizes of the clusters formed by HAC. Unlike LEACH and other clustering



(a) Lifetime for normal-center

(b) Lifetime for uniform-center

(c) Lifetime for normal-corner

(d) Lifetime for uniform-corner

FIGURE 4. Lifetime of the three algorithms under different network topologies.

techniques that previously set the number of clusters, HAC does not force the network to have a particular number of clusters. Initially, HAC considers every single node as a cluster, then only merges nodes that contribute to extending the network lifetime into clusters. This situation caused HAC to cluster a network of 100 nodes into a large number of small-size clusters. Regardless of the node deployment, many of those clusters are clusters with a single node. In other words, HAC keeps many nodes alone to transmit directly to the sink.

At first thought, those single-node clusters might be regarded as a burden to the network and diagnosed as isolated nodes. In reality, this appears to be the correct decision. Let us consider an example to examine the importance of the single-node clusters and their contribution to extending the network lifetime.

Let us consider a normal-centered topology. After applying the HAC clustering algorithm, the network lifetime depending on its definition is summarized in Table 4. Table 5 shows the number of clusters and sizes, and Fig. 5 depicts the node's location of each group of clusters. HAC clusters the 100-node network into 69 clusters, with 46 of those 69 clusters being single-node clusters and 18 clusters made of only two nodes.

We can see that the majority of those single-node clusters are located at the center of the network, while the non-single-node clusters have member nodes that are close to the sink merged with other nodes that are far from the sink. Since the nodes that are far from the sink are expected to have short lifetimes, it is expected that those far nodes will seek to merge with nodes that are closer to the sink to mitigate their lifetime. As a result, many nodes in the middle of the network remain single-node clusters, as the clusters that are further away from the sink are the ones with the shorter lifetimes that HAC focuses on. As long as there are other nodes further away from the sink, the nodes towards the middle of the network will remain as "clusters" of a single node.

This is confirmed by looking at the lifetime of these single-node clusters. As seen from the table above, all of the single-node clusters have a lifetime above the one of the network at the death of 10 percent of nodes. Thirty-one out of the 46 single-node clusters secured a lifetime beyond the network's lifetime at the death of 50 percent of nodes in the whole network. By the time 70 percent of the nodes died in the whole network, more than half of the single nodes were still alive and contributing to the lifetime.

We conclude that the clusters composed of multiple nodes boost the network lifetime in cases where the network lifetime is determined by a low number of nodes' deaths, such as FND and 30 percent of nodes' deaths. For the cases where the lifetime is determined by a high number of nodes' deaths, such as 50 percent and 70 percent of nodes' deaths, the

TABLE 4. HAC: Nodes' death over the number of rounds for normal-center scenarios.

Nodes' death	Number of rounds
First node death	228
30 %	273
50 %	314
70 %	334
Last node death	906

TABLE 5. HAC: Formed clusters and their sizes.

Cluster size	Cluster count
1	46
2	18
3	3
4	1
5	1

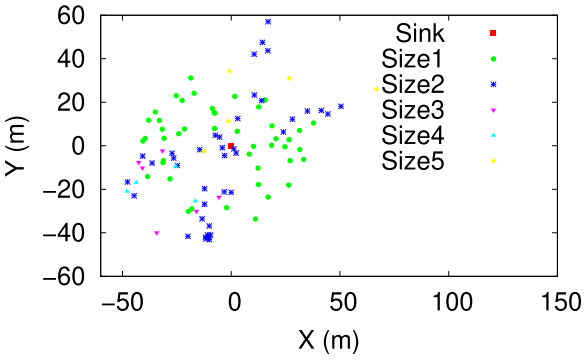


FIGURE 5. Distribution of nodes in the formed clusters.

single-node clusters, which are 46 percent of the total number of nodes in the network and have a high lifetime expectancy, boost up the network's lifetime.

We have seen how the single-node clusters helped HAC to dominate the other algorithms in the case of the normal-center scenarios. The same applies to the other topologies, with the distinction that topologies with the sink located in the corner naturally result in more nodes being far away from the sink, resulting in more clusters being formed and fewer nodes remaining single.

C. ROBUSTNESS

As seen in Section III-B, HAC is a heuristic approach based on forming clusters based on the exact solution of the linear model obtained from LPCHS. Hai [30] highlighted that sub-optimality is a serious concern in such approaches. The robustness against various input sizes of networks is the main problem in the performance evaluation of heuristic approaches.

To test the robustness of HAC against various network sizes, we take the following measures:

- Compare the performance of HAC against the exact solution for small-size networks.
- Compare the performance of HAC across varied network sizes.

TABLE 6. Number of single node clusters formed through HAC that have a longer lifetime than the lifetime of the network depending on the definition used.

Lifetime definition	Nb of single-node clusters with longer lifetime than the network
First node death	46
30 %	46
50 %	31
70 %	25

1) COMPARISON BETWEEN HAC AND THE OPTIMUM SOLUTION

As seen in Section I, finding the exact solution to the clustering problem is not feasible for large-size networks as the complexity exhibits a non-polynomial growth. However, to examine the ability of HAC to attain a near-optimal performance, we compared its performance to the exact solution for small-size networks. To obtain the exact solution, we investigated all possible cluster combinations that result from partitioning a network of ten nodes into all possible clusters (from a single cluster up to ten clusters). Fig. 6 illustrates the performance of both approaches.

The results above reveal that HAC is capable of attaining a near-optimal performance for small-size networks. However, that does not guarantee that HAC will succeed in maintaining this high performance for large-size networks.

2) PERFORMANCE OF HAC UNDER VARIOUS NETWORK SIZES

To demonstrate its robustness against various network sizes, we compare the performance of HAC across various network sizes. In Section IV, we executed HAC on a network of 100 nodes deployed in an area of 100 m by 100 m with a node density of 1 node per 100 squared meters. The initial energy of nodes in this network is 1 Joule per node. Taking the setup in Section IV as our reference, we execute HAC on a set of different-size networks: a 40-node network, a 60-node network, an 80-node network, a 120-node network, and a 140-node network. To maintain consistency between the setup of the reference experiment in Section IV and the scaled network, we scale the area of the newly generated different-size networks to maintain a density of one node per 100 squared meters. We also scale the initial energy of nodes in the newly generated different-size networks.

The initial energy of nodes is proportionally scaled with the scale of the generated network number of nodes and area. As a result of scaling down the network area, nodes will be relatively closer to the sink. As the energy model scales with the square of the distance, we adjust the initial energy level with the surface area square. For instance, if we consider our reference of a 100 node-network, with an area deployed in an area of 100 m by 100 m, and initial energy of 1 Joule per node as a reference, the 40-node network deployed in 4000 squared meters will have $0.63 = \sqrt{4000/10000}$ Jules as initial energy. The size, area, and initial energy of the generated networks are summarized in Table 7.

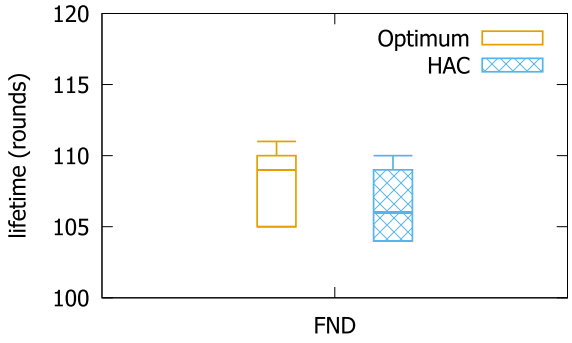


FIGURE 6. Lifetime comparison between HAC and the optimum solution.

TABLE 7. Attributes of scaled networks.

Network	Size (nodes)	Area (m^2)	Initial energy (Joule/node)
1	40	4000	0.63
2	60	6000	0.77
3	80	8000	0.89
4	100	10 000	1.00
5	120	12 000	1.09
6	140	14 000	1.18

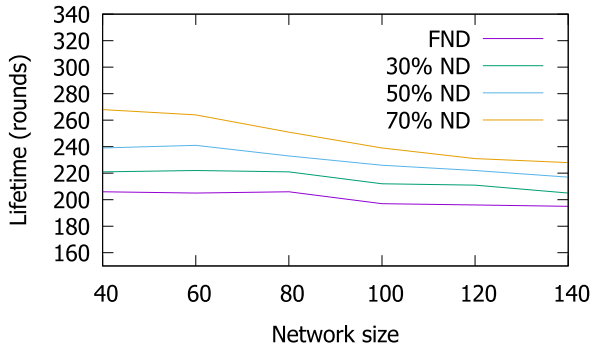


FIGURE 7. Network lifetime under different input sizes.

The performance of HAC on these scaled networks is summarized in Fig. 7. Fig. 7 shows the FND, 30 percent node death, 50 percent node death, and 70 percent node death of HAC under the six scaled networks. The FND lifetime definition exhibits small changes in its value for the networks with sizes between 80 and 100 nodes. This change does not exceed ten rounds. In the other segments, the FND has almost the same value for all the scaled networks. The other lifetime definitions (30 percent node death, 50 percent node death, and 70 percent node death) exhibit decreasing performance for larger-size networks. However, our primary concern in this study is the performance of FND. Since FND succeeds in approximately maintaining the same performance for all network sizes, we can conclude that HAC is robust against various network sizes.

VII. CONCLUSION AND FUTURE WORK

In this study, we proposed the Hierarchical Agglomerative Clustering (HAC) method, a novel clustering method to optimize the energy consumption in wireless sensor networks. HAC was designed to leverage the LPCHS cluster head selection method to further improve the efficiency of the network by taking into account the lifetime of the clusters during their formation.

HAC presents a novel clustering method that forms clusters that are not based on the location of the cluster head. The sparsity of clusters in the network area is based on the trade-off between the energy spent to transmit data between cluster members and the energy spent to transmit from the cluster head to the sink. Balancing the energy consumed by inter-cluster and intra-cluster communication resulted in balanced energy consumption in HAC.

Simulation results revealed that HAC outperforms LPCHS under all network topologies and for all definitions of network lifetime. HAC also outperformed LEACH when executed to all input sets with the sink located at the center of the network. In cases where the sink is at the network's edge, HAC still outperforms LEACH up to the death of half of the network nodes.

The scope of this study was restricted to single-hop transmission wireless networks. However, in real-life applications, there might be cases where some node's transmission range cannot reach the sink, thereby requiring them to use multi-hop transmission schemes to transmit their data. In future work, we want to extend this novel clustering approach to multi-hop clustering.

REFERENCES

[1] M. M. Warrier and A. Kumar, "Energy efficient routing in wireless sensor networks: A survey," in *Proc. Int. Conf. Wireless Commun., Signal Process. Netw. (WiSPNET)*, Mar. 2016, pp. 1987–1992.

[2] M. K. Singh, S. I. Amin, S. A. Imam, V. K. Sachan, and A. Choudhary, "A survey of wireless sensor network and its types," in *Proc. Int. Conf. Adv. Comput., Commun. Control Netw. (ICACCCN)*, Oct. 2018, pp. 326–330.

[3] G. C. Jagan and P. J. Jayarin, "Wireless sensor network cluster head selection and short routing using energy efficient ElectroStatic discharge algorithm," *J. Eng.*, vol. 2022, pp. 1–10, Feb. 2022.

[4] R. Choudharyl, S. Kumar, A. Deepak, and D. Dash, "Data aggregation in wireless sensor network: An integer linear programming formulation for energy optimization," in *Proc. 13th Int. Conf. Wireless Opt. Commun. Netw. (WOCN)*, Jul. 2016, pp. 1–6.

[5] S. A. Sibi and R. V. Prabhu, "Survey on clustering and depletion of energy in wireless sensor network," in *Proc. 3rd Int. Conf. Intell. Sustain. Syst. (ICISS)*, Dec. 2020, pp. 1341–1345.

[6] A. Shahraki, A. Taherkordi, Ø. Haugen, and F. Eliassen, "Clustering objectives in wireless sensor networks: A survey and research direction analysis," *Comput. Netw.*, vol. 180, Oct. 2020, Art. no. 107376.

[7] R. Samir, M. S. El-Mahallawy, S. M. Gasser, and N. Zaher, "Exploring the effect of various cluster structures on energy consumption and end-to-end delay in cognitive radio wireless sensor networks," *IEEE Access*, vol. 6, pp. 38062–38070, 2018.

[8] A. S. Raghuvanshi, S. Tiwari, R. Tripathi, and N. Kishor, "Optimal number of clusters in wireless sensor networks: An FCM approach," in *Proc. Int. Conf. Comput. Commun. Technol. (ICCCT)*, Sep. 2010, pp. 817–823.

[9] S. K. Singh, P. Kumar, and J. P. Singh, "A survey on successors of LEACH protocol," *IEEE Access*, vol. 5, pp. 4298–4328, 2017.

[10] D. G. Melese, H. Xiong, and Q. Gao, "Consumed energy as a factor for cluster head selection in wireless sensor networks," in *Proc. 6th Int. Conf. Wireless Commun. Netw. Mobile Comput. (WiCOM)*, Sep. 2010, pp. 1–4.

[11] B. C. Rennie and A. J. Dobson, "On stirling numbers of the second kind," *J. Combinat. Theory*, vol. 7, no. 2, pp. 116–121, Sep. 1969.

[12] F. Liu and Y. Chang, "An energy aware adaptive kernel density estimation approach to unequal clustering in wireless sensor networks," *IEEE Access*, vol. 7, pp. 40569–40580, 2019.

[13] A. Chniguir and Z. Ben Jemaa, "Lifetime maximization of the first dead node in a WSN," in *Proc. Int. Wireless Commun. Mobile Comput. (IWCMC)*, May 2022, pp. 378–383.
[14] S. Bhushan, R. Pal, and S. G. Antoshchuk, "Energy efficient clustering protocol for heterogeneous wireless sensor network: A hybrid approach using GA and K-means," in *Proc. IEEE 2nd Int. Conf. Data Stream Mining Process. (DSMP)*, Aug. 2018, pp. 381–385.
[15] K. Biswas, V. Muthukkumarasamy, X.-W. Wu, and K. Singh, "An analytical model for lifetime estimation of wireless sensor networks," *IEEE Commun. Lett.*, vol. 19, no. 9, pp. 1584–1587, Sep. 2015.
[16] M. D. S. Mohamed, F. Patrick, and C. Ohta, "LPCHS: Linear programming based cluster head selection method in wireless sensor networks," *IEICE Commun. Exp.*, vol. 12, no. 9, pp. 511–516, 2023.
[17] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proc. 33rd Annu. Hawaii Int. Conf. Syst. Sci.*, 2000, p. 10.
[18] J. Mao, M. Gu, and Y. Huo, "Improved routing algorithm for wireless sensor networks based on LEACH," in *Proc. Int. Commun. Eng. Cloud Comput. Conf. (CECCC)*, Oct. 2022, pp. 33–36.
[19] A. Bhih, W. Abushiba, and A. Elashheb, "An improved leach algorithm based on hierarchical clustering approach for wireless sensor network application," in *Proc. IEEE 5th Int. Conf. Electron. Commun. Eng. (ICECE)*, Dec. 2022, pp. 78–83.
[20] M. Adnan, L. Yang, T. Ahmad, and Y. Tao, "An unequally clustered multi-hop routing protocol based on fuzzy logic for wireless sensor networks," *IEEE Access*, vol. 9, pp. 38531–38545, 2021.
[21] A. Quanbiao, W. Muqing, and L. Sixu, "Research on unequal clustering protocol based on fuzzy logic and entropy weight method," in *Proc. 7th Int. Conf. Comput. Commun. (ICCC)*, Dec. 2021, pp. 215–219.
[22] S. S. Nidhi, K. Kumar, and D. Sridharan, "Uniform and randomly distributed fuzzy logic based clustering in wireless sensor networks," in *Proc. 12th Int. Conf. Adv. Comput. (ICoAC)*, Aug. 2023, pp. 1–6.
[23] J. Singh, S. S. Yadav, V. Kanungo, and V. Pal, "A node overhaul scheme for energy efficient clustering in wireless sensor networks," *IEEE Sensors Lett.*, vol. 5, no. 4, pp. 1–4, Apr. 2021.
[24] D. Sivakumar, S. S. Devi, and T. Nalini, "Energy aware metaheuristics unequal clustering protocol for WSN," in *Proc. 2nd Int. Conf. Artif. Intell. Smart Energy (ICAIS)*, Feb. 2022, pp. 1418–1424.
[25] S. Jan and M. Masood, "Multiple solutions based particle swarm optimization for cluster-head-selection in wireless-sensor-network," in *Proc. Int. Conf. Digit. Futures Transformative Technol. (ICoDT2)*, May 2021, pp. 1–5.
[26] G. Jin and W. Muqing, "Genetic-based cluster routing algorithm for wireless sensor networks," in *Proc. 7th Int. Conf. Comput. Commun. (ICCC)*, Dec. 2021, pp. 48–52.
[27] Z. Eskandari, S. A. H. Seno, M. Shenify, and R. Budiarto, "Optimal cluster head selection in wireless sensor networks using integer linear programming techniques," *Int. J. Informat. Commun. Technol. (IJ-ICT)*, vol. 3, no. 3, p. 186, Dec. 2014.
[28] J. A. P and U. Sakthivel, "CFCLP–A novel clustering framework based on combinatorial approach and linear programming in wireless sensor network," in *Proc. 2nd Int. Conf. Comput. Commun. Technol. (ICCCT)*, Feb. 2017, pp. 49–54.
[29] C. Li, J. Bai, J. Gu, X. Yan, and Y. Luo, "Clustering routing based on mixed integer programming for heterogeneous wireless sensor networks," *Ad Hoc Netw.*, vol. 72, pp. 81–90, Apr. 2018.
[30] D. T. Hai, "On Achilles heel of some optical network designs and performance comparisons," *Opt. Quantum Electron.*, vol. 54, no. 2, p. 69, Feb. 2022.

SIDI MOHAMED MOHI DINE was born in Nouakchott, Mauritania, in 1992. He received the B.E. degree in mechatronics engineering from International Islamic University Malaysia, in 2016, and the M.S. degree in mechatronics engineering from Newcastle University, in 2019. He is currently pursuing the Ph.D. degree with the Graduate School of System Informatics, Kobe University.

PATRICK FINNERTY was born in Cholet, France, in 1995. He received the French Engineering degree in computer science and information technology from INSA Lyon, in 2018, and the Ph.D. (Eng.) degree from Kobe University, Japan, in 2022. Since February 2022, he has been an Assistant Professor with the Graduate School of System Informatics, Kobe University. His research interests include parallel and distributed computing techniques and distributed systems. He is a member of IPSJ, the IEEE Computer Society, and ACM.

CHIKARA OHTA (Member, IEEE) was born in Osaka, Japan, in 1967. He received the B.E., M.E., and Ph.D. (Eng.) degrees in communication engineering from Osaka University, Osaka, in 1990, 1992, and 1995, respectively. In April 1995, he was an Assistant Professor with the Department of Computer Science, Faculty of Engineering, Gunma University. In October 1996, he was a Lecturer with the Department of Information Science and Intelligent Systems, Faculty of Engineering, University of Tokushima. In March 2001, he was an Associate Professor. In November 2002, he was an Associate Professor with the Department of Computer and Systems Engineering, Faculty of Engineering, Kobe University. From March 2003 to February 2004, he was a Visiting Scholar with the University of Massachusetts, Amherst, MA, USA. In April 2010, he was an Associate Professor with the Graduate School of System Informatics, Kobe University. In January 2015, he was a Professor. In April 2016, he was a Professor with the Graduate School of Science, Technology and Innovation, Kobe University. Since April 2022, he has been a Professor with the Graduate School of System Informatics, Kobe University. His current research interest includes the performance evaluation of communication networks. He is a member of IPSJ and ACM SIGCOMM.