


# An Energy Efficient Data Transmission Approach in Smart IoT Systems

1<sup>st</sup> Shreeram Hudda 

*SDN Lab, Department of CSIS*

*BITS-Pilani, Pilani Campus*

Jhunjhunu-333031, Rajasthan, India

hudda.shhudda@gmail.com

2<sup>nd</sup> K. Haribabu 

*SDN Lab, Department of CSIS*

*BITS-Pilani, Pilani Campus*

Jhunjhunu-333031, Rajasthan, India

khari@pilani.bits-pilani.ac.in

3<sup>rd</sup> Vikas Balani

*SDN Lab, Department of CSIS*

*BITS-Pilani, Pilani Campus*

Jhunjhunu-333031, Rajasthan, India

vikasbalani002@gmail.com

**Abstract**—Improving energy efficiency and maximizing network longevity are two pressing issues in the Internet of Things (IoT) and wireless sensor networks (WSN). Clustering aids in enhancing energy efficiency and extending network life. A cluster head is selected in each cluster to collect and aggregate data from its cluster members. While electing appropriate nodes as cluster heads is important, associating nodes with the elected cluster heads is another component that can aid improve the network's longevity. In this study, the authors proposed a new algorithm belonging to the family of local search problems for performing connection migration of nodes between different cluster heads. Furthermore, the simulation environment and the toolkit developed to evaluate several Cluster Head algorithms in this simulation environment have both been presented in detail.

**Index Terms**—SDN, IoT, Clustering, WSN, Connection Migration, K-Medoids, Cluster Head Selection.

## I. INTRODUCTION

Modern technology is making the world smarter over time. The development of new technologies has given rise to a plethora of new possibilities for human endeavor. The IoT is providing numerous advancements that improve the elegance of our homes. There are now many types of sensors that are all linked together to provide functionalities that improve our lives. As the name implies, IoT devices scour the physical world for information and relay that data to a central processing system, where it may be used to make decisions. However, there is no structure in place to organize the collection of data followed by any methodical and scientific procedures, therefore, IoT devices acquire data in a scattered way [1]–[6]. Software defined networking (SDN) is used with IoT applications to deal with vast amounts of scattered data by providing a centralized controller that manages and configures the network. It can regulate the data's dynamic behavior and modify operational procedures programmatically without affecting the physical architecture. Furthermore, the IoT network has some weaknesses that make it susceptible to cyber attacks of various kinds. For example, a denial-of-service (DoS) attack can temporarily disable the service. Therefore, it is a required issue that needs to be addressed in the IoT infrastructure. SDN is utilized to alleviate these challenges since it can readily detect anomalies and give main inhibition to attackers. Additionally, energy consumption is a key concern when millions of smart devices are connected

across an area where such a design is employed [7]. The major power consumption in IoT devices is for sending out all of the data that it has gathered. As most of the devices that the author are interested in are battery-powered, it is of much importance to regulate the amount of energy consumed by the whole network of devices. One of reducing power consumption is by grouping the devices into clusters, where one device in a cluster is designated as the cluster head responsible for sending data to the base station for processing [6]. Cluster head selection is one of the key factors in determining energy consumption. Many algorithms for selecting cluster heads based on various criteria have been proposed in the past [6]–[14]. Some of them have been discussed in the section II. Motivated by this premise, the authors offer an algorithm that takes energy efficiency into account while choosing cluster heads and assigning nodes to them. In each iteration, the algorithm seeks a new distribution that is as least as good as the present one.

### A. Contributions of this study

The contributions of this study area are as follows:

- To propose an energy aware algorithm for cluster head selection and distributing the nodes amongst the selected cluster heads.
- Implement the SDN network on the mininet-WiFi emulator
- Evaluate and compare the proposed model's performance to that of some related works on metrics such as response time, throughput, cluster head ratio, CPU utilization during denial of service attacks, etc.

The remainder of this study is formed as follows. The background and related works is discussed in section II. Section III contains a discussion on the energy model used for the simulation and a research gap. After that, section IV presents the proposed algorithm. Section V contains the details of the simulation environment used for implementing and testing the baselines. Section VI presents the toolkit developed for the purpose of comparing different baselines on various metrics. Section VII presents the results. The work done so far is concluded in section VIII and potential future work has been mentioned in section IX.

## II. RELATED WORKS

### A. LEACH and its variants

Distributing the network's energy load uniformly among its sensors, LEACH [1] is a self-organizing, adaptive clustering mechanism. In LEACH, the nodes are arranged in clusters, where one node from each cluster acts as a cluster head, responsible for collecting the data from the nodes that connect to it. The quantity of data being sent by a node determines how much energy the node uses. As the volume of data being transmitted through a node rises the nodes' energy usage will increase as well. It is thus evident that the nodes which are selected as cluster heads will consume more energy and die down faster. To prevent any one sensor's battery from running down, LEACH employs a system of random rotation for the high-energy cluster-head position. To further reduce energy dissipation and improve system longevity, the cluster heads aggregate the data from the cluster before sending it to the base station, effectively reducing the amount of data transported from the clusters to the base station. In a network of sensors, some fraction of the nodes at any given time will elect themselves to act as a cluster head with a certain probability. Each sensor node makes its own decision about which cluster to join by selecting the cluster head that is closest to it. After collecting information from all of the nodes in its cluster, the cluster-head node sends an aggregated version of that information to the base station. As was previously mentioned, a node's battery life decreases when it serves as the cluster's leader. To distribute this load more evenly over the network, the cluster heads are not fixed; a node that acts as a cluster head at time  $t$  may not act in this capacity at time  $t+d$ . This allows energy consumption to be evenly distributed between nodes, ensuring that all nodes die at a uniform, random rate. Every turn of the LEACH algorithm has two stages.

- 1) The first stage is the cluster head selection stage, a random number between 0 and 1 is allotted to each node in the network. Now for each node  $n$  if  $T(n) > R(n)$  then node  $n$  is elected as a cluster head.  $R(n)$  is the random number assigned to  $n$ . The value of threshold function is defined as -

$$T(n) = \begin{cases} \frac{P}{1-P \times r(\text{mod } \frac{1}{P})}, & \text{if } n \in G \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

Where  $P$  is the desired proportion of cluster heads,  $r$  is the number of rounds and  $G$  is the set of all nodes that have not been a cluster node in last  $\frac{1}{P}$  rounds

- 2) The second stage involves data transmission where normal sensor nodes transmit their data to the nearest cluster head.

LEACH-Centralized [2] is a variant of the LEACH algorithm where the cluster heads are determined at the base station. In the initialization phase of this algorithm, each sensor node communicates its energy level and geographical location to the base station. The base station then has the responsibility of selecting the cluster heads and spreading the energy load evenly amongst all nodes. The Base station does this by

calculating the average energy of all nodes and disqualifying any nodes in the current round whose energy is lower than the average energy  $E_{avg}$  (as shown in (2)). The base station employs a simulated annealing algorithm to solve the NP-hard problem of finding optimal clusters. In the simulated annealing problem, the objective function is the total of the squared distances between all of the nodes that aren't cluster heads and the cluster heads that are closest to them. Using this goal function, simulated annealing finds an appropriate choice for the cluster heads that minimizes the system's overall energy consumption. The average of energy is calculated as -

$$E_{avg} = \frac{1}{N} \times \sum_{i=1}^N E_i, \quad (2)$$

Because CHs are randomly selected in the LEACH and LEACH-C protocol, not all nodes have an equal chance of becoming CH, leading to an energy imbalance in the network. As a result, CHs with more cluster members can exhaust their energy early than others since the number of nodes in clusters is not uniform. In the protocols discussed above, nodes choose their clusters based on the strength of the signal they receive from CHs. Thus, information from nodes that can not connect to any cluster will not be able to go to the BS. This challenge is addressed by an approach proposed in study [3]. This approach also consists of two phases.

- 1) The first phase is the same as that of the LEACH algorithm but with a modified threshold function  $T_p(n)$ -

$$T_p(n) = \begin{cases} \frac{P}{1-P \times r(\text{mod } \frac{1}{P})} \times \frac{E_{rem}}{E_i}, & \text{if } n \in G \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

Where  $E_{rem}$  is the remaining energy at the  $i^{th}$  iteration and  $E_i$  is the initial energy of the node. The uneven load distribution across cluster heads is addressed by limiting the number of nodes that can connect to each cluster head. So each CH can relay the data from at most  $N_c$  nodes. Non-cluster head nodes try to connect to the nearest CH within a cutoff distance  $d_0$  given by -

$$d_0 = \sqrt{\frac{E_{fs}}{E_{mp}}} \quad (4)$$

Where,  $E_{fs}$  is the energy consumed by the amplifier in the free space, and  $E_{mp}$  in the multipath propagation. The nodes which remain unconnected with any cluster heads are called abandoned nodes. These abandoned nodes directly send their data to the BS.

- 2) The second phase is for data transmission where the CHs aggregate the data from the cluster members and relay the data to BS over an optimal path.

### B. Number of cluster heads

The authors of research work [4] proposed a Fuzzy decision-making-based energy-efficient scheme for WSNs. They estimated the optimal number of cluster heads required in a network for optimal energy consumption. The optimal

number of cluster heads  $k_{opt}$  is given as in (5). It was found that as nodes start to die in the network because of energy depletion, the number of cluster heads required for the network also decreases.

$$k_{opt} = \sqrt{\frac{N}{2 \times \pi}} \times \sqrt{\frac{\epsilon_{fs}}{\epsilon_{mp}}} \times \frac{M}{d_{toBS}^2} \quad (5)$$

Where  $N$  is the number of IoT devices,  $M$  is the area space, and  $d_{toBS}$  is the average distance between nodes and the sink. Energy usage increases globally if there are fewer than  $k_{opt}$  cluster heads because certain nodes must send data over long distances to reach the cluster head. More than  $k_{opt}$  cluster heads doesn't significantly reduce the distance nodes have to transmit to reach the nearest cluster-head, but it does increase the number of cluster heads that must transmit data over long distances to the base station, In this scenario, the data compression on cluster heads will not be large either.

### C. Opportunity Cost for Job Assignment in a Scalable Computing Cluster

The method proposed in study [5] ensures near-optimal overall system performance on every instance of job generation and resource availability. They achieve this with the help of state-of-the-art online algorithms that make no assumptions about the relationship between the past and the future, and function just in the present. In this paper's proposed method, the jobs are initially distributed at random to any available machine. Subsequently, after every specified amount of time, for each machine  $m$  in the cluster and for each job  $j$  assigned to the machine, the marginal cost associated with the job being assigned with machine  $m$  is calculated as  $c_{jm}$ . If the job's marginal cost on the current machine is higher than the job's marginal cost on another machine  $m'$  i.e.  $c_{jm} < c_{jm'}$  then the job  $j$  is migrated to machine  $m'$ . Any group of workstations will function better if their resources are utilized effectively. The processing capability of a cluster can be severely impaired by using an ineffective job assignment approach, which leads to extremely uneven loads and thrashing machines. By seamlessly shifting jobs from one machine to another, the cluster can better utilize its resources. In work [6], the authors proposed an approach based on distance and energy for cluster head selection. The distance determined between member nodes and base stations based on received signal strength. In work [7], the researchers introduced a cluster head selection method based on distance and energy. A flooding approach is used for calculating distance between member nodes and base stations. In work [8], the contributors proposed an approach to select a cluster head in a cluster based on distance and energy. In contrast to work [6], the work [8] computes the competition radius for each active cluster head based on distance to the base station and residual energy. This work calculates the competition radius using a fuzzy logic technique. In work [9], a distributed technique is proposed for uniform energy distribution between cluster heads. In this research work, three parameters (distance, node degree, and energy) are used for the cluster head selection method. In work [10], a

scheme is proposed based on distance, sink node location, and residual energy by combining the unequal clustering algorithm with multi hop routing. In work [11], a fuzzy logic based approach is proposed. The three input variables for fuzzy logic are: (a) node degree, (b) distance to the base station, and (c) residual energy. A research study [12] is a method for balancing energy based on network partition and distance. This study first determines the distance between the base station and member nodes using the received signal strength. The three parameters (distance, node degree, and energy) used to select a cluster head. A distributed clustering technique [13] proposed to overcome the energy hole problem. The cluster head selection was performed based on the distance, energy, sink node location, and node degree. The researchers proposed a scheme [14] that was proposed by combining hierarchical clustering with MAC. In this scheme cluster head selection was also performed based on the distance, energy, sink node location, and node degree. In this present study, the BS decides the CH based on the information obtained. The remaining nodes (i.e. non cluster head nodes) then join the closest CH to form a cluster. So in comparison to work [5], the CHs in present work can be viewed as machines, and remaining nodes can be viewed as jobs that are being joined to a CH.

## III. DISCUSSIONS

### A. Energy Consumption

In this paper, a basic model is adopted for the energy dissipated by the sensor nodes. The transmitter and the receiver dissipate energy to run the radio electronics, and in addition, the transmitter also dissipates power for operating the power amplifier. Depending on the distance between two nodes, either the free space or the multipath fading channel models are used. If the distance between the nodes is less than  $d_0$  then the free space model is used where the power loss is proportional to  $d^2$  otherwise the multipath model where the power loss is proportional to  $d^4$  is used. Formally, the energy consumption by the nodes for transmitting a  $l$  bit message is given as -

$$E_{Tx}(l, d) = E_{Tx-elec}(l) + E_{Tx-amp}(l, d) = \begin{cases} l \times E_{elec} + l\epsilon_{fs}d^2, & d < d_0 \\ l \times E_{elec} + l\epsilon_{mp}d^4, & d \geq d_0 \end{cases} \quad (6)$$

The energy dissipated by the receiver is given as -

$$E_{Rx} = l \times E_{elec} \quad (7)$$

### B. Assigning nodes to cluster heads

All the methods covered so far revolve around picking a few nodes to serve as the cluster-head of respective clusters. Selecting the cluster head for a node with the goal of reducing global energy consumption receives little to no consideration. All the algorithms talked about so far connect non-cluster heads to the nearest cluster head to conserve as much energy as possible. In this study, the authors suggest an algorithm that tries to reduce the amount of energy used by a network as a whole by taking into account the different permutations and

combinations of nodes that can be assigned to the same cluster. This proposed technique belongs to the family of local search algorithms, which offer a reasonable solution to NP problems.

#### IV. PROPOSED WORK

The proposed algorithm begins with the use of k-medoids to find the cluster heads initially. Provided with the locations of all the sensor nodes, BS decides the nodes which should act as the cluster head so that the total sum of distances from sensor nodes to their corresponding cluster heads is minimized. The Algorithm 1, and Algorithm 2 contain the pseudocodes for the k-medoids and the proposed approach respectively. In this present work, the proposed approach can be divided in two phases - (a) Initialization phase, and (b) Cluster node migration phase.

Algorithm 1: K-MEDDOIDS

**Input:** NODES, K  
**Output:** ClusterHeads

- 1: Initialize  $k$  medoids as  $m_1, m_2, \dots, m_K$
- 2: Initialize  $k$  clusters as  $c_1, c_2, \dots, c_K$
- 3: **while**  $i = 1$  to  $MAX\_ITER$  **do**
- 4:   PREV\_COST = Sum of dissimilarities between medoids and their corresponding cluster members
- 5:   **for**  $j = 1$  to  $n$  **do**
- 6:      $x = \text{node} \in c_i$  with least sum distances from all other nodes in  $c_i$
- 7:      $m'_i = x$
- 8:     NEW\_COST = Sum of dissimilarities between new medoids and their corresponding cluster members.
- 9:     **if** NEW\_COST  $\geq$  PREV\_COST **then**
- 10:       break
- 11:     **end if**
- 12:     assign each  $m_i = m'_i$
- 13:   **end for**
- 14: **end while**
- 15: **return** Array of nodes corresponding to  $m_1, \dots, m_k$

#### V. SIMULATIONS

The simulations are carried out using Mininet-WiFi and a single Ryu Controller.

##### A. Topology and Routing

The topology consists of sensors being randomly distributed over a 250x250 meter area, with the Base Station positioned at (25,125). The sensor nodes should have the capability to receive and transmit messages, redirect their traffic to any other specified node (including BS), and act as a cluster head for aggregating and relaying the data of other nodes to the base station. To achieve this functionality each sensor node is simulated as a combination of a host and a switch. To enable communication between any two nodes in the topology, a link is constructed between every pair of switches. At the start of the simulation, the controller is informed of the locations of all the nodes and there are no flow rules inserted in any of

Algorithm 2: Proposed Algorithm

**Input:** NODES

**Output:** NULL

*Initialisation Phase :*

- 1: Initialize  $k_{opt} = \sqrt{\frac{N}{2 \times \pi}} \times \sqrt{\frac{\epsilon_{fs}}{\epsilon_{mp}}} \times \frac{M}{d_{toBS}^2}$
- 2: clusterHeads = K-MEDDOIDS(NODES.location,  $k_{opt}$ )
- 3: **for** node  $n \in$  NODES and  $n \notin$  ClusterHeads **do**
- 4:   closestCH = CH closest to  $n$
- 5:   connect( $n$ , closestCH)
- 6: **end for**

*Cluster-node Migration Phase*

- 7: **for** after every T seconds **do**
- 8:   **for** each CH  $\in$  clusterHeads **do**
- 9:     **if**  $E_{rem}(CH) < E_{th}$  **then**
- 10:       closestNode = closest non cluster head node with  $E_{rem} > E_{th}$
- 11:       add closestNode to clusterHeads
- 12:       migrate all connections from CH to closestNode
- 13:       remove CH from clusterHeads
- 14:     **end if**
- 15:     numNodesCH = number of nodes connected to CH
- 16:     **for** each node  $n$  connected to CH **do**
- 17:       marginalCost = energyCH(CH, aggregated(CH, numNodesCH,  $q$ )) - energyCH(CH, aggregated(CH, numNodesCH-1,  $q$ )) + energy\_node(CH,  $n$ ,  $q$ )
- 18:       currBestCH = CH
- 19:       diff = 0
- 20:       **for** each CH'  $\in$  clusterHeads and CH'  $\neq$  CH **do**
- 21:         marginalCost' = energy(CH', aggregated(CH', numNodesCH'+1,  $q$ )) - energyCH(CH', aggregated(CH, numNodesCH'+1,  $q$ )) + energy\_node(CH',  $n$ ,  $q$ )
- 22:         **if** diff > marginalCost - marginalCost' **then**
- 23:         diff = marginalCost - marginalCost'
- 24:         currBestCH = CH'
- 25:         **end if**
- 26:       **end for**
- 27:       **if** currBestCH  $\neq$  CH **then**
- 28:         disconnect( $n$ , CH)
- 29:         connect( $n$ , CH')
- 30:       **end if**
- 31:     **end for**
- 32: **end for**
- 33: **end for**

the switches other than to send a *packet-in* request to the controller. Hence, it is the job of the controller to program the switch to forward the information of all the nodes to the base station via appropriate cluster heads, and to allow packets to flow back from the base station to the nodes. This functionality is achieved by inserting flow rules into switches. At each *packet-in* request made to a controller, the controller finds out the switch to which the sensor should be forwarding



its data. The controller then provides a flow rule to correctly redirect the packets received at the switch. Furthermore, to allow back-flow of packets from the base station to sensor nodes, the controller installs flow rules that instruct the switch to forward the packets coming in with the destination MAC address as the current packet's source address to the sender of the current packet. The base station node consists of a host  $sta\_bs$  and a switch  $ap\_0$ , all the non-base station nodes are simulated using a combination of  $sta\_i$  and  $ap\_i$  connected with a link such that port 0 of  $sta\_i$  is connected to port  $i + 1$  of  $ap\_i$ . Furthermore, to simplify the insertion of flow rules switches  $ap\_i$  and  $ap\_j$  are connected such that port  $j + 1$  of  $ap\_i$  and port  $i + 1$  of  $ap\_j$  have a link in between.

### B. LEACH and LEACH-Improved

LEACH is implemented as a baseline for this proposed algorithm in the above described simulation environment. The controller maintains the state information for each of the nodes. The flow rules are inserted with a 15 seconds timeout, so the controller chooses new cluster heads after every 15 seconds. This is done by using  $T(n)$  (which explained earlier in (1) in section II-A). It might be possible that after one iteration of  $T(n)$ , there is no cluster heads with probability  $1 - P$ . If such a scenario arises then the authors keep repeating  $T(n)$  until end up with at least one cluster head. The nodes chosen as the cluster heads are injected with a flow rule to forward any packet addressed for the BS directly to the BS and to forward the packets coming from the BS to the appropriate cluster member node. This reverse flow from BS to sensor nodes is established by maintaining a dynamic mapping between Host MAC addresses (MA) and the port number on which the packet with source MAC as MA was received. It is possible that the controller receives a *packet - in* request while it is finding the set of new cluster heads for the next iteration, such requests are handled by instructing the switch to redirect the packet according to previous most recent cluster formations. The controller also keeps track of the energy used by each node. The energy calculations are done as per (6). The controller reads the `\proc\net\dev` file to find the number of bits transmitted by any switch. The information about the distance over which these bits were transmitted is found by using the topology information available to the controller. If the energy consumed by a sensor exceeds a pre-defined threshold energy level then that node is declared dead, and it is not used in further calculations for finding the cluster heads. Another thing that needed consideration was accounting for the aggregation that is done at each cluster head, this was done by finding the number of bits that would have been transmitted if aggregation was taking place, by using the same concept as aggregated function described in section V. The improved version of LEACH is implemented in the same manner, but by making use of the remaining energy of a sensor node in finding the cluster head. The cluster head selection algorithm is governed by (2).

### C. Proposed algorithm

In this section, the authors discuss certain implementation aspects of the proposed algorithm. The aspects related to the death of sensor nodes, energy calculations, and handling *packet - in* requests at cluster head calculation time are handled in the same way as described in the subsections V-A, and V-B.

- aggregated(CH, numNodes, q) - This function returns the expected data sending rate of cluster head  $CH$  when it has  $numNodes$  many active connections where data receiving rate on each connection is  $q$  bits/sec. The implementation of this function requires the use of basis functions to model a mathematical relation between the number of active connections, the *data-receiving-rate* on each connection, and the aggregated *data-sending-rate* using actual sensor data.
- energyCH(CH, data-sending-rate) - This function uses (6) for finding the power required for sending data from  $CH$  to BS at *data - sending - rate*.
- energy\_node(CH, n, data-sending-rate) - This function also uses (6) for finding the power required for sending data from node  $n$  to cluster head  $CH$  at *data-sending-rate*.

## VI. EXPERIMENTS

A toolset was created to calculate key metrics for each simulated algorithm. Below, the authors will detail the approach used to record each metric.

### A. Response Time vs Number of Transaction Requests

The *ping* command of the Mininet framework is used to measure how long it takes for the BS to respond to a query from the node. The *ping* command is run on different nodes for a number of iterations by running a *bash* script. Then, the average response time is calculated. This script runs when *Ostinatotrafficgenerator* sends different amounts of traffic to the BS. Then, the average response time against the number of transaction requests is plotted.

### B. CPU Utilization

Since switches and controllers are just processes that mininet starts, the authors track how much CPU these processes use to get an idea of how much CPU the algorithm would use in a real IoT environment. A Denial of Service (DoS) attack is simulated against the BS for a couple of seconds and then plot the CPU usage during this attack.

### C. Throughput vs Number of nodes

The throughput is calculated by counting how many bits are sent over the  $ap\_0 - sta\_bs$  link per second. The BS runs a *bash* script that starts  $N$  *iperf* servers. Another *bash* script starts  $N$  *iperf* clients, one on each node. These *iperf* clients connect to the *iperf* server on the BS and send uniform traffic to the server. The throughput is then recorded by using either the *Wireshark* I/O graphs or the number of bits sent over  $ap\_0 - sta\_bs$  link by reading the statistics from the `\proc\net\dev` file every second.

#### D. Energy Consumption

Energy consumption calculation is done by the controller as it has information about all the active connections and routes. To find the energy consumption by node  $i$  the number of bits transmitted over the outlink of switch  $ap\_i$  is recorded using the `\proc\net\dev` file. The distance over which the node transmitted the data is found using the topology and routing information available to the controller. Then using (6) energy consumption by the node is calculated. Once the energy consumed by a node exceeds a threshold value, the node is declared dead, and the authors do not perform energy calculations for these nodes afterward. To carry out this experiment, a *UDP* server is started on the base station that listens for all incoming *UDP* traffic on a port. Then a *UDP* client is started on each of the client nodes to send traffic to the server at  $2Kb/s$ .

#### VII. RESULTS

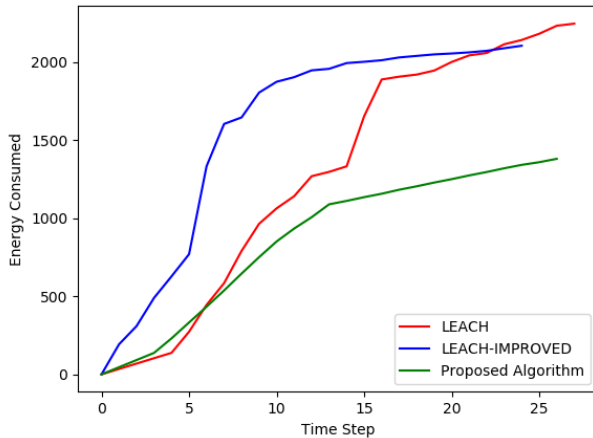


Fig. 1: Comparison of Proposed Algorithm with LEACH and LEACH-Improved

In this section, the authors showcase the energy consumed in a 10-nodes topology. The threshold energy for each of the nodes is set to 250 units. Therefore, the total energy in the system is 2500 units. Figure 1 illustrates the energy consumed by each of the implemented algorithms over a period of 25 time steps. Each time step being of 15 seconds. It can be noticed that the proposed algorithm does a far better job than LEACH and LEACH-Improved algorithms at reducing the global energy consumption in the system.

#### VIII. CONCLUSIONS

Using the Mininet-WiFi framework and the Ryu Controller, the authors have investigated several different Cluster Head selection algorithms. A novel technique is proposed with the goal of minimizing the total amount of energy used by a network. This algorithm is different from others in that it not only selects a cluster head but also attempts to optimize energy usage by moving connections from one cluster head to

another cluster head. In addition, a set of tools has been built to compare the performance of existing algorithms to the one proposed here, taking into account factors such as response time, throughput, energy consumption, etc.

#### IX. FUTURE WORK

The future work involves finding the various metrics associated with the cluster-head selection algorithms by using the developed toolkit on a high-performance cluster. To showcase the capability of implemented algorithm, a large number of sensor nodes (at least 100 plus nodes) need to be simulated. This necessitates the need for having a machine with high I/O capabilities and a fast CPU clock, to handle the processes associated with each of these simulated nodes.

#### REFERENCES

- [1] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the 33rd annual Hawaii international conference on system sciences*, pp. 10–pp, IEEE, 2000.
- [2] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on wireless communications*, vol. 1, no. 4, pp. 660–670, 2002.
- [3] I. Daanoune, A. Baghdad, and A. Ballouk, "Improved leach protocol for increasing the lifetime of wsns," *Int. J. Electr. Comput. Eng. IJECE*, vol. 11, pp. 3106–3113, 2021.
- [4] P. Azad and V. Sharma, "Cluster head selection in wireless sensor networks under fuzzy environment," *International Scholarly Research Notices*, vol. 2013, 2013.
- [5] Y. Amir, B. Awerbuch, A. Barak, R. S. Borgstrom, and A. Keren, "An opportunity cost approach for job assignment in a scalable computing cluster," *IEEE Transactions on parallel and distributed Systems*, vol. 11, no. 7, pp. 760–768, 2000.
- [6] C. Li, M. Ye, G. Chen, and J. Wu, "An energy-efficient unequal clustering mechanism for wireless sensor networks," in *IEEE International Conference on Mobile Adhoc and Sensor Systems Conference, 2005.*, pp. 8–pp, IEEE, 2005.
- [7] P. Liu, T.-I. Huang, X.-y. Zhou, and G.-x. Wu, "An improved energy efficient unequal clustering algorithm of wireless sensor network," in *2010 International Conference on Intelligent Computing and Integrated Systems*, pp. 930–933, IEEE, 2010.
- [8] H. Bagci and A. Yazici, "An energy aware fuzzy unequal clustering algorithm for wireless sensor networks," in *International conference on fuzzy systems*, pp. 1–8, IEEE, 2010.
- [9] B. Baranidharan and B. Santhi, "Ducf: Distributed load balancing unequal clustering in wireless sensor networks using fuzzy approach," *Applied Soft Computing*, vol. 40, pp. 495–506, 2016.
- [10] R. Zhang, L. Ju, Z. Jia, and X. Li, "Energy efficient routing algorithm for wsns via unequal clustering," in *2012 IEEE 14th International Conference on High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems*, pp. 1226–1231, IEEE, 2012.
- [11] R. Logambigai and A. Kannan, "Fuzzy logic based unequal clustering for wireless sensor networks," *Wireless Networks*, vol. 22, pp. 945–957, 2016.
- [12] D.-g. Zhang, S. Liu, T. Zhang, and Z. Liang, "Novel unequal clustering routing protocol considering energy balancing based on network partition & distance for mobile education," *Journal of Network and Computer Applications*, vol. 88, pp. 1–9, 2017.
- [13] J. Hamidzadeh and M. H. Ghomanjani, "An unequal cluster-radius approach based on node density in clustering for wireless sensor networks," *Wireless Personal Communications*, vol. 101, no. 3, pp. 1619–1637, 2018.
- [14] S. Gajjar, M. Sarkar, and K. Dasgupta, "Famacrow: Fuzzy and ant colony optimization based combined mac, routing, and unequal clustering cross-layer protocol for wireless sensor networks," *Applied Soft Computing*, vol. 43, pp. 235–247, 2016.