A Location-Unaware Distributed Clustering Algorithm for Mobile Wireless Networks Using Fuzzy Logic

Shahram Nourizadeh, Ye Qiong Song, Jean-Pierre Thomesse

A Location-Unaware Distributed Clustering Algorithm for Mobile Wireless Networks Using Fuzzy Logic

Shahram Nourizadeh, Y.Q. Song, J.P. Thomesse

LORIA research laboratory - Nancy University, France

Emails: {Shahram.Nourizadeh, Song, Thomesse}@loria.fr

Abstract—this paper proposes a decentralized algorithm to organize an ad hoc sensor network into clusters by using Fuzzy Logic. Each sensor uses a Fuzzy decision making process to find the best Cluster Head. Simulation shows that this protocol is able to dynamically adapt to network mobility and also shows that with fuzzy logic we have stable clusters and so a cluster head have greater lifetime. Therefore respect to mobility, we have the minimum message exchange and so minimum energy consumption.

Keywords: Wireless networks, dynamic clustering, distributed algorithm, Fuzzy Logic

I. INTRODUCTION

An ad hoc network is comprised of wireless nodes and requires no fixed infrastructure. Any device with a microprocessor, mobile or stationary, is a potential node in an ad hoc network. An example of ad hoc networks is Tele-surveillance networks. In the field of medical, Tele-surveillance the aged peoples is an important challenges of the researchers. Patients at a disability or the problems of Alzheimer scene can greatly benefit from technologies that continuously track their locations until they are admitted to the hospital or at home. Because of the mobility of the persons we need an algorithm that is adaptable with the changes of network topology when patients moving. A typical network architecture could be like figure 1.

In this architecture the sensors attached to the patients communicate with the central computer via communication backbone. The Communication backbone consists of the mobile or fixed sensors and also the sensors attached to doctors and nurses. Therefore in this architecture we need a routing protocol that is adaptive with mobility of the sensors.
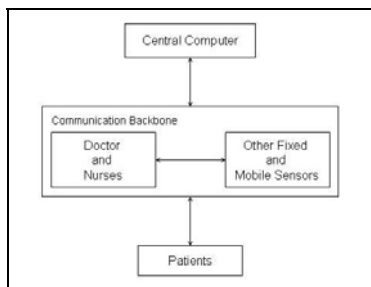

Figure 1. A typical network architecture

Dynamic cluster-based routing is one of existing techniques for routing in ad hoc networks. In dynamic cluster-based routing, the network is dynamically organized into partitions called clusters, with the objective of maintaining a relatively stable effective topology [1]. Clustering is advantageous because it:

- conserves limited energy resources and improve energy efficiency,

- aggregates information from individual sensors and abstract the characteristics of network topology,

- provides scalability and robustness for the network.

- provides ability to mobile sensors localisation.

In addition, the membership in each cluster changes over time in response to node mobility and so it can be used to mobility management in mobile networks.

The remainder of this paper is structured as follows: section II discusses related work. Our algorithm, its uses in mobile networks and its decision making, are described in section III, while section IV describes its functionality with some message exchange diagrams. Our simulations are resented in section V and section VI provides concluding remarks and outlines directions for future research.

II. RELATED WORKS

Several dynamic clustering strategies have been proposed in the literature. The zone routing protocol (ZRP) proposed by Haas and Pearlman [2] is a hybrid strategy that attempts to balance the trade-off between proactive and reactive routing. The objective of ZRP is to maintain proactive routing within a zone and to use a query–response mechanism to achieve inter-zone routing. In ZRP, each node maintains its own hop-count constrained routing zone; consequently, zones do not reflect a quantitative measure of stability, and the zone topology overlaps arbitrarily.

LEACH [3] is an application-specific data dissemination protocol that uses clustering to prolong the network lifetime. LEACH clustering terminates in a constant number of iterations (like HEED [4]), but it does not guarantee good cluster head distribution and assumes uniform energy consumption for cluster heads. In contrast, HEED makes no assumptions on energy consumption and selects well-

distributed cluster heads but HEED assumes quasi-stationary nodes.

In [5], a fuzzy logic approach to cluster-head election is proposed based on three descriptors - energy, concentration and centrality. In this approach the cluster-heads are elected by the base station in each round by calculating the chance each node has to become the cluster-head by considering three fuzzy descriptors – node concentration, energy level in each node and its centrality with respect to the entire cluster. This technique is proposed to use in LEACH [3], but it can not support the mobility of the node and in addition it is centralized algorithm and therefore it cannot be scalable.

In fixed infrastructure networks, hierarchical aggregation achieves the effect of making a large network appear much smaller from the perspective of the routing algorithm. Cluster-based routing in ad hoc networks can also make a large network appear smaller, but more importantly; it can make a highly dynamic topology appear much less dynamic. Unlike the cluster organization of a fixed network, the organization of an ad hoc network cannot be achieved offline. The assignment of mobile nodes to clusters must be a dynamic process wherein the nodes are self-organizing and adaptable with respect to node mobility. Consequently, it is necessary to design an algorithm that dynamically implements the self-organizing procedures in addition to defining the criteria for building clusters.

III. PROPOSITION

A. Why a new approach?

In a network with mobile nodes and dynamic topology, we need to manage the mobility of the nodes. As we described in section I, one of the suitable solutions is the clustering. In a network with mobile nodes we need a dynamic clustering algorithm that can support the mobility and also can manage the power level of the nodes.

As we saw in section II, in the existing algorithms, the authors assumed many conditions that are far to reality. Some algorithms are based on centralized control that makes them not to be scalable. In some algorithms we have periodic rounds and in each round the cluster head will be elected and will be fixed for one round, but in a dynamic network with mobile nodes we can not have a fixed cluster node for a period, because the location of the nodes may be change in each period and in this case we need to elect another cluster head, therefore this type of algorithms will be good for networks with low mobility nodes. To solve these problems we proposed a new approach that can manage the mobility of the nodes without assuming any condition that is far to reality. The algorithm is totally distributed and this will make the algorithm, scalable, energy efficient.

B. Some definitions

Before talking about our approach, we must take a look at some definition used in this report:

- We will use: A→B to show the communication ability from node A to B.

- {ALL}: set of all nodes of network.

- Base-Station (BS): Central computer to monitoring the network or to gathering the information. (i.e. BS in Figure 2)

- Zone-Head (ZH): Node (mobile or stationary) that can communicate directly with BS (i.e. A, B and C in Figure 2):

$$\forall n \in \{ALL\} \mid n \to BS \Rightarrow n \text{ is a ZH}$$

- {ZH}: set of all ZHs of network.

- Cluster-Head (CH): Node (mobile or stationary) that can communicate with one or more ZH or a node with some children that can communicate with other CH (i.e. E and G in Figure 2) :

$$\forall n \in \{ALL\}, \exists\, m, z \in \{ALL\} \mid n \to m \wedge z \to n \Rightarrow n \text{ is a CH}$$

- {CH}: set of all CHs of network.

- Zone: set of one or more cluster. Each ZH constructs a Zone. In Figure 2, {A, D} is a Zone, also {B, E, F, H} and {C, G, I}.

- Leaf-Node (LN): a node without child (i.e. D, H, I and F in Figure 2):

$$\forall n \in \{ALL\}, \nexists\, m \in \{ALL\}\text{-}n \mid m \to n \Rightarrow n \text{ is a LN}$$

- Cluster-Member (CM): Node (mobile or stationary) that can communicate with one or more Cluster-Heads.

- {CM}: the set of CMs of a CH.

- Data: the information which a node wants to send to the other node.

- {Unknown}: the set of the nodes with unknown situation.

- Level: Level of a node is the number of hops between the node and BS. In Figure 2, A, B and C are in Level 1, nodes D, E, F and G, in Level 2 and nodes H and I in Level 3.

- {myNeighbours}: Neighbours of a node.

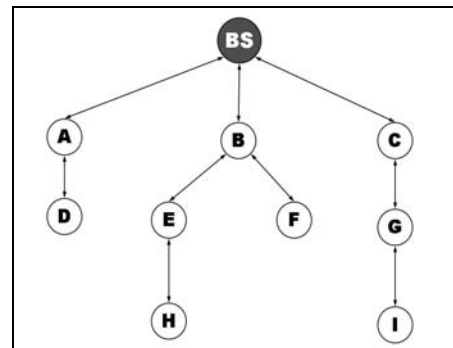- {myChild}: Nodes that joined a specific node (ZH, CH).

Figure 2. Topology example

- Zone-Information (ZI): all of the information of a specific zone, i.e. number of CHs in the Zone.

- Cluster-Information (CI): all of the information of a specific cluster, i.e. number of CMs of the cluster.

- Mobility: This is a parameter to evaluate the change of a node. You can find the complete description of this parameter in III.E.

- Quality of Link (QoL): this is a parameter that shows the Reliability of the connection between a node and his parent. (See III.E).

- Load: Load is sum of the QoL between a node and his children. The condition to accept a new Child in node n is :

$$(n.Load + NewChild.QoL)/ n.numberLowLevelNodes + 1 \le n.QoL$$

  In this formula the number of low level nodes means the number of all the nodes that are connected to this node directly or indirectly. We named this condition "*admission condition*". Load is a new definition that can be used as a new parameter in QoS. Figure 3 shows a sample load distribution tree in part of a network. The weight of each connection is QoL.

- In this algorithm, we have a successful clustering if and only if:

$$\forall\, n \in \{ALL\},\ \exists\, m \in \{ALL\}\text{-}n \mid n \to m$$

  and also:

$$\forall\, n,m \in \{ALL\} : (n \to m) \Rightarrow m \in \{ZH\} \cup \{CH\}$$

- ZH_Advertisement: this message sends with a ZH to invite the nodes. The node that can receive this message is a potential CH.

- CH_Advertisement: this message sends with a CH to invite the nodes to elect it as a CH.

- Join: this message use by CH to answer the ZH_Advertisement, and also with N to answer the CH_Advertisement.

- Quit: message that send to a CH or a ZH, for advertising of membership annulations.

- Join-Other: That a message that a ZH or a CH sends to one of its child to request find another parent. This will be when the ZH or CH:

  i) is in a low level energy state,

  ii) received a new request of join from a node with no other possible parent, and the admission condition is not satisfied. Therefore it must reduce its load by reducing number of its child.

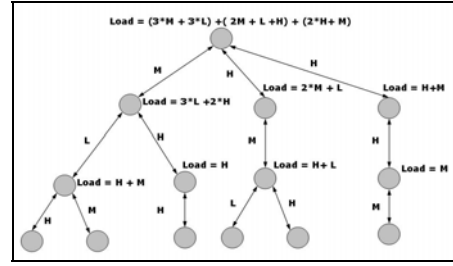- Hello: it is a message, between the nodes to exchange the information.
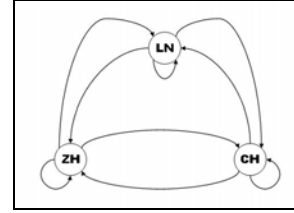

Figure 3.   A load distribution tree

Figure 4.   Node state diagram

NOTES:

- Figure 4, shows the state diagram of a node during its lifetime, in our approach. As it shows a node has three possible states: LN, CH and ZH.

- The ZHs and CHs, in the critical state, i.e. low energy level, will send the Join_Other message to their members to join other node.

- In the ZH_Advertisement message, CH_Advertisement and Hello messages, the nodes sends all of the information needed to make a correct decision, like: energy level, mobility frequency, CH id, Zone Head id…

- In all type of the nodes (ZHs, CHs), by receiving Hello, ZH_Advertisement or CH_Advertisement, the Nodes' Id will be added to myNeighbours set with a priority. That means, in the top of set we have the ZHs, then CH, and then the LNs, and in each category, on the top we have the best one, i.e. in the CHs category first we have the CH with highest chance, and after that the others with lower chance than the first.

C. Algorithme description

There are two questions that the algorithm must be able to answer:

- How can manage the mobility of the nodes?

- When a node has more than one potential parent, how it can choose the best one?

And so our approach has two main parts:

  i) Mobility management

  ii) Decision making

The two main parts of the algorithm will answer these questions. In next part we will describe the first part and the second part will be described in IV.B.

D. Mobility management

This section will describe the clustering algorithm, separately by node's function.

1) Zone Head

In Table I, you can see the action of a ZH by receiving Join message for a node (*n*). First, the ZH will check if *n* is in the {Unknown} or not, if *n* is in the set, it will remove *n* from the set and if *n* is not his neighbour, *n* will be added into {myNeighbours}. Then the ZH will check its capacity by verifying *admission condition*: If OK, it will accept *n* as a new member; If KO, the ZH will choose one of his actual members that is connected with a smaller QoL than *n* and will send a request the join another one and will wait to a response:

- If the answer is OK then the ZH will add the new node to the set of CHs and will drop the other member.

- If the answer is KO, the ZH will be send Join_Other to the new node, and will wait to have a response.

2) Cluster Head

Like in the ZHs, in CHs we have same clustering process when the CH receives Join from another node. We have not a different between two cases. Just in the last sentence we have a small change, we must replace it with:

Send (*this.CH, CI)*;

```
Receive (Join from n)
 accepted = False;

 If  n∈ {Unknown} then {Unknown}={Unknown} – n;

 If  n∉{myNeighbours} then
        {myNeighbours}={myNeighbours}+ n;
 If (admission condition) then  accepted = true;
 Else // admission condition not satisfied

    For  c ∈ {myChild} |
               (c.QoL is Min) & (n.QoL > c.QoL)
       Send (c,  Join_Other );
       Wait (receive (msg, c));
       If (OK or Timeout)
          accepted=true;
          {myChild}={myChild}-c;
          If (Timeout)
              {Unknown}={Unknown} – c;
              {myNeighbours}={myNeighbours}- c;
 If ( accepted == true)
      {myChild}={myChild}+ n;
       Send (n, OK);
 else
      Send (n,  Join_Other );
      Wait (receive (msg, n));
      If (KO) // n has not other connection possibility
         Send (n, OK);
         {CH}= {CH} + n;
 Send (BS, ZI);
```

TABLE I.   RECEIVE JOIN IN A ZONEHEAD

3) Node

This section describes the procedures of clustering in a node. In Table II, when a node revives *Hello* from c, it will add the node in its {myNeighbours} set and then if it has not a CH, it will send a Join message to its neighbour. As shown in Table III, by receiving a ZH_Advertisement, the Node will send a Join message to the ZH and if it has a CH or a CN, it will also send a Quit message for that. Table IV shows that in a Node, by receiving Join_Other message, the Node will send Join message to its neighbours and if it receives OK, from one of them, it will send OK to its parent node, if no, it will send KO.

```
Receive (CH_Advertisement from c)
    If (this.CH==null)
        Send (c, Join)
        Wait (receive (OK from c);

        For all n ∈ {myNeighbours}
             Send (n, Hello);
    Else // this.CH ≠ null
        Best = Find-the-best (this.CH, c);
        If (Best == c)
            Send (c, Join);
            Wait (receive (OK from c)
        For all n є {myNeighbours}
            Send (n, Hello);
```

TABLE II.   RECEIVE *CH_ADVERTISEMENT* IN A NODE

```
Receive (Hello, n)
    {myNeighbours} = {myNeighbours} + n;
    If (this.CH==null and n.CH ≠ null)
        Send (r, Join);
        Wait (receive (OK from n)
        this.CH = n ;

        For all m ∈ {myNeighbours} - n
            Send (m, Hello);
```

TABLE III.   RECEIVE *HELLO* IN A NODE

```
Receive (CH_Advertisement from r)
Send (r, Join);
Wait (receive (OK from r)
If (this.CH ≠ null)
  Send (this.CH, quit)
Send (all, CH_Advertisement);
```

TABLE IV.   RECEIVE *CH_ADVERTISEMENT* IN A NODE

```
Receive (Join-Other from n)
    Found=false;
    For all m ∈ {myNeighbours}
        Send (m, Join);
        Wait (receive (OK from m)
        If (OK)
            Found=True;
            Break;
    If (Found = true) then
        Send(n, OK)

        For all c ∈ {myNeighbours} - m
             Send (c, Hello);
    Else
        Send(n, KO)
```

TABLE V.   RECEIVE *JOIN_OTHER* IN A NODE

Table V shows that in a Node, by receiving Join-Other message, the Node will send Join message to its neighbours and if it receives OK, from one of them, it will send OK to its parent node, if no, it will send KO.

E. Decision making

As we said in section 3, when a node receives more than one advertisement from CHs or ZHs, it will run a function to choose the best one and this function may use any logic to make the decision. In our algorithm we use Fuzzy logic but other technique or logic can be used. You may ask why Fuzzy Logic?

Fuzzy logic control is capable of making real time decisions, even with incomplete information. Conventional control systems rely on an accurate representation of the environment, which generally does not exist in reality. Fuzzy logic systems, which can manipulate the linguistic rules in a natural way, are hence suitable in this respect. Moreover it can be used for context by blending different parameters - rules combined together to produce the suitable result.

In [5], a fuzzy logic approach to cluster-head election is proposed based on three descriptors - energy, concentration and centrality. They improved LEACH routing algorithm [3] by using Fuzzy Logic and the cluster-heads are elected by the base station in each round by calculating the chance each node has to become the cluster-head by considering three fuzzy descriptors – node concentration(Number of neighbours), energy level in each node and its centrality with respect to the entire cluster.

As we said, the proposed algorithm in [5] is centralized and therefore is not scalable and also cannot support the mobility of the nodes. But, which descriptors we can choose in a mobile network? In a dynamic network with mobile nodes, can we use concentration and centrality as factors to decision making in a distributed approach?

The answer is that in a distributed approach each node chooses its parent by processing its local information therefore centrality of a node cannot be suitable to its decision, this parameter will be good for a centralized algorithm to reduce the routing hops between CH and nodes in the same cluster, but a node can not have a general view of network and thus can not evaluate this parameter for a CH. In addition, in a dynamic network with mobile nodes the number of neighbours is not a good descriptor, because location of the node is not fixed and we cannot be sure about number of neighbours in each time. Thus we must find the best parameter to evaluate the candidates. In addition we need a parameter to evaluate the movement rate of each node to choose a node with minimum movement rate as parent node. But in a distributed approach we cannot evaluate exactly this parameter without a location detection system like GPS, but in a system without GPS antenna how can we detect the movement of a node, assume two nodes A and B, and assume that distance AB is time t is X, and in time t+1 the distance is X+Y. with this information we cannot say which node is moved? A, B or both of them?

Without a lot of calculations or a GPS system, we cannot answer this question. Because of energy consummation, we cannot have a lot of computation process in each node to find the moved node and also because of cost of materiel of GPS system we don't want to add this system in our network; in addition the performance of GPS in an Indoor system is not interesting. What we can do?

To solve this problem we proposed a new parameter named Mobility. This parameter shows frequency of parent, level or zone change of a node. (Number of CH or level change of a node in his life time). Therefore each time that the node changes his CH or his level, it must increment value of a variable named *Change* and divide it to his lifetime to find the Mobility. It is clear that the mobility of a fix node can be greater than zero, because of the mobility of his parent.

To have a complete evaluation of the reliability of the connection, we must also determine the quality of communication between the node and the candidates to be his parents. As we said, in a mobile network we cannot use distance between the nodes because it is not fix, but we can use the quality of received signal to evaluate the quality of communication.

In each ZH or CH advertisement the node will send necessary information to be evaluated by the other nodes, as like as: his *Energy level*, his *Mobility* and his *QoL*, and the node will compute the QoL of the connection between candidate and itself. The QoL of a node is *Reliability* parameter that he was calculated for his parent. This parameter helps us to choose the best parent node, a node with maximum energy, maximum stability, and higher reliability of connection.

We use four parameters: *Energy level* of the node (Battery charge), *Mobility*, *Quality of Link - QoL* (*Reliability* between a node and his parent) and the *Quality of Received Signal – QoRS* (Received Strength Signal Indicator).

These parameters will be the *Fuzzy Logic Descriptors* and each of them has three possible values: *low, medium, high*. Therefore we have 81 rules to evaluate a node. The result of the rules will be *Reliability* with five possible levels: *Very Low, Low, Medium, High and Very High*. Table VI shows a part of fuzzy rules. By finding the Reliability of a candidate we must evaluate the chance of the candidate to be a parent. To do this, we chose a new parameter, named Level. Level presents the number of hops between a node and Base Station. How can use Level?

When a node receives more than one Advertisement, it will choose the node with smaller Level; it will help us to restrict depth of network's tree. Therefore we use:

Chance = Reliability / Level

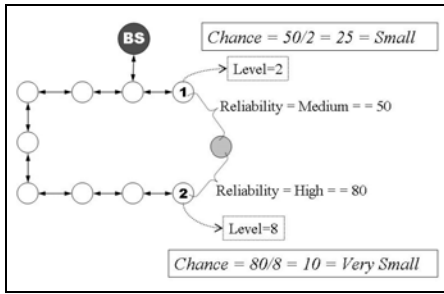| Energy Level	QoL	Mobility	QoRS	*Reliability*
High	*High*	*Low*	*High*	Very High
Low	*Low*	*High*	*Low*	Very Low

TABLE VI.   SOME FUZZY RULES

Figure 5.   A Chance computation scenario
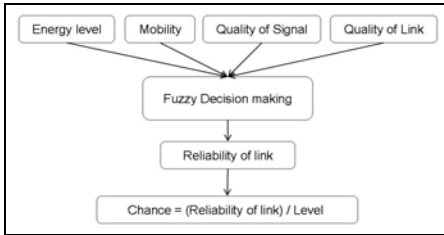

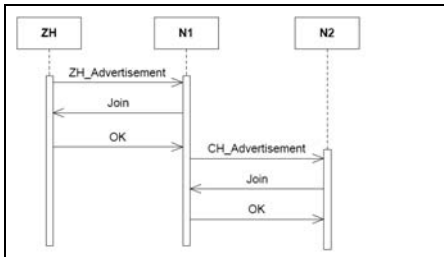Figure 6.   Fuzzy decision making process

Figure 7.   Communication between ZH and Node
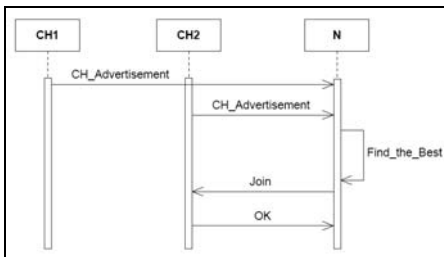

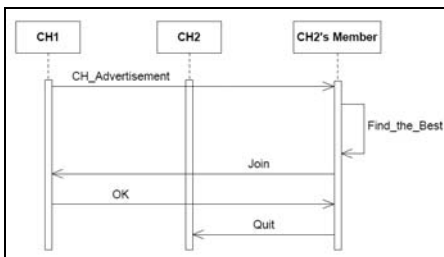Figure 8.   Communication between ZH and Node

Figure 9.   Find a new CH

Figure 5 shows a scenario of Chance computation. In this picture, the gray node wants to find a parent. It has two candidates, nodes 1 and 2. We can see that without using Level, the node will choose node 2, but the distance between 2 and BS is 8 hops and for 1, this distance is 2. Now by using Level, node 1 will be elected as CH for the gray node. Figure 6 also shows the fuzzy decision making diagram.

IV.   How does it work?

In the initial state, all of the nodes send Hello message periodically until find a CH, CN or the BS. The BS will send the zone address to ZHs. Each ZH will then send ZH_Advertisement periodically, until find the CHs up to its capacity. Each node that can receive ZH_Advertisement will send a Join message to the ZH to receive the cluster address. By receiving cluster address, the nodes will function as CHs, which means they will send CH_Advertisement periodically until find the cluster members up to their capacity. The CHs will construct the Clusters and each node is member of just one cluster. Each node will send Hello message to find a ZH, a CH or its neighbours. By finding a ZH, or a CH, the nodes will stop the Hello message. An important question is: How can detect the movement of a node? The answer is very simple. We can choose one of the approaches below:

- Sending ZH_Advertisement periodically, and when a CH receives a new ZH_Advertisement it will find a movement, because the ZHs are stationary and receiving a new ZH_Advertisement means that the CH is moved. In this case, it will send a movement message to the ZHs and the ZHs will send the Advertisement message to their CHs. When a CH received an Advertisement message, it will send CH advertisement to find the moved CH's members.

- Using ACK messages. That means, when a node sends data to its parent node, the parent will send back an ACK message by receiving the data, if the node did not received the ACK message from its parent node it will start to send Hello message until find another parent node. In this method we can use the ACK message of the MAC protocol.

In a ZH or a CH, by receiving a new Join request, they will verify the Admission condition. If it is not true, the node will send a Join_Other to one of his children that have the minimum QoL. This will help us to have clusters and also Zones with minimum Dynamicity.

A.   Message Exchange Diagrams

In figures 7 to 9, we can see the message exchange diagrams. Figure 7, shows the messages communicated between Root and Node1, the Node1 will be a CH and then it will communicate with Node 2. At the end Node 2 will be a Cluster1 member.

Figure 8, is the message exchange diagram between a two Cluster Heads and a Node. The Node received 2 CH_Advertisement and by using fuzzy logic decision making, it finds that CH2 is better and will send a Join to CH2.

In figure 9, you can see a Node that is a member of Cluster2 and receives a CH_Advertisement from CH1. By fuzzy logic decision making process, it found that CH1 is better, therefore it will send a Join to CH1 and after receiving the new address, it will send a Quit message to CH2.

V. SIMULATION

To simulate our approach, we have developed a special simulator with Visual C# .net. Figure 10 shows the interface of this simulation tool. Table VII shows the simulation parameters that we have used. Note that for focusing on the assessment of the performance of the proposed clustering algorithm, we do not generate any user data traffic during a simulation.

A. Node and Load Distribution

We used a random deployed network with 150 nodes and with a *Random Mobility Pattern*. To reduce the complexity of simulation we fixed the ZHs number to 6. Table VIII shows the results of run of the algorithm. In this table we can see that we have a uniform distribution of the nodes in zones and in the clusters with *Admission Condition*. Also the Load is well distributed between the ZHs. Table IX shows the network's state after 10 rounds. In our simulation each round is the change of all the CHs. In this table we can see after 10 rounds we have 7 zones in the network and we have the better Load distribution in the network.

B. Network Lifetime

We can see result of our simulation to evaluate the network lifetime in table IX. This table shows results of our simulation. In this simulation, we calculated the number of rounds before the first death in the network. As we said each round is the period that all the cluster heads are changed.

As we can see in this table, when we add the number of the nodes of the network, the lifetime will rise, because in this simulation the surface of simulation is fixed and adding the number of the nodes will be a help to better distribution of the Load. In our approach, to choose a CH we use many parameters as the election parameters and energy level, communication quality between the node and CH and the other parameters can help us to make the best decision with fuzzy logic.

C. Network Lifetime

We can see result of our simulation to evaluate the network lifetime in table X. This table shows results of our simulation. In this simulation, we calculated the number of rounds before the first death in the network. As we said each round is the period that all the cluster heads are changed. As we can see in this table, when we add the number of the nodes of the network, the lifetime will rise, because in this simulation the surface of simulation is fixed and adding the number of the nodes will be a help to better Load distribution.

In our approach, to choose a CH we use many parameters as the election parameters and energy level, communication quality between the node and CH and the other parameters can help us to make the best decision with fuzzy logic. In addition, in our approach the number of zones and also the number of clusters are not constant and can change in time, therefore with the best decision making and the best load and node distribution and also with variable number of zone and cluster we can have a greater life time for our network.
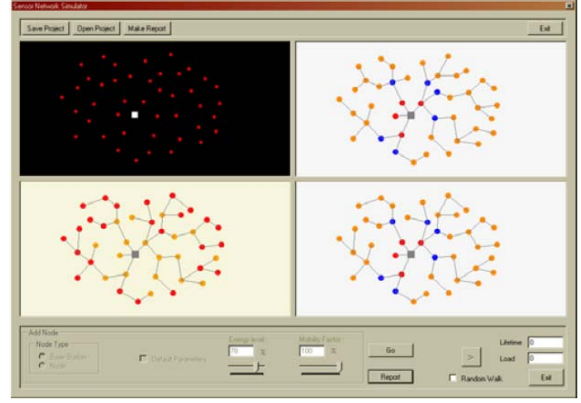

Figure 10. Our simulation tool's Interface

Speed of the Nodes	1 m/s
Initial Energy in each Node	5 J
Energy consumption(Calculation, receive and send)	10 nJ/bit
Packet size	100 Bytes
Dead Energy Level	1% * 5 J
Sensor communication radius	20 m
Surface of simulation	600m * 400m

TABLE VII.   SIMULATION PARAMETERS

Zone	Number of Members	Number of Clusters	Average QoL in Zone	Zone Head Load
1	29	21	Medium	High
2	10	6	Very High	High
3	28	25	High	Medium
4	37	30	High	Medium
5	23	18	Medium	Medium
6	23	17	High	High

TABLE VIII.   ALGORITHM WITH *ADMISSION CONDITION* IN FIRST ROUND

Zone	Number of Members	Number of Clusters	Average QoL in Zone	Zone Head Load
1	16	10	High	High
2	17	13	High	Medium
3	22	19	Medium	Medium
4	25	20	High	High
5	23	16	Medium	Medium
6	22	19	High	High
7	25	22	Medium	Medium

TABLE IX.   ALGORITHM WITH *ADMISSION CONDITION* AFTER 10 ROUNDS

Number of Nodes	Number of round before death of First Node of Network	Lifetime
100	213	2704
150	390	3898
200	425	4400
250	636	6730
300	545	5905
350	523	5707
400	565	6085
450	590	6480
500	611	6804

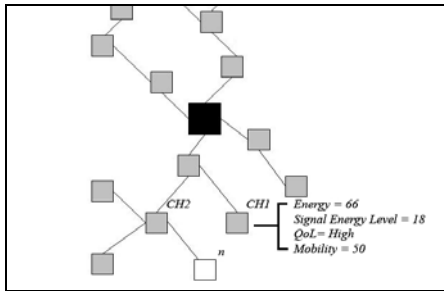TABLE X.   NETWORK LIFETIME SIMULATION

Figure 11. Simulation scenario

D. Parent selection

In this part we will describe a simulation scenario. The Figure 11 shows the simulated scenario. In this picture, the black node is BS and the node *n* that is the child of *CH2*, received an advertisement from *CH1* and is going to evaluate the *CH1*. The *QoL* between *CH2* an *n* is *low*. We verified our algorithm with and without fuzzy logic and to facilitate the simulation we fixed the Mobility and *QoL* in *CH1* and *CH2*.

The simulation show that with fuzzy logic the chance of CH1 for be the CH of n is calculated as *low,* and that means the node *n* will not change his parent. But without Fuzzy logic (Choosing the mean of the values as chance) the chance will be *Medium* that means, *n* will change his CH and will choose the CH1. Therefore in this case CH1 has the smaller life time to be the Parent of *n* than the CH2 and therefore after a shorter time, we will have another message exchange between *n* and a CH to find a new parent.

VI. CONCLUSION

In this paper, we have presented a location-unaware, energy-efficient distributed clustering approach for mobile wireless sensor networks. This approach uses fuzzy logic to select a cluster head. It can be applied to the design of sensor network protocols that require energy efficiency, scalability and mobility adaptation.

Simulation shows that our approach can easily manage mobility of the nodes and also shows that with fuzzy logic we have stable clusters and a cluster head has greater lifetime. Therefore with respect to mobility, we have the minimum message exchange and thus minimum energy consumption.

Through simulations we showed the effectiveness of the proposed algorithm. Our future work will be the formal validation of the proposed protocol with model checking technique. Moreover, for proving the implementability, we are also implementing our proposal on a MicaZ wireless sensor network.

REFERENCES

[1] C. R. Lin and M. Gerla, "Adaptive clustering for mobile wireless networks," *IEEE J. Select. Areas Commun.*, vol. 15, no. 7, pp. 1265–1275.

[2] Z. J. Haas and M. Pearlman, "The zone routing protocol (ZRP) for ad hoc networks," Internet Draft, Tech. Rep.

[3] W. R. Heinzelman, A. 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.

[4] Distributed Clustering in Ad-hoc Sensor Networks: A Hybrid, Energy-Efficient Approach – HEED, Ossama Younis and Sonia Fahmy, Department of Computer Sciences, Purdue University.

[5] I. Gupta, D. Riordan and S. Sampalli: Cluster-head Election using Fuzzy Logic for Wireless Sensor Networks. Faculty of Computer Science. Dalhousie University.