A Novel Cluster-based Self-organization Algorithm for Wireless Sensor Networks

M. Lehsaini, and M. Feham

***Abstract*—Wireless sensor networks (WSNs) consist of a large amount of tiny sensor nodes. Hence a hierarchical cluster-based structure can be used to deal the self-organization issues of large networks. This cluster-based organization can prolong network lifetime and reduce the broadcast overhead. In this paper, we propose an Efficient Self-Organization Algorithm for Clustering (ESAC), which uses a weight-based criterion for cluster-head's election. This weight relies on the combination of the k-density, residual energy and mobility. In ESAC, the node that has the greatest weight in its 2-hop neighborhood is chosen as cluster-head for a fixed service time $Time_{Service}$. Furthermore, ESAC enables to generate a low number of stable and balanced clusters. Simulation results show that ESAC provides better results when compared with WCA (Weight Clustering Algorithm), and with the algorithms proposed respectively by Lin et al. and Chu et al. in terms of the number of clusters formed. Besides, when we compared ESAC to LCC (Least Cluster-head Changes) with lowest ID algorithm, it provides good results in terms of the number of cluster-heads changes.**

***Keywords*—Cluster-based algorithm, k-density, Self-organization, Residual energy, Wireless Sensor Networks.**

I. Introduction

WSNs may be composed of a large number of small individual sensor nodes, which are randomly deployed in an interest area, and collaborate with each other to form a sensor network capable of reporting the phenomenon to a data collection point called sink or base station [1]. Nevertheless, they present some constraints such as low storage and processing power, limited battery lifetime, and short radio ranges. Furthermore, the networked sensors have many potential civil and military applications i.e., they can be utilized for object tracking, intrusion detection, habitat and other environmental monitoring, disaster recovery, hazard and structural monitoring, traffic control, inventory management in factory environment and health related applications etc. [11,13].

Information gathering in WSNs can follow different patterns, depending mostly on the specific needs of the applications. In a time-driven scenario, all sensors send data periodically to the sink. As opposed to this, in the event-driven case, sensors communicate with the sink only if a sensor node sense a particular event, i.e., a situation that is worth reporting, and in a query-driven scenario, a sensor transmits its collected data to the base station only in the case where if this latter requests it. Finally, in a dissemination-driven scenario, the base station consults or updates the overall sensor nodes of the network. Thus, to carry out information gathering while reducing the overall energy consumption in the network and the broadcast overhead, it is suitable to design an efficient scheme that is able to self-organize dynamically the network. One promising approach is to use clustering process, which has been considered as an efficient approach to mimic the operation of the centralised infrastructure. Therefore, we should involve determining factors in the calculation of node's weight to generate a reduced number of stable and balanced clusters with a bounded number of sensor nodes, and whose their members are at an adequate distance from their corresponding cluster-head.

Previous clustering schemes differ on the criterion for the cluster-head election, which are either based on the lowest (or highest) ID among all unassigned nodes, or based on the maximum node degree [15], or based on the mobility [4], or based on some generic weight [8] wherein the node with greatest weight will be selected as cluster-head. The computation of the node's weight takes into account various parameters such as node degree, mobility, and residual energy.

All of the above characteristics and constraints make the design of an efficient scheme for better management of WSNs a real challenge. In response to this challenge, we propose in this paper an Efficient Self-Organization Algorithm for Clustering (ESAC), which consists of grouping sensor nodes into a set of disjoint clusters. Each cluster has a designated leader called cluster-head, which is the node with greatest weight among its 2-hop not affiliated neighbors. The weight of each node is based on the combination of the following parameters: the 2-density of the node, residual energy, and node's mobility. Furthermore, the size of clusters is bounded by two values $Thresh_{Lower}$ and $Thresh_{Upper}$, which respectively represent the minimal and maximal number of sensor nodes that a cluster can contain. Besides, inside a cluster, each sensor node is at most two hops from its corresponding cluster-head contrary to the distributed algorithm Low-Energy Adaptive Clustering Hierarchy (LEACH) [16,17], which allows only single-hop clusters to be constructed.

Manuscript received January 21, 2008.

F. M. Lehsaini is with the Laboratory of Computer Engineering of Franche-Comté, 16, Route de Gray, Besançon 25030, France; phone: +33-3-81-66-20-75; fax: +33-3-81-66-64-50; e-mail: lehsaini@lifc.univ-fcomte.fr.

S. M. Feham is with the Faculty of Engineering Sciences, Tlemcen University, 13000; e-mail: m_feham@mail.univ-tlemcen.dz.

Since cluster members do not transmit their gathered data directly to the sink, but only to their respective cluster-head. Accordingly, the cluster-head should be responsible for coordination among the cluster members, aggregation of their data, and transmission of the aggregated data to the sink, directly or via multi-hop transmission mode. Thus, cluster-heads support a heavy load.

In WSNs, clustering algorithm would avoid a fixed cluster-head election scheme, because this latter with constrained energy may drain its battery power rapidly due to its heavy utilization, what causes bottleneck failures in its cluster, and thereafter triggers the cluster-head election process. Therefore, we foresaw in ESAC that the cluster-head election process will be carried out periodically after each service time $Time_{Service}$ to balance the load between the nodes.

We aimed with ESAC algorithm to generate a low number of stable and balanced clusters, while guaranteeing a long sensor lifetime and efficiently maintain these clusters. Hence, once the network is divided into smaller logically disjoint clusters, it will be easy to carry out the cluster maintenance process, which relates the admission of new sensor nodes inside a cluster or the departure of the sensor nodes from it either by migration to other clusters or by exhaustion of their battery power.

Finally, ESAC algorithm was simulated and compared on the one hand with WCA [8], and those proposed respectively by Lin et al. [19] and Chu et al. [18] in terms of the number of clusters formed, and the other hand to LCC[10] in terms of the number of cluster-heads changes.

The remainder of this paper is organized as follows. In Section 2, we provide preliminaries necessary for describing our scheme. Section 3 reviews several cluster-based algorithms proposed previously. In Section 4, we present our new weighted algorithm, and Section 5 presents the performance analysis of the proposed algorithm. Finally, Section 6 concludes our paper by pointing out some possible future research directions.

II. PRELIMINAIRES

Before heading into the technical details of our contribution, we first give some definitions and notations that will be used in our paper later.

A wireless sensor network is abstracted as an undirected graph $G=(V,E)$, called a connectivity graph, where V represents the set of wireless nodes and $E\subseteq V^2$ is the set of edges that gives the available communications : an edge $e=(u,v)$ belongs to E if and only if the node u is physically able to transmit messages to v and vice versa. Each sensor node $u\in V$ is assigned a unique value to be used as an identifier, so that the identifier of u is denoted by $Node_{Id}(u)$ and all links in the graph are bidirectional. The set of neighbors of a node u is represented by $N_1(u)$ where $N_1(u)=\{v\in V/v\neq u \wedge (u,v)\in E\}$. The size of this set is known as the degree of u, denoted by $\delta(u)$. The set of two-hop nodes of node u i.e. the nodes which are the neighbors of node u's neighbors except for the nodes that are the neighbors of node u, is represented by $N_2(u)$ as follows:
$N_2(u) = \{w\in V/(v,w)\in E \text{ where } w\neq u \wedge w\in N_1(u) \wedge (u,v)\in E\}$. The combined set of one-hop and two-hop neighbors of u is denoted as $N_{12}(u)$ wherein $N_{12}(u)=N_1(u)\cup N_2(u)$. In a general manner, the set of k-hop neighbors of a node u is represented by $N^k(u)=\{v\in V/v\neq u \wedge d(u,v)\leq k\}$ and its closet set of k-hop neighbors is denoted by $N^k[u]$ where $N^k[u]= N^k[u]\cup\{u\}$. Here, $d(u,v)$ represents the minimal distance in number of hops from u to v. The size of $N^k(u)$ is known as the k-degree of u.

The k-density of a node u represents the ratio between the number of links in its k-hop neighborhood (links between u and its neighbors and links between two k-hop neighbors of u) and the k-degree of u; formally, it is represented by the following formula:

$$k-\text{density}(u)=\frac{\left|(v,w)\in E \ / \ v,w\in N^k[u]\right|}{\left|N^k(u)\right|}$$

However, we are interested only in calculation of the 2-density nodes not to weaken ESAC algorithm of its performance. Hence, the previous formula results from it:

$$2-\text{density}(u)=\frac{\left|(v,w)\in E \ / \ v,w\in N_{12}[u]\right|}{\left|N_{12}(u)\right|}$$

Fig. 1 and Table I illustrate an example of the 2-density calculation.

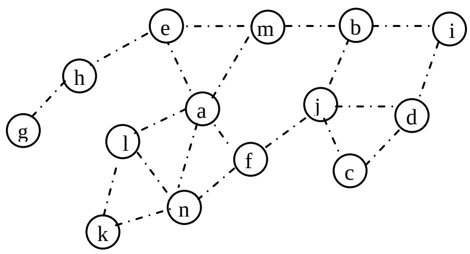

Fig. 1. Example of an abstracted network

TABLE I
CALCULATION OF THE 2-DENSITY

Node	a	b	c	d	e	f	g	h	i	j	k	l	m	n
1-density	1,60	1	1,66	1,33	1,33	1,33	1	1	1	1,25	1,66	1,66	1,33	1,75
2-density	1,55	1,50	1,40	1,40	1,37	1,60	1	1,25	1,40	1,50	1,75	1,60	1,44	1,57

In this paper, we assume that all sensor nodes are given in a two dimensional space and we measure the distance between two nodes u and v in terms of number of hops, which is simply the minimum number of edges that a message has to cross from u to v. Each sensor node has an omni-directional antenna what allows for a single transmission of it can be received by all nodes within its vicinity and we consider that the nodes are almost stable in a reasonable period of time during the execution of the clustering process. We also assume that each sensor node has a generic weight and it is able to evaluate it. Besides this weight represents the fitness of each node to be a cluster-head and the higher weight means the higher priority.

III. RELATED WORK

In the recent years, several cluster-based algorithms are proposed in the literature [2,3,8,9,15,16,19,20,23,24]. These algorithms considered different weights as a priority criterion to elect whether a node will be a cluster-head. However, most of these proposed weighted clustering algorithms applied the simple greedy algorithms where the nodes with highest weight become cluster-heads. For example, Chatterjee et al. [8] considered a combined weight metric for their clustering algorithm, which takes into account several parameters such as node's degree, transmission power, mobility and the battery power of the node. As the node that has the greatest weight among its neighbors is elected as cluster-head.

In [16], authors propose LEACH, which is a distributed, single hop clustering algorithm for homogeneous WSNs. In LEACH, cluster-head role is periodically rotated among the sensor nodes to evenly distribute energy dissipation. Moreover, since LEACH allows only single hop clusters to be constructed, it may generate a high number of clusters. Therefore, it results an increase of the overhead for the inter-cluster communications. In [20,21], the authors proposed an enhanced version of LEACH wherein the cluster members can be to more a hop from their corresponding cluster-head and communicate with it in multi-hop mode. Nevertheless, this strategy requires that each sensor node is capable to aggregate data, what increases the overhead for all the nodes. Therefore, to improve the performance of this strategy, in [23] the authors use heterogeneous sensor nodes instead of using homogeneous sensor nodes, where two types of sensor nodes are deployed: super sensor nodes and basic sensor nodes. The super sensor nodes have more capabilities on processing and communication, and act as cluster-heads, whereas the basic sensor nodes are simple with limited power, affiliate to their corresponding cluster-head and communicate with them via multi-hop mode.

Gerla and Tsai [15] studied two algorithms, the first rely on lowest ID and the second on highest node's degree. Since the change of the topology provokes the change of the nodes' degree, there is a strong probability that a cluster-head may give up its statute once that the topology changes. Therefore, the process of the clusters rebuilding will be triggered to structure the network again. Furthermore, the lowest ID algorithm always promotes the nodes with low ID to become cluster-heads and probably for a long time. Therefore, these nodes exhaust rapidly their battery power what causes a bottleneck in the cluster. We notice that in both algorithms, the clustering process does not guarantee the stability of the clusters because the cluster-head relinquishes its statute as and when another prospective (lowest ID/highest degree) node joins the cluster or the topology of the network changes. In our algorithm, the cluster-head keeps its statute until either it leaves the cluster or the service time $Time_{Service}$ is expired for its cluster-heads' role.

Lin and Gerla [19] have proposed an improved version of the lowest ID algorithm by using a distributed technique with a limited number of hops in the cluster as constraint. In this proposed version, the nodes communicate with each other in at most two hops inside a cluster. That may generate an important number of clusters with a reduce number of nodes or the clusters with an important size in the case of the dense networks. Hence, their management may become very difficult. Furthermore, when the number of nodes increases, a gradual degradation in WSNs performance is observed. This anomaly occurred because this algorithm does not put any restriction on the clusters size whereas with a limited number of nodes inside a cluster, it will be easy to manage the intra-cluster communications. Thus, it seems that additional procedures for merging or rearranging clusters may be desirable to deal the compromise between the generation of a great number of clusters with a reduced number of nodes and the generation of a small number of clusters with an important number of nodes. Hence, we have proposed a cluster maintenance scheme that has tackled this problem by using merging of clusters based on a threshold size in terms of the number of the nodes per cluster.

On the other hand, Chen and al. [9] have combined Highest-Degree and Lowest-ID approaches for the cluster-heads election process, wherein Highest-Degree is considered as a primary criterion and in case of tie lower ID as a secondary criterion. Moreover, in the generated clusters by their approach, all nodes are at a distance of at most *k* hops from the cluster-head. In another work, Chen et al. proposed an unified cluster-based scheme for wireless networks, in which each node has a weight, which takes into account suitability of each node for the cluster-head role, speed, degree, power, and energy left. Nevertheless, this proposed scheme enables to generate 1-hop and 2-hop clusters without taking into account the cluster size. However, in our algorithm, we restrict both the diameter and the size of the cluster.

WCA [8] considers the node's degree, transmission power, mobility, and battery usage in electing cluster-heads. It limits the size of the formed clusters so that the cluster-heads can support the load without causing degradation in performance. Although WCA has proved better performance than all the previous algorithms, it presents a drawback in knowing the weights of all nodes before starting the clustering process. Therefore, it results that the overhead induced by WCA is very high, as well as it can cause the energy draining of the cluster-head rapidly if this latter may keep its statute for a long time. These limitations prove that WCA is only suitable for small networks.

The Distributed and Mobility Clustering Algorithm (DMCA) [2] uses generic weights associated to the nodes and elects the node that has highest weight among its one-hop neighbors as cluster-head. It involves the node mobility during or after the clustering set-up phase. As it is proven in [6], DMCA is suitable for the networks in which the nodes are static or moving at a very low speed but its performance degrades considerably with the greatly mobile networks. For that reason, Basagni [3] has proposed an extended version of DMCA algorithm, called Generalized Distributed and Mobility-Adaptive Clustering Algorithm (DMAC) to deal the DMCA limitations. It implies that, when, due to mobility of the nodes, two or more cluster-heads become neighbors, none has to resign. Thus, it results that the clustering management with GDMAC requires less overhead than the one with DMAC in highly mobile environment. GDMAC algorithm is analyzed in [5], with respect to their convergence time and message complexity. Moreover, all of the above algorithms (WCA, DCA, DMAC, GDMAC) generate one-hop clusters, and have a complexity of $O(N)$. This allows them to be suitable only for networks with a restricted number of nodes.

K-clustering [14] enables to create non-overlapping clusters wherein the cluster diameter is at most *K* hops without taking into account their size. This may lead to generate the clusters having a large number of sensor nodes and thereafter complicate their management. The main purpose of this strategy is to support only partial information of network topology. However, its main disadvantage that it can generate no balanced clusters since there is no relationships between the diameter of the cluster and its size. Similarly, the clustering technique proposed by Lin and Chu [18] uses hop distance as parameter to control the cluster structure during their formation. In this technique, a randomize node takes the cluster-head role and the distance between it and each cluster member is within a predetermined maximum number of hops. However, restricting the number of hops does not restrict the number of nodes in the clusters. Thus, this technique can lead to the formation of the clusters with a large number of nodes as in K-clustering strategy. Therefore, it leads to the similar problems of *K*-clustering discussed earlier.

Our algorithm should coordinate between the size of the clusters and their diameter to generate balanced clusters. Hence, we used two thresholds $Thresh_{Lower}$ and $Thresh_{Upper}$ to control the size of generated clusters, as well as we fixed the distance between the cluster members and their corresponding cluster-head at most two hops. In *K*-clustering only the root of the sub-tree knows which belongs to the cluster, the other nodes of the sub-tree are unaware of their cluster members. However, in our algorithm, all nodes are aware of the cluster to which they belong.

IV. Our contribution

As stated above, a WSN consists of a large amount of individual sensor nodes. Therefore, it is essential that the network be able to self-organize. Moreover, the development of an effective algorithm for the clustering will be beneficial to obtain better performance. Hence, we used determining parameters for the cluster-head election, which permit to create stable and balanced clusters.

A. Cluster formation

In this section, we propose a new weight-based clustering algorithm, which consists of grouping sensor nodes into a set of disjoint clusters, hence giving at the network a hierarchical organization. Each cluster has a cluster-head that is elected among its 2-hop neighborhood according to their weights. The weight of each sensor node is a combination of the following parameters k-density, residual energy and mobility as illustrated by the formula:

$$\text{Weight}(u) = \alpha \cdot p_{\text{k-Density}} + \beta \cdot p_{\text{Res-Energy}} + \gamma \cdot p_{\text{Mobility}}$$

$$\text{with} \quad \alpha + \beta + \gamma = 1$$

Moreover the sum of the factors (α,β,γ) is equal to 1 and the coefficient of each parameter can be chosen depending on the application. For example, if we aim to generate stable clusters, we will assign to α an important value relatively to the other factors, whereas in low mobility environment, we can privilege the residual energy of a node, thus factor γ can be chosen smaller. Therefore, we attribute adequate values to the different coefficients in the purpose to generate stable clusters and guarantee a long network lifetime.

In our context, we grant to the cluster-head the responsibility to coordinate among the cluster members, aggregate their data and transmit them to the sink, directly or via multi-hop transmission mode. Accordingly, cluster members do not transmit their gathered data directly to the sink, but only to their corresponding cluster-head. In spite of this heavy load supported by the cluster-head, we find several cluster-based algorithms such as WCA scheme uses the cumulative time for cluster-head's role. However, that can exhaust rapidly its battery power and thereafter degrade considerably network performance. Hence, we proposed to set up periodically cluster-head election process. This strategy enables to prolong the lifetime of sensor nodes while forcing them to relinquish cluster-head role after the end of service time for playing this role.

In this paper, we aimed to design a cluster-based network architecture, wherein cluster formation takes into account the following constraints: each cluster has a size ranging between two thresholds $Thresh_{Upper}$ and $Thresh_{Lower}$ except in certain case its value can be lower than $Thresh_{Lower}$, and in which cluster members are at most 2-hops from their respective cluster-head. If during set-up phase, there was formation of clusters whose size is lower than $Thresh_{Lower}$, then re-affiliation process will be triggered. Furthermore, a cluster-head could be able to manage its cluster members, to accept or refuse adhesion of new arrivals based on its capacity without perturbing the functionality of the other cluster members.

In the proposed strategy, each node u is identified by a state vector: $(Node_{Id}, Node_{CH}, Weight, Hop, Size, Thresh_{Lower}, Thresh_{Upper})$ where $Node_{Id}$ is the identifier of sensor node, $Node_{CH}$ represents the identifier of its cluster-head, in particular if this node is a cluster-head then its identifier is assigned to $Node_{CH}$, *Hop* indicates the number of hops separating it from its respective cluster-head, and Size represents cluster size to which it belongs. Moreover, each node is responsible to maintain a table called "$Table_{Cluster}$" wherein the information of the local members cluster is stored. The format of this table is defined as $Table_{Cluster}(Node_{Id}, Node_{CH}, Weight)$. Sensor nodes could coordinate and collaborate between each other to construct and update the above stated table by using Hello message. Furthermore each cluster-head maintains another cluster-head information table so called "$Table_{CH}$" in which the information about the other cluster-heads is stored. The format of these tables is represented and $Table_{CH}(Node_{CH}, Weight)$. These tables contain the state vector of the nodes, which should be periodically exchanged either between cluster-heads or between each cluster-head and its cluster members to construct or update "$Table_{Cluster}$" and "$Table_{CH}$" respectively. The weight of each node is periodically calculated and exchanged among its 2-hop neighborhood to choose the node being appropriate to be a cluster-head.

In our approach, we tried to organize the sensor nodes into clusters by affiliating each sensor node to the nearest cluster-head from it. We used Hello messages for cluster formation in order to avoid the broadcast overhead and not degrade algorithm of its performance. Hence, at the beginning each sensor node calculates its weight and generates a Hello message, which includes two extra fields addition to other regular contents: weight and $Node_{CH}$, where $Node_{CH}$ is set to zero. Furthermore, clustering process is performed in two consecutive phases as well as the clusters are formed the ones after the others.

1) *The first phase*

Cluster-head election process proceeds in the following way as illustrated by Fig.2. Initially, a random node initiates clustering process while broadcasting a Hello message to its $N_{12}(u)$ neighbors. Then, node that has greatest weight among its $N_{12}[u]$ neighbors will be elected as cluster-head *(CH)*. This latter updates its state vector by assigning to $Node_{CH}$ the value of its identifier ($Node_{Id}$), sets respectively *Hop* value and *Size* value with 0 and 1. After that, it broadcasts advertisement message *ADV_CH* including its state vector to its 2-hop neighborhood to request them to join it. Each node belonging to $N_1(Node_{CH})$ whose $Node_{CH}$ value is equal to zero i.e. does not belong to any cluster and its weight is lower than CH's weight, transmits *REQ_JOIN* message to *CH* to join it. Corresponding cluster-head checks if size of its own cluster does not reach $Thresh_{Upper}$ i.e. *Size* value is lower than $Thresh_{Upper}$, it transmits *ACCEPT_CH* message to this node otherwise it simply drops the message of affiliation demand. Thereafter, *CH* increments its Size and the affiliated node sets *Hop* value with 1 and $Node_{CH}$ with $Node_{CH}$ of its corresponding cluster-head, then it broadcasts received message again with the same transmission power to its neighbors. Similarly, each sensor node belonging to $N_2(Node_{CH})$, which is not affiliated to any cluster as its weight is lower than that of *CH*, transmits *REQ_JOIN* message to corresponding *CH*. In the same way, *CH* checks if its *Size* value is always less than $Thresh_{Upper}$, so yes it updates its state vector; otherwise it drops message of affiliation demand. Finally, when no more Hello messages are broadcasted in the network, each sensor node will know which cluster it belongs to and which sensor node is its cluster-head. Clustering process will end after a fixed interval of time, which should be long enough to guarantee that every sensor can find its nearest cluster-head. Fig.3 illustrates the pseudo-code of the set-up phase.

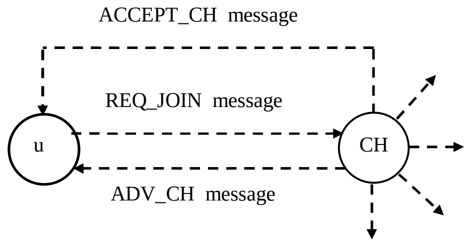

Fig. 2. *Node u joins a cluster*

Pseudo-code of set-up phase

1: Assign values to α, β and γ;
2 : Initialize *($Time_{Cluster}$)*
  /* Initialize the state vector of all nodes
3: For each Node $u \in G$ do
  $Node_{Id}$; /* Identifier of the node
  $Node_{CH}=0$
  $Size=0$
  $Hop=0$
 End for
  /* each node computes its weights according to its
  /* 2-density and residual energy.
4: For each Node $u \in G$ do
  $Weight(u)=\alpha*P_{2\text{-}density}+\beta*P_{Res\text{-}Energy}$
5: Repeat
 /* a random node u broadcasts a Hello message to $N_{12}(u)$
6: Broadcast Hello message by u;
 /* Choose node that has greatest weight among $N_{12}[u]$
 /* nodes as cluster-head
7: Choose $v \in N_{12}[u]$:
  $Weight(v)=Max(Weight(w) \mid w \in N_{12}[u])$
 /* Update CH's state vector
8: *Update_CH_State()*;
  $CH\text{->}Node_{CH}=CH\text{->}Node_{Id}$
  $CH\text{->}Size=1$
  $CH\text{->}Hop=0$
9: Send periodically *ADV_CH* message by *CH* to $N_{12}[u]$
10: Initialize *($Time_{CH}$)*
11: Repeat
12: if (*REQ_JOIN* is received from $u \in N_{12}[CH]$)
13: Send *ACCEPT_CH* to u
14: Perform Affiliation Procedure by *CH*
  $CH\text{->}Size=CH\text{->}Size+1$
15: Perform Adhesion procedure by u
  $u\text{->}Node_{CH}=CH\text{->}Node_{CH}$
  if $(u \in N_1[CH])$ $u\text{->}Hop=1$
  else $u\text{->}Hop=2$
16: *Update ($Table_{Cluster}$);*
  End if
17: Until *($CH\text{->}Size=Thresh_{Upper}$) Expired($Time_{CH}$)*
18: Until *Expired($Time_{Cluster}$)*

Fig. 3. *Pseudo-code of set-up phase*

2) *The second phase*

During the first phase, it may not be possible for all clusters to reach $Thresh_{Upper}$ threshold. Hence, we tried to reduce the number of clusters formed during this second phase. For that, we proposed to re-affiliate the nodes belonging to clusters that have not attained cluster size $Thresh_{Lower}$ to clusters that did not reach cluster size $Thresh_{Upper}$.

Since there is no constraint relating to the generation of clusters having a number of nodes lower than $Thresh_{Lower}$ during the execution of first phase, it is possible that there is creation of this type of clusters during this phase. For that, we proposed this second phase in order to reorganize the clusters, reduce their number and thereafter obtain balanced and homogeneous clusters.

The execution of the second phase proceeds in the following way. Cluster-heads that belong to clusters whose size is strictly lower than $Thresh_{Upper}$, broadcast a new message called *RE-AFF_CH* to re-affiliate nodes belonging to the small clusters to them. Then, each node that receives this message and belongs to a small cluster, should re-affiliate to the nearest cluster-head whose weight is greater than its, and the size of its own cluster does not always reach $Thresh_{Upper}$ threshold.

After the unfolding of our algorithm, we obtain balanced and stable clusters considering that we have involved k-density and residual energy to structure network in clusters. Moreover, ESAC avoids a fixed cluster-head election not to exhaust its battery power quickly due to its heavy utilization. Hence, cluster-head election process is carried out periodically after each service time $Time_{Service}$. Therefore, weight of each node is periodically calculated in order to illustrate the suitability of a node for playing cluster-head's role. We note that ESAC enables to generate clusters whose size does not reach $Thresh_{Upper}$ and their members are at most two hops from their corresponding cluster-head. Fig.4 represents the pseudo-code of re-affiliation phase.

Pseudo-code of re-affiliation phase

```
1: if (|Cluster| < Thresh_Upper)
2:   Send Re-AFF_CH message by CH
3:   if ( Re-AFF_CH is received from u∈N_12[CH])
        /* u belongs to a cluster whose size is lower than
        /* Thresh_Lower
4:      Send ACCEPT_Re-AFF message by CH
5:      Update u's state vector
          u->Node_CH=CH->Node_CH
          if (u∈N_1[CH])    u->Hop=1
           else u->Hop=2
6:      Update CH's state vector
          CH->Size=CH->Size+1
7:      Update (Table_Cluster);
```

Fig. 4. *Pseudo-code of re-affiliation phase*

B. Cluster maintenance

In our approach, cluster maintenance process should be triggered in the case where a cluster loses its cluster-head when this latter exhausts its battery power. Moreover, cluster-head's re-election process only concerns the cluster that lost its cluster-head and the future cluster-head would be chosen among the members of this cluster. We adopted this solution not to weaken our algorithm of its performance and avoid chain reactions that can occur during the launching of clustering process.

Cluster maintenance process is performed similarly as the set-up phase wherein a random node among the members cluster initiates the clustering process.

V. SIMULATIONS

In our experiments, we conducted extensive simulations to evaluate ESAC algorithm performance and compare it on the one hand with WCA algorithm, and those proposed respectively by Lin et al. and Chu et al. in terms of the number of clusters formed, and on the other hand with LCC algorithm in terms of the number of cluster-heads changes. To achieve these goals, we used NS-2 simulator [25] to implement ESAC algorithm. Moreover, since the mobility is considered, we chosen Random WayPoint model (RWP) [22] with zero pause time (continuous mobility) to generate the scenarios of the nodes mobility. In model RWP, each node chooses its direction and its displacement speed after every fixed interval of time.

In this section, at the beginning we evaluated the average number of clusters formed with varying transmission range in networks with various sizes. Then, we estimated the average number of clusters formed according to the speed in the networks wherein the nodes move slowly as in those wherein the nodes move with a high speed. For that, we considered a network topology where the sensor nodes are placed randomly on an area of size $100m \times 100m$ by using a uniform distribution function, we set the node's communication radio to 25 m, and we assumed that MAC layer behavior has not been taken into account so far, i.e. packet collisions do not occur in the simulation.

The simulations are carried out during 120s and the average values are calculated after each second. After this time, ESAC algorithm allots randomly different energy levels to the various nodes and triggers cluster-head's election process again. We adopted several contexts to carry out simulations according to the models used to evaluate the protocols, which we have chosen to compare them with ESAC performance.

At the beginning, to evaluate ESAC in terms of average number of clusters formed, we performed simulations using two distinct values for threshold $Thresh_{Upper}=50$ and $Thresh_{Upper}=30$ and a fixed value threshold $Thresh_{Lower}=10$. For that, we performed ESAC with network size 50, 75, and 100. In this model, the nodes move with a speed ranging between 0 and 10 (m/s).

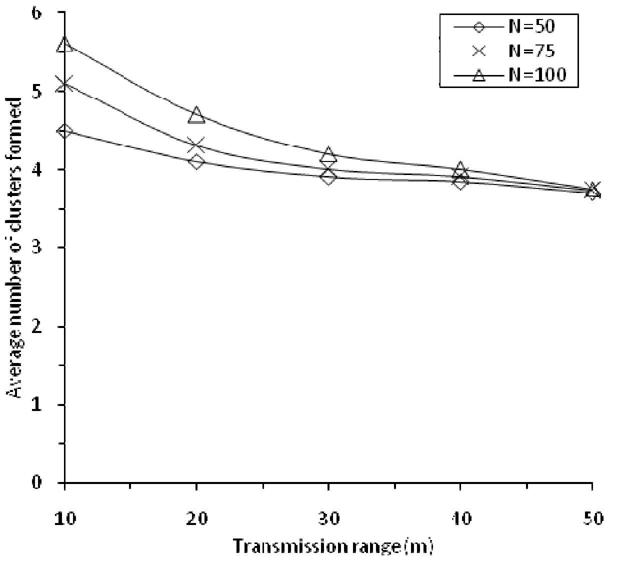

Fig. 5(a). Average number of clusters formed vs. transmission range.

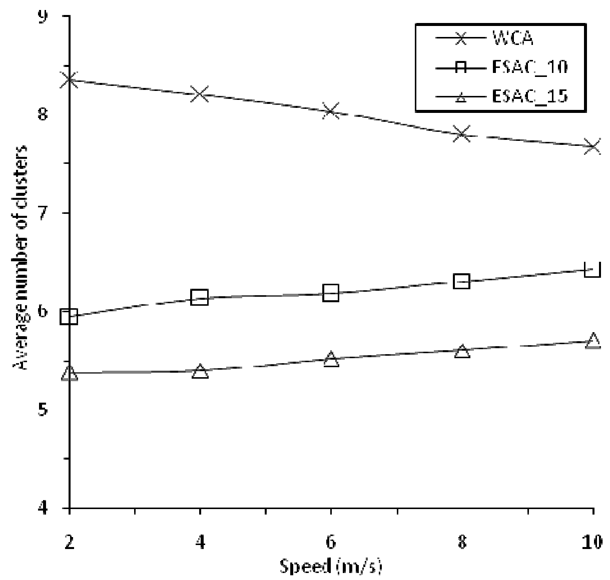

Fig. 6(a). Average number of clusters formed vs. speed

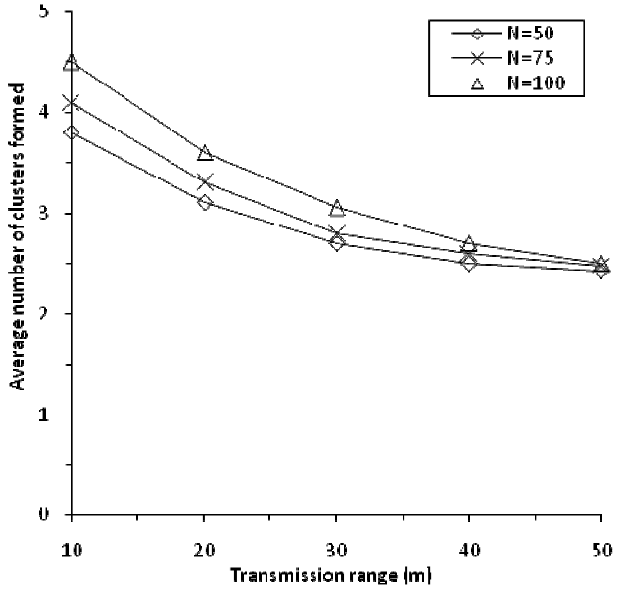

Fig. 5(b). Average number of clusters formed vs. transmission range.

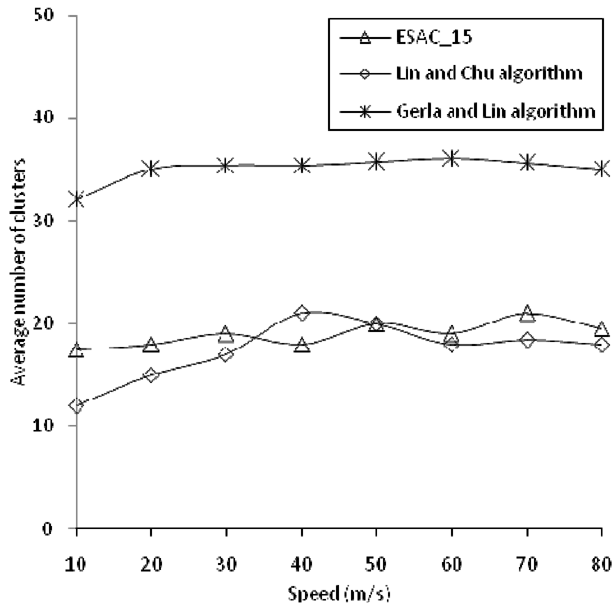

Fig. 6(b). Average number of clusters formed vs. speed

Fig. 5(a) and Fig. 5(b) show that the proposed approach allows the formation of a rather reduced number of clusters in both cases. Indeed, with networks respectively containing 50, 75 and 100 sensor nodes, ECSA_30 and ECSA_50 respectively generate 3,5 and 2.5 clusters when the transmission range is equal to 50 m.

In the second context, to compare ESAC with WCA, we used the same model presented in [8]. In addition, we performed simulations using two distinct values for threshold $Thresh_{Upper}=10$ and $Thresh_{Upper}=15$ and a fixed value threshold $Thresh_{Lower}=5$.

Fig. 6(a) and Fig. 6(b) respectively illustrate the average number of clusters formed in a network containing 40 nodes that move slowly, and in a network that contains 200 nodes moving with high speed. We observe that ESAC algorithm generates less clusters than WCA in both considered models, as their average number show a low variation in value. Furthermore, we notice that ESAC performance outperforms those of WCA [8] and the algorithm proposed by Gerla and Lin [19], and shows similar behavior to the algorithm proposed by Lin and Chu [18]. This may be due to the execution of the second phase of our algorithm which permits to affiliate the nodes belonging to the clusters which have a size lower than $Thresh_{Lower}$ to the nearest clusters.

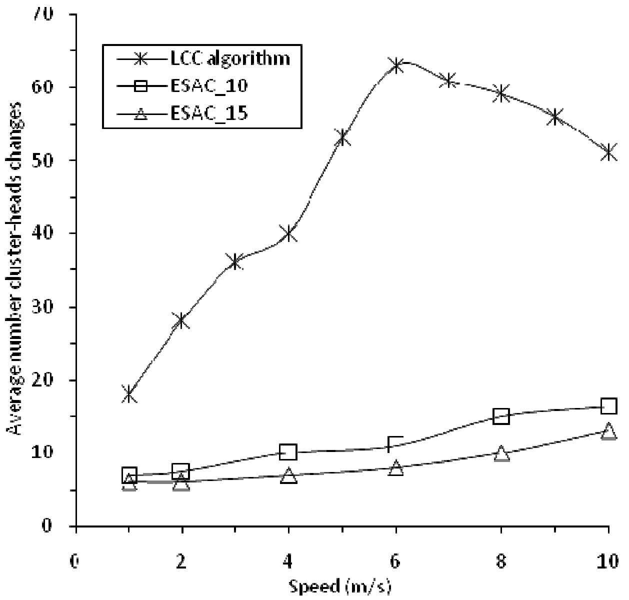

Fig. 7. Average number of cluster-heads changes vs. speed

Fig. 7 shows and compares the average number of cluster-heads changes when we apply algorithms LCC and ESAC. We notice that with LCC algorithm the cluster-heads give up their statute when node's speed increases. This is due on the one hand to the frequent changes of affiliation clusters by the nodes and on the other hand when two cluster-heads become neighbors, then the cluster-head with higher node ID would relinquish its cluster-head's role. However, in ESAC the cluster-heads changes are not frequent because the re-election of a new cluster-head before the end of the service time $Time_{Service}$ is done only either when a cluster-head leaves its own cluster or it exhausts its battery power. Besides, the future cluster-head will be elected among cluster members. It is suitable that a cluster-head keeps its statute for a long time because the frequent changes of cluster-head may negatively affect the performance of the clustering algorithm.

VI. Conclusion and Future Work

In this paper, we proposed an Efficient Self-Organization Clustering Algorithm for Clustering (ESAC) based on the weighting parameters k-density, residual energy and node mobility for cluster-heads' election, which enables to generate an low number of stable and balanced clusters, minimize the chain reactions for the rebuilding of the clusters. Furthermore, ESAC algorithm avoids a fixed cluster-head election not to drain its battery rapidly due to its heavy utilization contrary to several clustering algorithms. Hence, ESAC periodically carries out cluster-head election process to choose the node that has greatest weight in its 2-hops neighborhood as cluster-head. Therefore, ESAC algorithm permits to prolong the sensor lifetime by relinquishing cluster-head role once the service time $Time_{Service}$ for playing this role is expired.

Moreover, ESAC creates balanced clusters with a number of nodes facilitating their management and whose members are at most two hops from their corresponding cluster-heads. Simulation results show that ESAC provides better performance than WCA, the algorithm proposed by Gerla and Tsai and that proposed by Lin and Gerla in terms of the average number of formed clusters. Furthermore, it outperforms LLC algorithm in terms of number of cluster-heads changes.

With these observations and results obtained, ESAC algorithm can be promising to maximize lifetime and to minimize broadcast overhead in WSNs. Therefore, its evaluation could be the subject of future work.

Acknowledgment

The authors would like reviewers for their valuable feedback.

References

[1] N. Ahmed, S. S. Kanhere, and S. Sanjay, "The holes problem in wireless sensor networks: a survey," *ACM SIGMOBILE Mobile Computing and Communications,* vol. 9 N°.2, pp. 4-18, 2005.

[2] S. Basagni, "Distributed clustering for ad hoc networks," in *Proceedings of International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN),* Fremantle, Australia, June 1999, pp. 310-315.

[3] S. Basagni, "Distributed and mobility-adaptive clustering for multimedia support in multi-hop wireless networks," in *Proceedings of the IEEE 50th International Vehicular Technology Conference (VTC'99), vol.* 2, 1999, pp. 889–893.

[4] P. Basu, N. Khan, and T.D.C. Little, "A mobility based metric for clustering in mobile ad-hoc networks," in *Proceedings of Distributed Computing Systems Workshop*, 2001.

[5] C. Bettstetter and B. Friedrich, "Time and message complexities of the generalized distributed mobility-adaptive clustering (GDMAC) algorithm in wireless multihop networks," in *VTC'03: Proceedings IEEE Vehicular Technology Conference*, 2003, pp. 176-180.

[6] C. Bettstetter and S. König, "On the message and time complexity of a distributed mobility-adaptive clustering in wireless ad hoc networks," in *Proceedings European Wireless (EW)*, Florence, Italy, February 2002.

[7] C. Bettstetter and R. Krausser, "Scenario-based stability analysis of the distributed mobility-adaptive clustering (DMAC) algorithm," in *Proceedings of the 2nd ACM Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc'01),* 2001, pp. 232-241.

[8] M. Chatterjee, S.K. Das, and D. Turgut, "WCA: A Weighted Clustering Algorithm for Mobile Ad Hoc Networks," *Cluster Computing Journal*, vol. 5, N°. 2, pp. 193-204, 2002.

[9] G. Chen, F. Nocetti, J. Gonzalez, and I. Stojmenovic, "Connectivity-based K-hop clustering in wireless networks," in *Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02),* 2002.

[10] C. C. Chiang, H. K. Wu, W. Liu, and M. Gerla, "Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel," *IEEE Singapore International Conference on Networks (SICON*), April 1997.

[11] C. Y. Chong and S. P. Kumar, "Sensor networks: Evolution, opportunities, and challenges," in *Proceedings of the IEEE*, vol. 91 N° 8, 2003, pp. 1247-1256.

[12] A. Depedri, A. Zanella, and R. Verdone, "An energy efficient protocol for wireless sensor networks," in *Autonomous Intelligent Networks and Systems (AINS 2003*), Menlo Park, CA, June 2003.

[13] D. Estrin, R. Govindan, J. Heidemann, and S. Kumar, "Next century challenges: Scalable coordination in sensor networks," in *IEEE/ACM MobiCom'99*, pp. 263-270, August 1999.

[14] Y. Fernandess and D. Malkhi, "K-clustering in wireless ad-hoc networks," in *2nd ACM Workshop on Principles of Mobile Computing (POMC'02)*, Toulouse, France, 2002.

[15] M. Gerla and J. T. C. Tsai, “Multicluster, Mobile, Multimedia Radio Network,” *ACM/Baltzer Wireless Networks Journal*, vol. 1 N° 3, pp. 255-265, 1995.

[16] W.R. Heizelman, A. Chandrakasan, and H. Balakrishnan, “Energy-Efficient Communication Protocol for Wireless Micro Sensor Networks,” in *IEEE Proceedings of the Hawaii International Conference on System Sciences (HICSS '00)*, January 2000.

[17] W. R. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, “An application-specific protocol architecture for wireless microsensor networks,” *IEEE Transactions Wireless Communications*, vol. 1 N°. 4, pp. 660-670, 2002.

[18] H.C. Lin, Y.H. Chu, “A clustering technique for large multihop mobile wireless networks,” in *Proceedings of the IEEE Vehicular Technology Conference*, May 2000.

[19] C. R. Lin and M. Gerla, “Adaptive Clustering for Mobile Wireless Networks,” *IEEE Journal on Selected Areas in Communications*, vol. 15, N°. 7, pp. 1265-1275, 1997.

[20] V. Mhatre and C. Rosenberg, “Homogeneous vs Heterogeneous Clustered Networks: A Comparative Study,” in *Proceedings of IEEE ICC 2004*, June 2004.

[21] V. Mhatre and C. Rosenberg, “Design guidelines for wireless sensor networks: communication, clustering and aggregation,” *Ad Hoc Network Journal*, vol. 2 N°. 1, pp. 45-63, 2004.

[22] W. Navidi and T. Camp,”Stationary distributions for the random waypoint mobility model,” *IEEE Transactions on Mobile Computing*, vol. 3 N°.1, 2004.

[23] M. Ye, C. Li, G. Chen, and J. Wu, “EECS: an energy efficient cluster scheme in wireless sensor networks,” in *IEEE International Workshop on Strategies for Energy Efficiency in Ad Hoc and Sensor Networks (IEEE IWSEEASN2005)*, Phoenix, Arizona, April 2005.

[24] O. Younis and S. Fahmy, “Distributed clustering in ad-hoc sensor networks: A hybrid, energy efficient solution,” in *Proceedings of IEEE INFOCOM*, Hong Kong, March 2004.

[25] http://www.isi.edu/nsnam /ns/, “The network simulator - ns-2,” 2002.

First M. LEHSAINI received his B.S and M.S degrees in Computer Engineering from Oran University of Sciences and Technology (USTO), Algeria in 1992 and 2000, respectively. In 2000, he joined Tlemcen University where he worked as an assistant professor in Computer Science and Engineering. He is a member of SDR “Networks and Distributed Systems" group at LIFC ” Laboratory of Computer Engineering of Franche-Comté University" and also a member of STIC Laboratory at Tlemcen University. His research interests include Wireless Sensor Networks and discovery of services.

Second M. FEHAM received his Dr. Eng. degree in optical and microwave communication from the University of Limoges (France) in 1987, and his PhD in Science from the University of Tlemcen (Algeria) in 1996. Since 1987, he has been an Assistant Professor and Professor of microwave and communication engineering. He has served on the Scientific Council and other committees of the Electronics and Telecommunication Departments of the University of Tlemcen. His research interests are mobile networks and services.