A Distributed Clustering Algorithm without an Explicit Neighbourhood Knowledge

Abdelfettah Belghith[1], Imen Jemili[1,2] and Mohamed Mosbah[2]
[1] RIM Research Group CRISTAL Laboratory,
University of Manouba, Tunisia 2010, Campus universitaire de la Manouba, Manouba 2010, Tunisie
imen.jemili@cristal.rnu.tn, abdelfattah.belghith@ensi.rnu.tn
[2] LaBRI Laboratory,
University of Bordeaux I, French 33405, 351 cours de la Libération, Talence, France
[jemili, mosbah]@labri.fr

Abstract: *Ad-hoc networks have received significant attention over the last few years as these emerging networks provide a fast deployable multi-hop wireless infrastructure for a growing number of applications when a wireline network is neither available nor economical to build. To overcome the scalability problem, creating hierarchies among the nodes seems to be a promising approach and an effective way to organize the network as the number of nodes increases. In this paper, we present a novel distributed clustering algorithm for ad-hoc networks. Our algorithm is based on a synchronized and layered process. Its main objective is to minimize the number of control messages exchanged during the clustering process.We evaluate our algorithm using the Visidia simulation test bed and show that it outperforms one of the most known clustering algorithms in minimizing both the number of control messages and the number of elected clusterheads.*

Keywords: *Ad-hoc networks, clustering, synchronization, IEEE 802.11, PSM, routing.*

Received: May 1, 2006 | ***Revised:*** June 30, 2006 | ***Accepted:*** September 30, 2006

1. Introduction

Since few years, ad-hoc networks have represented a dynamic field which has knowing an ever growing interest thanks to their rapid deployment and dynamic reconfiguration. Ad-hoc networks offer an adequate mean for ground communications and information access where wire line networks are neither available nor economical to build. This field constitutes a hot research area concerning several topics related to the various problems confronting the ad-hoc network community such as routing, quality of service, energy consumption, security, clustering and synchronization [1]. In this paper, we describe and evaluate the performance of a novel distributed clustering algorithm based on a synchronized layering mechanism that deploys a limited number of control messages, yet provides a reasonable number of elected clusterheads within a bounded time.

The basic idea behind clustering is to aggregate nodes into groups or clusters controlled by elected nodes called clusterheads. These clusterheads act as local coordinators of their node members for different activities such as channel scheduling, routing, energy conservation and quality of service management. Nodes in the vicinity of just one clusterhead are termed as ordinary member nodes (or just members), while those directly linked to more than one clusterhead are useful to relay data traffic between adjacent clusters and if elected they are termed gateways.

A clustering algorithm aims to find a feasible interconnected set of clusters covering the entire node population where clusterheads form a virtual backbone in order to overcome the scalability problem and to assure efficient utilization of radio channel resources [2, 3]. However, we must keep in mind that nodes mobility may impose frequent updates on the clustering structure in order to maintain an accurate network topology. An efficient clustering algorithm should be able to select the appropriate nodes to serve as clusterheads. The number of these clusterheads should be minimized in order to adequately fulfill the objective. The clustering algorithm should be robust against topological changes by properly choosing the correct clusterheads, yet it should keep at minimum the overhead generated in order to re-cluster the net and

consequently does not degrade the system performance or the performance of others algorithms based on the built clustering structure.

In this paper, we present a new distributed algorithm for clustering nodes in ad-hoc networks that does not require any neighborhood knowledge. Our main motivation is to reduce the complexity of the clustering process in terms of the number of control messages required to establish, maintain and update the clustered network topology. To the best of our knowledge, virtually in all previous clustering algorithms, each node needs to discover explicitly the nodes in its vicinity to be able to decide locally whether to act as a clusterhead or not. In our proposed approach, we do not require nor assume this neighborhood knowledge which leads to a substantial gain in control traffic. We evaluate our algorithm using the software platform for the Visualization and the Simulation of Distributed Algorithms: ViSiDiA [17, 18].

The paper is organized as follow. Section 2 reviews recent clustering algorithms. Node synchronization in ad-hoc networks is described briefly in section 3. We expose our proposed algorithm for clustering in ad-hoc networks in section 4. We present several comparisons in order to evaluate our algorithm in section 5. Finally, we conclude in section 6 with a summary of the contribution of this paper and directions for future work.

2. Related work

Many clustering algorithms were proposed in the literature. We can classify these algorithms in different manners either based on the connectivity of clusterheads or based on the criterion used for the selection of the clusterhead or even based on the distance separating a clusterhead to any of its nodes members.

Based on the criterion for the clusterhead connectivity, we distinguish two different categories: the connected clustering which is based on the notion of dominating connected sets and the non connected clustering where clusterheads communicate through intermediate gateway nodes. In the former, a set of nodes (clusterheads) is called dominating if all the nodes in the system are either in the set or neighbors of nodes in the set. This approach was frequently used for routing in order to reduce the searching space for a route to nodes in the set.

Wu and Li presented a simple distributed algorithm for calculating small connected dominating sets (CDS) in ad-hoc wireless networks [4]. The algorithm first finds a CDS and then prunes certain redundant nodes from it. The initial CDS consists of all nodes which have at least two non-adjacent neighbors. The algorithm then removes all locally redundant nodes by the application of two rules. According to dominant pruning rule 1, a marked host can unmark itself if its neighbor set is covered by another marked host. According to Rule 2, a marked host can unmark itself if its neighborhood is covered by two other directly connected marked hosts. The marking process along with Rules 1 and 2 are purely localized algorithms. The algorithm needs at least two-hop neighborhood information and the status of each host is assigned based on the connectivity of its neighbors. This approach was first proposed for undirected graphs using the notion of dominating set only and was later extended to cover directed graphs by introducing another notion called absorbent sets [5].

Dai and Wu [6] proposed several enhancements to the definition of internal nodes. In [6], they generalize the one and two neighbor coverage of a node to k- neighbour coverage for both fixed and variable k. The case of a variable k is even computationally less expensive than the two nodes coverage case. All of these schemes, however, depend on periodic hello messages to collect neighborhood topological information.

We will rather focus our interest on the second category of clustering, namely the nonconnected clustering. Here the clustering problem pertains to classifying nodes hierarchically into equivalence classes according to certain attributes. Many criteria were proposed and used for the election of clusterheads. [7], [8], and [10] are based on the unique identifier associated to each node. In [7], the authors presented two distributed algorithms. The first one is the lowest-ID algorithm. As the name indicates, the elected node has the lowest ID among its neighborhood. A similar approach is adopted in [8], the lowest ID node is selected as clusterhead to form a cluster with its one hop neighbors, thus the distributed algorithm is initiated by all nodes whose IDs are lowest among all their one hop neighbors. Each node broadcasts its clustering decision exactly once. The neighbors will select the lowest ID node as clusterhead. Also in [10], nodes are candidates to be clusterheads based on their node IDs.

The second algorithm described in [7] proposed to elect instead the highest degree node in a neighborhood as a clusterhead. The authors in [11] proposed to select clusterheads based on the connectivity as the primary criterion and the lowest ID as the second one to just break ties. In [12], the authors presented a distributed label clustering scheme that partitions nodes into clusters and uses a weight-based criterion in order to identify the border nodes. The weight function is defined as the sum of the reciprocal of the neighbors' degree in which the ratio of the node identity and the number of nodes in the network is used as second item in order to break the symmetry. The proposed mechanism allows border nodes to determine their roles.

On the other hand, authors in [9] proposed a generalized approach by allowing the choice of the clusterheads based on a generic weight associated with each node. The main idea is that a node determines its role only when all its neighbors with bigger weights have decided their own roles, so it must wait until these neighbors decide which role to assume. The benefit in [9] is that any meaningful measure can be used for the clusterhead selection and it is possible to use node mobility-related parameters. In [13], the authors presented a combined weight which takes into consideration several criteria such as the ideal degree, the transmission power, the mobility and the battery power of the mobile node. They also considered the number of nodes a clusterhead can ideally handle by keeping this number around a predefined threshold to facilitate the optimal operation of the medium access protocol. The election procedure proposed in [13] is invoked as rarely as possible in order to lower the computation and communication costs associated with it. In [14], a new clustering scheme named Availability Clustering (AC) is proposed taking into consideration the possibility for nodes to be heterogeneous in terms of connectivity, available energy and relative mobility.

All above described algorithms require at least the immediate neighborhood knowledge in order to permit to every node to decide locally about the role to ensure. In large and dense networks, exchanging control messages to collect neighborhood information may on the contrary degrade considerably the network performances.

3. Node synchronisation

Our proposed clustering algorithm assumes that all nodes are synchronized. Synchronization in an ad-hoc network is for example defined in the IEEE 802.11 standard [19] in the context of the power saving mechanism (IEEE 802.11 PSM). It is based on a distributed Timing Synchronization Function (TSF) that keeps the timers of all stations of the Independent Basic Service Set (IBSS) synchronized. Time is divided into beacon intervals [19]. A beacon interval is divided into two subintervals: the Ad-hoc Traffic Indication Message window subinterval (called the ATIM window) followed by the rest of the beacon interval. A series of TBTTs (Theoretical Beacon Transmission Time) are then defined which are exactly a beacon interval time units apart. In PSM, a node can either be awake (in the awake state) or dozing (in the doze state). A node in the awake state can transmit and receive frames. In the doze state, the node radio is turned off and therefore the node can neither transmit nor receive frames. In PSM, when a node has a packet destined for another node, this packet is announced during the subsequent ATIM window using an “ATIM frame”. All nodes stay awake during the ATIM windows. A node that sends an ATIM frame stays awake for the rest of the current beacon interval. A node that receives an ATIM frame replies by sending an ATIM ACK and remains in the awake state for the rest of the current beacon interval.

Now the question naturally arises as to how we accomplish the time synchronization of all nodes; and hence define the series of TBTT. Each node maintains a local TSF timer. Nodes expect to receive beacons every beacon interval. A node sending a beacon sets this beacon timestamp to its own local TSF time. The beacon interval of the IBSS is established by the station that initiated the IBSS. A node shall transmit a beacon according to the following procedure:

- Each node awakes just before the end of the TBTT to be able to receive Beacons. If the node is already in the awake state, it suspends decrementing the back off timer for any pending non beacon or non ATIM transmission.
- Each node calculates a random delay upon which it transmits a beacon.
- Each node decrements its random delay according to the defined back off algorithm.
- Each node sends its own beacon if no beacon was received and its random delay has expired.
- Each node suspends decrementing its random delay and renouncing on its beacon transmission if a beacon was received before the random delay has expired.
- Each node adopts the timing received from any beacon that has a TSF value later than its own TSF timer.

We notice that the above described node synchronization enables indeed a certain cluster formation. In fact, nodes receiving a beacon will follow and adopt the timestamp of the sender. So, if we consider the beacon sender as a virtual clusterhead and the nodes adopting the TSF value of this clusterhead as its members, we find that we have a virtual clustering. However, the TSF function presents certain drawbacks. For example, in case of beacon collisions, synchronization fails in certain parts of the network.

The virtual clustering was already adopted by a MAC protocol [15] designed for wireless sensor networks in order to reduce energy consumption and support self-configuration. To reduce energy consumption in listening to an idle channel, [15] allows nodes to periodically sleep. Neighbouring nodes form virtual clusters to auto-synchronize on sleep schedules. Thus, the virtual clustering urges nodes to form clusters with the same schedule, without enforcing this schedule to all nodes in the network. It allows efficient broadcast, and obviates the need to maintain information on individual neighbours. In [16], the authors also used this synchronization scheme. They

allowed nodes to transmit data traffic only during an active period. So, nodes with the same schedule will be active simultaneously forming a virtual cluster.

The beaconing protocols just described divides time into the so called beacon intervals. In this paper we are not considering the problem of synchronizing the nodes; but rather assume that a certain mechanism exists for this purpose and not necessarily the IEEE 802.11 beaconing protocol. More specifically we just assume that the time axis is indeed divided into a series of TBTTs which are exactly a beacon interval time units apart.

4. Proposed clustering mechanism

The goal of clustering is to reduce route discovery overhead and to optimize the utilization of the different resources like battery power and network capacity. We recall that in the existent clustering approaches, each node needs to know its immediate neighborhood or even its two hops neighborhood (for some protocols) to be able to decide whether or not to be a clusterhead. To collect this information nodes exchange the so called hello and other control messages. This constitutes indeed a large amount of control traffic that deteriorates the network capacity and penalizes the flowing of normal data traffic.

We propose a novel clustering mechanism. The main contribution is that our mechanism does not necessitate any type of explicit neighborhood knowledge. Consequently more bandwidth is left for transporting data traffic. The basic idea is to organize the clustering process into layered stages and to inform and allow only the two hop neighbors of a declared clusterhead to compete locally to become a clusterhead. The clustering process is initiated by a specific node called the Designated Node (DN). This node is the first node that formed the IBSS.

At the beginning of every synchronization period (i.e., beacon period); that is at exactly every TBTT, the node DN sends a cbeacon (clustering beacon) packet after a sensing period equals to a SIFS (Short Interframe Space time) or a DIFS (Distributed Interframe Space time). We adopt here the inter frame spacing SIFS for no power saving mechanism is considered. Otherwise, the period DIFS will be used. How to use conjointly a power saving mechanism is out of the scope of this paper.

We assume that we have a unique IBSS and a unique node DN in our IBSS. The existence of several DN nodes in an IBSS and the delegation of the DN role will be treated in a future work.

4.1 Clusterhead and member declaration

At the beginning of the clustering process nodes are marked unaffected. At the termination of the clustering process each node will be marked either a clusterhead, a gateway or a member of one or more clusterheads. The DN is the first declared clusterhead and remains a clusterhead forever. We recall here that no treatement of DN role delegation is assumed in this paper. The DN triggers the clustering process by sending a cbeacon message exactly at TBTT. Its immediate neighbors will join its cluster (they become members of the DN cluster) and reply after a SIFS by an ACK. All these neighbors send their ACK at the same time generating hence a sure collision, considered like a 'Busy Tone'. The busy tone will be intercepted by the two hop neighbours of the DN informing them that they are able to compete to become clusterheads. The clustering process will progress layer by layer. A node which has heard a cbeacon message is not allowed to participate into the clusterhead election, whereas a node which intercepts a BT is eligible to become a clusterhead. The eligible nodes can compete to gain the clusterhead status in a randomized manner. Each eligible node initializes its back off timer with a random delay to be decremented according to the defined CSMA/CA back off algorithm. At the expiration of its back off timer, an eligible node declares itself a clusterhead by sending a cbeacon. An eligible node gives up its competition to be a clusterhead immediately upon the reception of a cbeacon message from a neighbour. In this way only a limited subset of the eligible nodes becomes clusterheads.

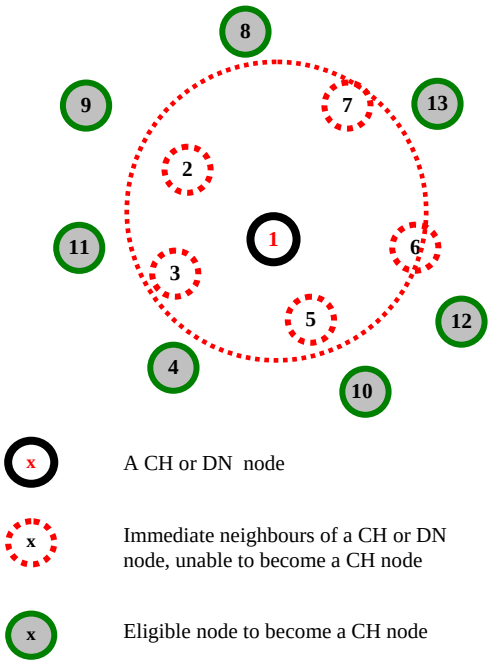

Figure 1. The clustering propagation

Our aim is to realize the clustering process in a layered manner. As portrayed in Figure 1 above, the DN (node 1) and its immediate neighbors (nodes 2, 3, 5, 6 and 7) constitute the first layer. When receiving the DN cbeacon, these neighbors will be aware that they are not allowed to contend to be clusterheads. They declare themselves members of the DN cluster. They wait for a SIFS before sending an ACK which

will be perceived like a busy tone by the two hop neighbours (nodes 4, 8, 9, 10, 11, 12 and 13) notifying them that they are eligible to be elected as clusterheads. So the two hop neighbours of the DN node and their neighbors which have not yet participated in the clustering process form the second layer. Each eligible node waits for a SIFS and then calculates a backoff time in order to defer broadcasting to become a clusterhead. We note here that the backoff time may take into account various parameters such as the node energy level, the node mobility and so on. Different criteria could be envisioned to properly choose clusterheads. In this paper, we do not assume any preference. An eligible node sends its cbeacon, and consequently declares itself a clusterhead, when its backoff timer expires and if it has not already heard another neighbor cbeacon. Its immediate neighbors in the same layer (that is its neighbors which have not already send a BT) wait a SIFS and then reply with the BT in order to trigger the clustering process at the next layer. This, however, will eventually create collisions between cbeacon and BT messages. A node hearing a collision will not be able to differentiate between collisions which happened between cbeacon messages from collisions which happened between a cbeacon and BT messages.

To resolve this problem, we will impose the termination of the clustering process in a specific layer before allowing nodes in the subsequent layer to start the clustering process. We force the layering to be centered at the DN. We require that all eligible nodes belonging to the same layer have been declared either as clusterheads or members before authorizing those of the following layer to begin their clustering process. Furthermore, nodes in the current layer that have heard cbeacon(s) transmit their ACK at the same time upon the termination of the election process at this layer.

At every TBTT, every node regards its time axis as divided in a succession of periods of time denoted by T_i, i=1,…,N where N represents the number of layers needed to cover the entire network. Depending on the context, we consider that T_i, i=1,…,N denote also the instant at which period Ti ends. Period $T_i$defines the time duration required by the execution of the clustering process for layer i. It includes a sub period for the announcements of cbeacon messages followed by a second sub period for the simultaneous transmission of BT; as shown in Figure 2. The first sub period comprises the sensing period SIFS, the maximum back off time that can be taken for sending a cbeacon message and the time spent for a cbeacon transmission. The second sub period is dedicated to the BT transmission and thus it starts upon the end of the first sub period. We denote by TiBT the instant at which this second sub period starts. All nodes, having heard one or several cbeacon or intercepted a collision during the first period of T_i, are authorised to send their unique BT at this specific predetermined instant. In this manner, we avoid the collisions between cbeacons and BT messages. Moreover, every node will be able to differentiate between the collisions occurring among cbeacons messages (only possible during the first sub period) and the collisions between BT messages(happening during the second sub period).

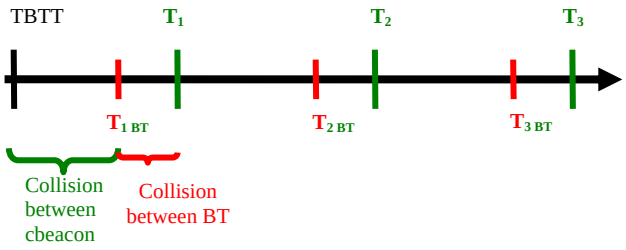

Figure 2. The time division for the clustering process

If a collision occurs during the first sub period of T_i, then it is systematically considered as a collision between two or more cbeacons. This can happen when more than one node transmits its cbeacon at the same time.

The simultaneous sending of BT message generates a collision which aims to notify nodes in the following layer that they can start their clustering process.

Let us now express explicitly the T_i, i=1,…,N. We define the following quantities:

Txbeacon : the transmission time of a beacon
TxBT : the transmission time of a busy tone
SlotTime : the slot time
SIFS : Short Interframe Space time
DIFS : Distributed Interframe Space time
CWsize : the contention window size times the SlotTime

We get the following expressions for T_i for i=1,…,N.

T_1= (D)SIFS+Txcbeacon+SIFS+TxBT
T_2= T_1 + SIFS+CWsize+Txcbeacon +SIFS +TxBT
T_3= T_2 + SIFS +CWsize+ Txcbeacon +SIFS + TxBT
…
T_N= T_1 + (N-1) * (SIFS + CWsize + Txcbeacon +SIFS + TxBT)

and the following expressions for $T_{i\,BT}$ i=1,…,N.

$T_{1\,BT}$= SIFS+Txcbeacon
$T_{2\,BT}$ = T_1 + SIFS+CWsize+Txcbeacon
$T_{3\,BT}$ = T_2 + SIFS+CWsize+Txcbeacon
…
$T_{N\,BT}$ = T_{N-1} + SIFS+CWsize+Txcbeacon

We note that upon the first reception of a cbeacon or a BT, a node other than the DN realizes that the current clustering process concerns its layer. Consequently, the series T_i and T_{iBT} allow each node to determine how many hops it is far away from the DN node. We should also note that each node knows

its serving clusterheads since cbeacons contain the identity of the sending clusterhead.

4.2 Gateway declaration

During the clustering process, a node that hears a correct cbeacon message modifies its status to a Member node (M) and adds the sender node id (the clusterhead id) to its clusterhead list (CH list). An M node that receives two or more correct cbeacon messages becomes a candidate gateway (candGW). This node is then eligible to ensure the role of a gateway between its clusterhead neighbors. Figure 3 below exhibits this case. Node 6 has heard a correct cbeacon from node (clusterhead) 1 and then another correct cbeacon from node (clusterhead) 12 and consequently becomes a candGW for these two clusterheads.

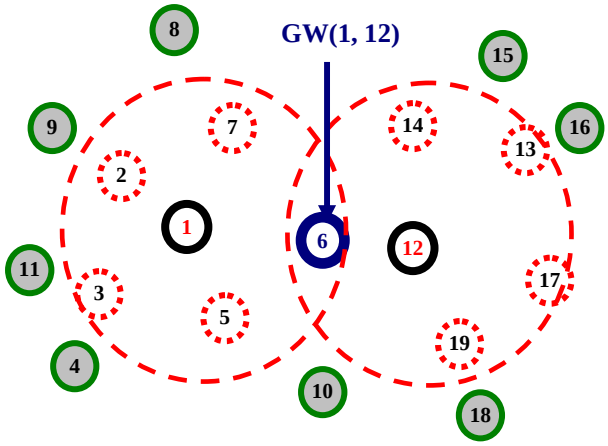

Figure 3. A normal gateway declaration

Moreover, a node intercepting a collision during the first sub period of its T_i becomes also a candGW for this collision is normally caused by a simultaneous transmission of two or more cbeacons from eligible neighbors. However the node, in this case, adds an anonymous clusterhead to its CH list to represent the existence of two or more unknown clusterheads. Figure 4 below exhibits this case. Node 6 has received a correct cbeacon from node 1 during the execution of the clustering process in the first layer (we assume here that node 1 is the DN). Node 6 gets then status M. Upon the execution of the clustering process of the second layer, it hears a collision during the first sub period of T_2.This is the simultaneous transmission of node 14 cbeacon and node12 cbeacon. Node 6 gets then the status of a candGW. Its CH list contains as clusterheads now node 1 and an anonymous one denoted by ?. Note that a node in layer i could receive a cbeacon collision either during the clusterhead election process of its layer (during the first sub period of T_i) or during the clusterhead election process of the subsequent layer (during the first sub period of T_{i+1}).

A candGW node must postpone announcing (transmitting) its declaration to be an effective gateway for a certain time in order to avoid collisions with the on going clustering traffic. A candGW node within layer i must then wait a period of time denoted by TG_i in such a way to not perturb the clustering process traffic of the current and following layers (layers i and i+1). In fact, nodes belonging to layer i are currently competing to become clusterheads and our candGW should not prevent them from correctly receiving or sending their cbeacon messages. Nodes in layer i+1 will subsequently compete to become clusterheads and no collisions with other traffic is permitted.

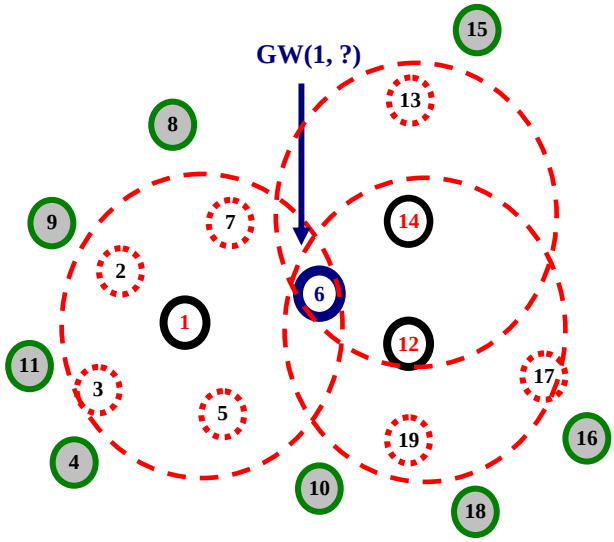

Figure 4. The case of an anonymous clusterhead caused by collision between cbeacon sent by nodes 12 and 14 simultaneously

To this end, we get the following expression for TG_i :

$$TG_i = T_{i+2} - t \quad i=1,\ldots,N$$

where t denotes the current time.

After waiting the TGi period, a candGW chooses a random backoff delay. Upon the expiration of this back off delay, the candGW transmits its declaration to be an effective gateway for the clusterheads in its CH list. A candGW stops its back off timer and renounces to be a gateway if it hears another candGW declaring itself for a set of clusterheads that covers its own. The random back off delay may take into consideration different criteria such as the total number of neighboring clusterheads (the length of its CH list); the degree of mobility and the remaining energy of the node or any combination or function of these. For the purpose of this paper we do not assume any criterion.

4.3 A comprehensive example

We are now ready to illustrate our proposed clustering mechanism through a complete example. Figure 5 below shows the topology adopted. We put an edge between any two nodes able to communicate directly (within range of each other). Node 0

represents the 'Designated Node' which will initiate the clustering process.

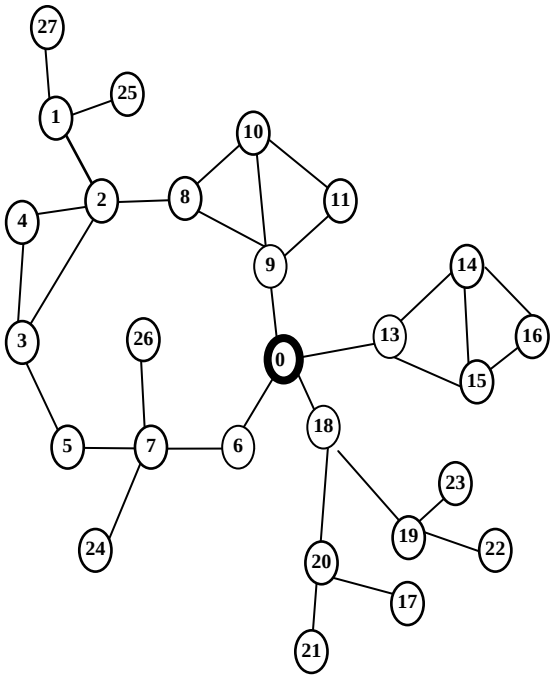

Figure 5. The example topology

The DN declares itself a clusterhead and triggers the clustering process by sending the first cbeacon. Its immediate neighbours, nodes 6, 9, 13 and 18 will accept to be members of this clusterhead and thus they all get status M. As shown in Figure 6 below, the nodes 6, 9, 13 and 18 become dotted. They reply by sending simultaneously a BT at the T_{1BT} instant. This terminates the layer 1 clustering process.

Now, the layer 2 clustering process starts. Nodes 7, 8, 10, 11, 14, 15, 19 and 20 hear the BT and therefore become eligible to compete to be clusterheads. These eligible nodes become grayish on Figure 6. Each one of these eligible nodes calculates a random back off delay for the transmission of its own cbeacon.

During the first sub period of this second layer, we suppose that node 7 has got the shortest back off time and therefore transmits its beacon the first. It becomes a clusterhead. Consequently, nodes 5, 24 and 26 being neighbors of nodes 7, hear this cbeacon and thus get status M. Node 6 is also a neighbor of node 7. It correctly receives this beacon, however since it has already had received a cbeacon in the first process layer, it becomes rather a candGW. Next, node 10 transmits its beacon and becomes a clusterhead. Nodes 8 and 11 being neighbors of node 10 get status M. Node 9 also is a neighbor of node 10. Since it is an M node, it becomes candGW. Now, node 14 back off timer expires and consequently becomes a clusterhead and transmits a cbeacon. Its neighboring nodes 22 and 23 become M and its neighbor 18 becomes candGW. Now node 14 transmits a cbeacon and becomes a clusterhead. Nodes 15 and 16 become M and node 13 becomes candGW. Finally node 20 back off timer expires and consequently node 20 becomes a clusterhead and sends a cbeacon. Neighboring nodes 17 and 21 become M and neighboring node 18 becomes a candGW.

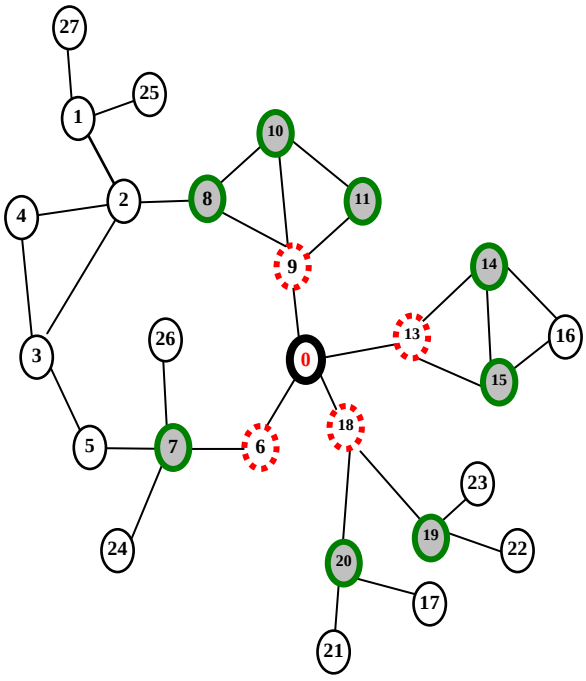

Figure 6. The DN declaration

Now all eligible nodes, grayish nodes on Figure 6, are declared either clusterheads or members M. This ends then the clustering competition at layer 2 (the first sub period of layer 2). These nodes are represented in Figure 7 below either in dotted for member nodes or in black for clusterheads. Nodes in dotted on Figure 7, that is nodes 24, 5, 26, 8, 11, 16, 15, 23, 22, 17 and 21, wait for instant T_{2BT} to send simultaneously a BT. Nodes 6, 9, 13 and 18 are candidate gateways and they do not participate in sending the BT. We recall that each node is allowed to send a BT exactly once. These nodes are shown in blue on Figure 7.

Exactly at T_{2BT} nodes declared as members in layer 2 transmit a BT. This ends the clustering process of layer 2. The nodes that receive this BT are node 2 and node 3 indicated by grayish nodes on Figure 7 below. Now starts the layer 3 clustering process. Nodes 2 and 3 are eligible to compete for the clusterhead role. We suppose, as portrayed on Figure 8 below, that node 3 back off timer expires first. Consequently node 3 becomes a clusterhead and transmits a cbeacon. It is represented in black on Figure 8 below. Node 2 being a neighbour to node 3 renounces to become a clusterhead and gets the M status. Node 4 receives node 3 cbeacon and consequently becomes an M node. Node 5 however is already in status M therefore it becomes a candGW. It is represented in blue on Figure 8 and nodes 2 and 4 are indicated in dotted. This terminates the clustering competition at layer 3. Exactly at T_{3BT} nodes 2 and 4 transmit

simultaneously their BT. This ends the layer 3 clustering process.

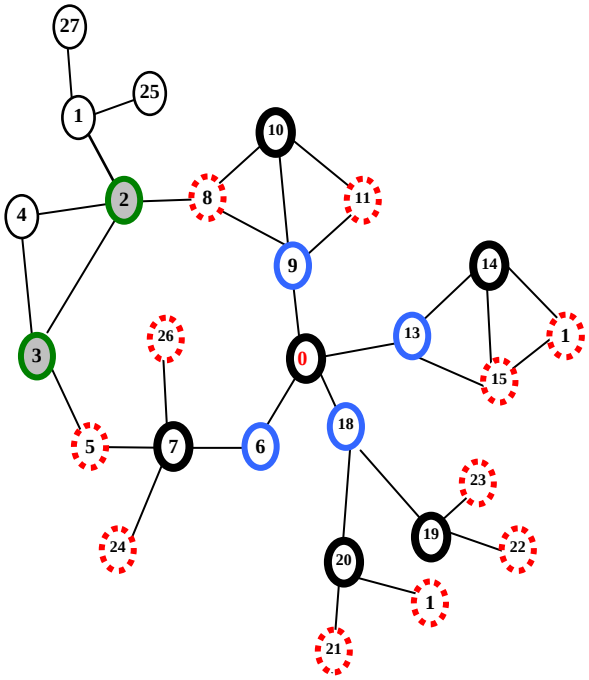

Figure 7. The second layer clustering

Now starts the layer 4 clustering process. Node 1, indicated by grayish nodes on Figure 8, receives the BT and therefore becomes eligible to be a clusterhead. Note that node 8 receives also this BT however it is already in status M and consequently remains in that status. Upon the expiration of its back off timer, node 1 becomes a clusterhaed and sends a cbeacon. It is represented in black on Figure 9 below. Nodes 25 and 27 receive this cbeacon and then become members indicated in dotted on Figure 9. Node 2 receives also this cbeacon however since it is in status M it becomes a candGW. Node 2 is represented in blue in Figure 9.

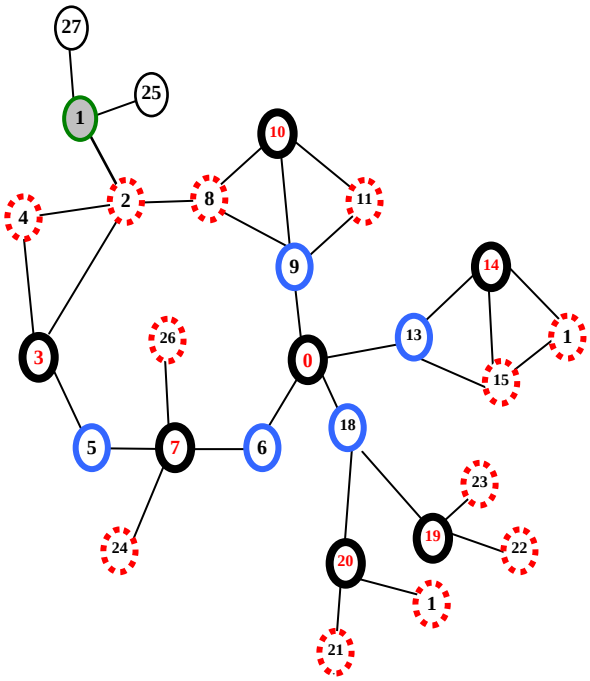

Figure 8. The clustering of the remaining layers

This ends the clustering competition in layer 4. Exactly at T_{4BT} nodes 25 and 27 transmit simultaneously their BT. This terminates the layer 4 clustering process. Now all nodes are declared and consequently the whole clustering process comes to an end.

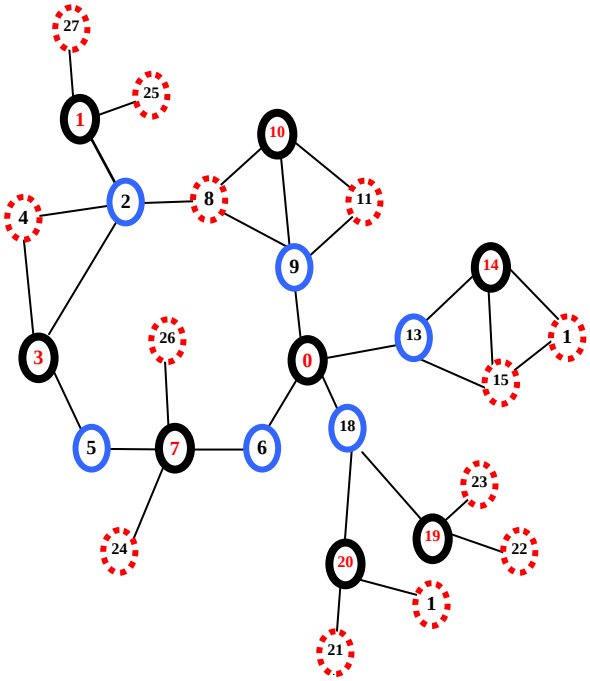

Figure 9. The clustering of the remaining layers

5. Mechanism evaluation

Clustering algorithms can be compared using different performance criteria. We believe that the control traffic load induced by a clustering algorithm constitutes one of the most important performance measures. Indeed in ad hoc networks, the bandwidth of the wireless medium is the scarcest resource and hence should be carefully managed. Clustering algorithms are usually intended to help in routing, power conservation and quality of service management among other things and provide a solution to the scalability problem of very large ad hoc networks. Consequently a good algorithm is one that induces the least control traffic possible, yet the number of control packets should not be directly related to the number of nodes in the network. We will show that the control traffic load of our proposed algorithm depends only on the diameter of the network independently of the total number of nodes. Virtually all other known algorithms do depend on the number of nodes forming the ad hoc network. A second performance criterion is the number of declared clusterheads. This is one of the most used performance criterion for clustering algorithms. This number should be as low as possible for the clustering algorithm to fulfill its objectives. In this paper, we limit ourselves to these two performance measures and show that our proposed algorithm outperforms nicely one of the most reputed algorithms, namely WCA [13].

Now, we focus on the number of elected clusterheads which is portrayed in Figure 12 below as a function of the number of nodes composing the network. This Figure shows that our mechanism provides the election of fewer clusterheads than WCA algorithm.

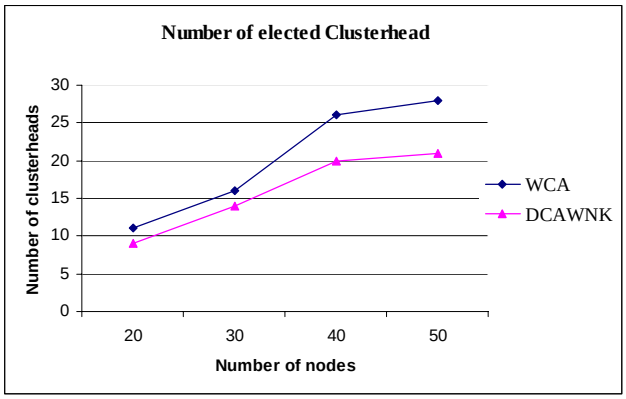

Figure 13. The average number of elected clusterhead

This is especially due to the use of the simultaneous transmission of the busy tones and the layered actions of the clustering process. Moreover, our proposed clustering algorithm inherently avoids electing clusterheads which are neighbors.

6. Conclusion and future work

In this paper, we presented a new clustering mechanism for ad hoc networks. Our primary objective was to alleviate to network from the additional messages exchanged during the neighbourhood discovery phase in clustering algorithms. Our proposed clustering algorithm does not rely on exchanging any neighbourhood information.

Performance evaluations in terms of the total number of induced control messages and the number of declared clusterheads are performed using the ViSiDia software tool for execution of distributed algorithms. We showed that our proposed algorithm nicely outperforms one of the most reputed clustering algorithms. Further evaluations and functional extensions are currently being investigated to better ascertain the behaviour of the proposed algorithm and compared it to other clustering schemes.

References

[1] Chlamtac, I., Conti, M., Lui, J.N., “Mobile Ad Hoc Networking: Imperatives and Challenges”; Elsevier Ad Hoc Networks Journal, vol. A, no. 1, pp. 13-64, July 2003.

[2] Chen, Y.P., Liestman, A. L., Liu, J., “Clustering Algorithms for Ad Hoc Wireless Networks”. In Ad Hoc and Sensor Networks ed. Y. Pan and Y. Xiao, Nova Science Publishers, 2004.

[3] Pei, G., Gerla, M., Hong, X., Chiang, C.C., “A Wireless Hierarchical Routing Protocol with Group Mobility”, Proc. of IEEE WCNC’99, New Orleans, LA, September 1999.

[4] Wu, J., Li, H. “On Calculating Connected Dominating Sets for Efficient Routing in Ad Hoc Wireless Networks”, Proc. 3rd Int'l. Wksp. Discrete Algorithms and Methods for Mobile Comp. and Commun. (DialM’99), pp. 7–14, 1999.

[5] Wu, J. “Extended dominating-set-based routing in ad hoc wireless networks with unidirectional links”, IEEE Transactions on Parallel and Distributed Systems, vol. 13, no. 9, pp. 866–881, Sep. 2002.

[6] Dai, F., Wu, J. “Distributed dominant pruning in ad hoc wireless networks”, Proc. IEEE Int. Conf. on Communications (ICC 2003), Anchorage, AK2003, May 2003.

[7] Gerla, M., Tsai, T. J.-C., “Multi cluster Mobile Multimedia Radio Network”, Wireless Networks, 1, pp. 255-265, 1995.

[8] Lin, C. R., Gerla, M., “Adaptive Clustering for Mobile Wireless Networks”, IEEE Journal on Selected Areas in Communications, Vol. 15, pp. 1265-1275, September 1997.

[9] Basagni, S., “Distributed Clustering for Ad Hoc Networks”, Proceedings of the 1999 International Symposium on Parallel Architectures, Algorithms, and Networks (I-SPAN'99). IEEE Computer Society Australia, pp. 310-315, 1999.

[10] Amis, A. D., Prakash, R., Vuong, T. H.-P, Huynh, D. T., “Max-Min D-Cluster Formation in Wireless Ad Hoc Networks”, Proc. IEEE Infocom 2000, pp. 32-41, 2000.

[11] Chen, G., Nocetti, F. G., Gonzolez, J. S., Stojmenovic, I., “Connectivity Based k-hop Clustering in Wireless Networks”, Proceedings of the 35th Hawaii International Conference on System Sciences (HICSS-35), January 2002.

[12] Dow C.R., Lin J.H., Hwang S.F., Wang Y.W., “An Efficient Distributed Clustering Scheme for Ad-hoc Wireless Networks”, Transactions Communications (IEICE), Hawaii 2002.

[13] Chatterjee, M., Das, S. K., Turgut, D., “WCA: a Weighted Clustering Algorithm for Mobile Ad Hoc Networks”, Cluster Computing Vol. 5, pp.193-204, 2002.

[14] Blasi, D., Cacace, V., Casone, L., Losacco, A. "Availability Clustering: a Spine-based Clustering Scheme to Exploit Nodes Heterogeneity in Ad Hoc Networks", proc. 2-nd Med-Hoc-Net 2002, Mahdia June, 2002.

[15] Ye, W., Heidenmann, J., Estrin, D. "An Energy-Efficient MAC Protocol for Wireless Sensor Networks", in *Proceedings of IEEE INFOCOM*, New York, NY, June 2002.

[16] Dam, T. V., Langendoen, K. "An Adaptive Energy Efficient MAC Protocol for Wireless Sensor Networks". Proceedings of the 1st international conference on Embedded networked sensor systems**.** Los Angeles, California, USA, pp. 171 – 180, 2003.

[17] Bauderon, M., Métivier, Y., Mosbah, M., Sellami, A., "From Local Computations to Asynchronous Message Passing Systems", Technical Report RR-1271-02, LaBRI, 2002. http://www.labri.fr/visidia/

[18] Bauderon, M., Métivier, Y., Mosbah, M., Sellami, A., "Graph Relabelling Systems: A Tool for Encoding, Proving, Studying and Visualizing Distributed Algorithms", In Getgrats Workshop (ENTCS), France, 2001.

[19] ANSI/IEEE Standard 802.11, "Wireless LAN Medium, Access Control (MAC) and Physical, Layer (PHY) Specifications", 1999.

Dr. Abdelfettah Belghith has received his Master of Science ad his PhD degrees from the University of California at Los Angeles (UCLA) respectively in 1982 and 1987. He is since 1992 a full Professor at the National School of Computer Science (ENSI), University of Mannouba, Tunisia. His research interests include computer networks, wireless networks, multimedia Internet, mobile computing, distributed algorithms, simulation and performance evaluation. He runs several projects in cooperation with other universities and/or research laboratories and institutions. He is currently the responsible of the graduate studies department and the head of the RIM research group at ENSI.

Imen Jemili had received her M.S. degree in Computer Science in 2002 from the National School of Computer Science (ENSI), University of Mannouba, Tunisia. She is currently a Ph.D. student at ENSI and member of the RIM research group at ENSI and Labri Laboratory of the University of Bordeaux I, France. Her research interests include wireless networks, power conservation in ad hoc networks, synchronization algorithms and QoS management.

Dr. Mohamed Mosbah has received his PhD degree from the university of Bordeaux 1 (France) in 1993. He has been an associate professor between 1994 and 2002. He is a full professor in computer science since 2003 at ENSEIRB (National Graduate School of Electronics, Computer Science and Telecommunications of Bordeaux), France. His research interests include distributed algorithms, mobile agent algorithms, programming environments, ad hoc networks and graph algorithms. He participated to several national and European research projects. Currently, he is the responsible of the graduate studies at the University of Bordeaux 1.