Towards the Definition of a Mobility-based Clustering Environment for MANET

Aida Ben Chehida, Ryma Abassi, Sihem Guemara El Fatmi
Higher School of Communication, Sup'Com
University of Carthage
Tunis, Tunisia
Emails : {bechehida.aida, ryma.abassi , sihem.guemara}@supcom.rnu.tn

***Abstract*— A MANET is a collection of mobile nodes cooperating using low bandwidth wireless links. Its use is increasingly growing because it can be rapidly and easily deployed, and allows users to access and manipulate data anytime and anywhere. Because of these characteristics, classical routing protocols cannot be applied in such environment. To be efficient, routing protocols in MANET should in fact manage mobility, handle nodes energy dissipation and ensure security. Paradoxically, these constraints are not really taken into account. This paper presents a first step of our global objective towards the definition of a complete and generic security environment for MANET routing protocols. Its main contribution is then to propose a mobility-based clustering algorithm for MANET constituting the basis for our security architecture. From a given configuration of mobile nodes, we focus our attention on the definition of two major phases of the clustering process life cycle: (1) the *setting up phase* where clusters are generated and cluster-heads are elected and (2) the *clusters maintenance phase* where the organization of the clusters is maintained in the presence of mobility. This phase reacts to all network topology changes that may occur in the network such as the displacement of a node, the failure of a node, or the arrival of a new node. Simulation experiments are conducted to evaluate the performance of our algorithm in terms of the number of cluster-heads, the node lifetime and the packets delivery ratio.**

Keywords-MANET; clustering algorithm; energy; mobility.

I. INTRODUCTION

A Mobile Ad hoc NETwork (MANET) is a set of battery-powered mobile nodes connected by low bandwidth wireless link and cooperating with each other to route packets [1]. Because routing process is essential for a successful MANET deployment, efficient routing protocols are necessary. Unfortunately, because nodes are mobile and behave simultaneously as hosts and as routers, classical routing protocols cannot be applied to this type of networks. This is why some specific routing protocols have been proposed for MANET. The main disadvantage of the majority of these protocols is that they lose their efficiency when the network size is large or the mobility of nodes is high [1]. One way to overcome this problem is to organize a MANET into clusters. Clustering in MANET is used to organize nodes into groups (clusters) characterized by cluster-head (CH) and member nodes [2]. This organization minimizes the amount of storage for communication information, makes the routing process easier, optimizes the use of the network bandwidth, etc. Clustering is generally deployed using two phases: setting up and maintenance. In the first phase, some nodes are chosen to act as coordinators (CHs) and each CH is associated to some members' nodes, the whole making one cluster. CHs are responsible for coordination among the nodes within their clusters (intra-cluster coordination) as well as communicating with other CHs (inter-cluster communication). Because the network topology changes over time (displacement, failure, arrival or departure of a node), a clustering maintenance is required to update the cluster organization.

In this paper, we propose a novel clustering approach for MANET taking into account nodes mobility. This proposition is the first step towards the definition of a complete and generic security environment for MANET routing protocols. This security environment that will be based on a Watchdog mechanism imposes to cluster members' nodes to be 'one-hop neighbors'. In our context, this constraint improves the intra-communication quality due to the interference and reduces the energy consumption. In the initial phase of the algorithm, a cluster is composed of one-hop members and its CH is the node having the smaller weight according to the mobility and energy parameters. The maintenance phase is performed according to the several kinds of changes that may happen i.e. link failure and/or new link appearance. The remaining part of this paper is structured as follows. Section 2 reviews some most related clustering protocols. Section 3 introduces the proposed clustering algorithm. Section 4 focuses on the clustering maintenance phase. Simulation results are presented in Section 5. Finally, Section 6 concludes this paper.

II. RELATED WORKS

In the literature, several clustering algorithms were proposed for MANET. These algorithms can be classified into proactive (most commonly used) and reactive ones [4].

Proactive algorithms differ from each other by the CH selection criteria in the setting up phase. In the lowest-ID algorithm [5], the node with the lowest ID within its closed neighborhood is selected as a CH. The cluster is then formed by the selected CH and all its neighbors. This criterion might be an inconvenient since every time node IDs are reshuffled, the neighboring list of all the nodes needs to be changed. In the CONnectivity-based clustering algorithm CON [6], nodes broadcast their identifier (ID) and according to the number of received IDs, every node computes its degree. The node with the highest degree is selected as CH. The major drawback of this algorithm is that the node degree may change very

 ISBN: 978-1-61208-284-4

frequently which, may inhibit CHs to play their role for very long. In the previous algorithms, the election of CHs does not take into consideration the node's energy, which may lead to battery drainage. This is all the more unacceptable, given that CHs battery drainage causes a frequent re-invocation of the clustering algorithm. The lowest MOBIlity Clustering algorithm MOBIC [7] is another proactive approach that ameliorates the previously cited works by considering mobility during CH election. However, it does not take into account the energy as an election criterion making possible a rapid failure among the CHs. Globally, proactive algorithms lead not only to useless exchanged clustering messages (high computational overhead) but also to frequent updates in the clusters structure due to the periodicity of clustering process invocation. An original optimization for the proactive clustering algorithms was proposed by M. Elhdhili et al. [4]. In this work, a local reactive update is specified following the mobility of the node and the mobility of the CH.

M. Chatterjee et al. [8] described a reactive protocol called Weighted Clustering Algorithm (WCA). In WCA, CHs are selected based on a collection of attributes: the ideal number of nodes it can support, mobility, transmission power and battery power. The node with the minimum weight is selected as CH. To maintain the cluster organization, the CH chooses new CHs for its member nodes going far from it. If a mobile node cannot reach any existing CH, it re-invokes the clustering algorithm to form new clusters. This might be an inconvenience, especially for high mobility and it generates an important computational overhead.

To our best knowledge, there is no existing work dealing with clusters with one-hop members. This characteristic is however useful for our work because the clustering process will be used as a basis for the security environment that we plan to define in a future work. For the moment, we will limit ourselves to the introduction of a mobility-based clustering algorithm for MANET. Our algorithm is designed to react to the unpredictable topology changes that may occur, i.e. link failure, new link, etc.

III. THE CLUSTERING SETTING UP PHASE

In this section, we describe the properties of the proposed mobility-based clustering algorithm and present the first phase: the setting up.

A. Basic properties

The mobility-based clustering algorithm has the following properties and/or conditions:

-It is completely distributed. Each node decides its own role (CH or member).

-It is adaptive to the changes in network, due to the node displacement, addition or failure.

-Each cluster is fully-connected in that sense that the members and the CH are one hop neighbors.

-Each node must belong to one and unique cluster.

-The clustering algorithm is based on a combined weight used in the decision of selecting a CH. The node having the lowest weight is elected as CH. The parameters of the weight are: the mobility and the residual energy of the node. The elected CH should have the lowest relative mobility and the highest remaining battery power to allow the better stability to the clusters.

- ✓ The node mobility (*M*): To calculate the metric *M* for each node, the simple heuristic mechanism MOBIC [8] is used. Once calculated, *M* is included in the HELLO message sent by each one of the cluster nodes.
- ✓ The node residual energy (*E*): *E* can be easily retrieved from the node at a given time.

The setting up phase deployment requires that each node in the network discovers its one-hop and two-hop neighbors, computes its weight and broadcasts it. This is done during the pre-processing phase.

B. Pre-processing phase

Pre-processing involves the following two steps: *neighbors discovery* and *weight computing and exchanging.*

1) Neighbor discovery

Each node *i* finds its direct neighbors DN_i by periodically broadcasting a HELLO message including its identifier (ID) as well as its mobility index M_i and by waiting for an ACK_HELLO message. Formally, DN_i (1) is the set of *i*'s neighbor, let us say *j*, such that the distance between *i* and *j* is lesser than *i*'s transmission range *TXrange (i)*. V is the set of nodes in the network.

$$DNi = \{j \in V / dist(i, j) < TXrange(i)\} \quad (1)$$

Note that ACK_HELLO message contains *j's* neighbors. These latter are stored in the neighbor table as two-hop neighbors. These messages are summarized in Table I.

2) Weight computing and exchanging

Each node *i* computes its combined weight value W_i using the average mobility metric (M_i) and the node residual energy (E_i) by the following formula.

$$W_i = w_1 (1 - E_i) + w_2 M_i \quad (2)$$

where w_1 and w_2 are the weights and $w_1+w_2=1$. Once computed, each node *i* diffuses its weight to all its one-hop neighbors using a WEIGHT message as defined in Table I.

C. Setting up phase

The setting up phase consists of two components: *cluster identification* and *cluster-head election.*

1) Cluster identification component

This component is used to generate the restricted (one-hop) neighborhood. Because this latter may not be unique, a choice can be made according to the cluster mobility in order to ensure stable clusters.

a) The identification of the Restricted neighborhood

Each node *i* computes its restricted neighborhood RN_i. Each two nodes that belong to the same *RNi* must be one-hop neighbors. Formally, *RNi* is the set of *i*'s neighbor, let us say *j*, such that for each node *k* in *RNi*, *k* and *j* are neighbors.

$$RN_i = \{j \in DN_i / k \in RN_i \rightarrow j \in DN_k\} \quad (3)$$

This identification may generate several sets of *RN*.

b) The choice of the restricted neighborhood

 ISBN: 978-1-61208-284-4

The chosen *RN* is the one having the least mobility. Knowing that each node receives the relative mobility metric *M* of its neighbors (in the HELLO message), it can calculate the mobility average of each generated *RN*. The chosen *RN* represents then the node's cluster.

Cluster identification is completed by broadcasting the RN message defined in Table I. Each node receiving this message is aware about the cluster appurtenance of its neighbors, i.e. whether a neighbor belongs to the same cluster as it.

2) Cluster-head election component

Upon receiving the weights from all *RN* members, the nodes with the lowest weight among their *RN* neighbors declare themselves as CHs by multicasting (to all *RN* members) a CH message. All *RN* neighbors of the elected CHs join them as members by broadcasting JOIN messages as defined in Table I.

TABLE I. MESSAGES EXCHANGED AND NOTATIONS: CLUSTERING SETTING-UP

Message	Meaning
HELLO (my_ID, my_M)	Notify neighbors about my ID and my relative mobility M.
ACK_HELLO (my_ID, list_my_neighbors)	Notify neighbors about my ID and my one-hop neighbors.
WEIGHT (my_ID, my_W)	Notify neighbors about my ID and weight.
RN (my_ID, my_RN)	Notify neighbors about my ID and RN.
CH (CH_ID,CH_Member)	Notify RN neighbors about my role: I am a CH, my ID is CH_ID and my members are CH_Member.
JOIN (my_ID, CH_ID)	Notify neighbors that I am going to join the cluster whose CH's ID is CH_ID.

If two nodes have the same weight, then the node with the smaller *ID* becomes the CH.

IV. THE CLUSTERING MAINTENANCE PHASE

As explained previously, the main contribution of this work concerns mobility handling in clustering environment. In this section, we introduce the proposed algorithms. Note that, when describing the procedures of our algorithm, we assume that each node has already performed the pre-processing phase, i.e., discovered its one-hop and two-hop neighbors, computed its weight and broadcasted it.

A. Initialization

Initialization is used either during the setting up phase to form clusters and to elect CHs or after the clustering setting up when a novel node joins the network or when a node is detached from its cluster. Such a node executes the procedure INIT in order to determine their own role (setting up phase) or to know in which cluster they will belong (a new arriving node or a moving node). Note that this is the first procedure that is executed by each node in the network.

Before the clustering setting up, if a node *i* joins the network, it participates to the cluster identification and CH election as presented in Section III.

After the clustering setting up, if a new arriving node *i* joins the network or a moving node *i* detaches from its cluster, it can be detected by a CH or by a member node.

If a cluster-head j detects a node i, it checks if *i* is neighbor with all its cluster members (*j* knows the *i*'s neighbors). If that is the case, *j* sends to *i* the NEW_CH message containing its *ID*, its cluster members and a flag set to 0. This flag indicates to *i* that it receives the message NEW_CH from a CH and that it is neighbor with all the *j*'s members. When receiving this message, *i* may join the *j*'s cluster by broadcasting the JOIN message. If node *j* receives the JOIN message from *i*, it adds this node into its cluster.

However, if i is not neighbor with some j's cluster members, node *j* sends to *i* the NEW_CH message with the flag set to 1. This flag will indicate to the node *i* that it receives the message NEW_CH from a CH but it is not neighbor with all the *j*'s cluster members. If node *i* sends the JOIN message to *j*, node *j* creates a new cluster with *i (*and eventually some members nodes which are neighbors with *i* in order to equilibrate the two clusters*)*, leaves its function of CH to one of its member node and informs its members by multicasting the ELECTION message. In both cases (*i* is neighbor with all *j*'s cluster members or not), *j* does not send the NEW_CH message if the number of its members is less than two.

If a member j detects the node i, it first ensures that its CH *k* is not neighbor with *i*. In such case, it sends MEMBER message to its CH *k* and waits for the CH decision. If *k* authorizes node *j* to create a cluster with *i*, node *j* sends to *i* the NEW_CH message with the flag set to 2. This flag will indicate to node *i* the reception of a NEW_CH message from a member node. If node *i* broadcasts the JOIN message, node *k* removes the node *j* from its cluster and node *j* creates a cluster with node *i*. In both cases (*i* is detected by a CH or by a member node), if node *i* detects more than one NEW_CH message, it joins the node that send the NEW_CH with flag set to 0. However, if node *i* does not detect any NEW_CH message with flag set to 0, it joins the node's cluster that send NEW_CH message with flag set to 1. Else, node *i* joins the node that send NEW_CH message with flag set to 2. If node *i* detects more than one NEW_CH message with flag set to 0, 1 or 2, the node *i* joins the cluster with the lowest CH weight. The messages exchanged are defined in Table II.

TABLE II. MESSAGES EXCHANGED AND NOTATIONS: CLUSTERING MAINTENANCE

Message	Meaning
NEW_CH(CH_ID, CH_Member, Flag)	This message is sent by a CH. It includes its ID, its cluster members and a flag.
MEMBER (my_ID, new_node_ID)	This message is sent by a member node to its CH to indicate the detection of a new node.
ELECTION (my_ID, New_CH_ID)	This message is sent by a CH. It includes its ID and the new elected cluster-head ID.

 ISBN: 978-1-61208-284-4

The following algorithm depicted in Fig. 1 illustrates the procedure INIT.

```
Procedure INIT;
Begin
        If (Receive_j NEW_CH (j, Member_j,"0"))
                Then Send JOIN (i, j);
                Else If (Receive_j NEW_CH (j, Member_j,"1"))
                        Then
                        Send JOIN (i, j);
                Else If (Receive_j NEW_CH (j, Member_j,"2"))
                        Then
                        Send JOIN (i, j);
        Else (i Form RN_i)
        If ( ∀ x∈ RN_i, w_i < w_x) Then
                Begin
                Send_RNi CH (i, Member_i);
                clusterhead_i:= i;
                End
        Else If (Receive_j CH (j))
                Then
                        If ( ∀ x∈ RN_i, w_j < w_x) Then
                        Begin
                        clusterhead_i:= j;
                        Send JOIN (i,j);
                        End
End;
```

Figure 1. Initialization algorithm

The second step concerns link failure handling. It is detailled in the following subsection.

B. Link failure handling

When a node *i* detects the failure of node *j*, there are two possibilities: (1) *j* was a cluster member and *i* its CH (2) *j* was a CH. In the first case, *j* is dropped from *i*'s cluster where in the second case, *i* compares its weight with its RN_i neighbors weights. If *i* has the lowest weight, it considers itself as CH and informs the rest of the *RN* members by sending a CH message. Else, node *i* simply waits for this message from another node. This is illustrated by the following algorithm depicted in Fig. 2.

```
Procedure Failure_Node;
Begin
        If (clusterhead_i ==i) And (j∈ cluster_i)
                Then cluster_i := cluster_i /{j};
        Else if (clusterhead_i == j)
                Then
                If ( ∀ x∈ RN_i, w_i < w_x) Then
                                Begin
                                Send_j CH (i, Member_i);
                                clusterhead_i := i;
                                End
                Else If (Receive_k CH (k, Member_k))
                                Then
                                If ( ∀ x∈ RN_i, w_k < w_x)
                                Then
                                Begin
                                clusterhead_i:= k;
                                Send_RNi JOIN (i,k);
                                End
End;
```

Figure 2. Failure node handling algorithm

C. New link handling

As detailed by the following procedure depicted in Fig. 3, when a node *j* moves inside the network (*j* is detached from its cluster), it may be detected by a CH or a member *i*. For the first case, *i* checks if *j* is neighbor with all its cluster members then unicasts a NEW_CH message such as presented previously. In the second case, i notifies to its CH *k* the existence of a new node *j* using a MEMBER message. Note that the decision of integrating this node in an existing cluster is left to the CH.

```
Procedure New_link;
Begin
                If (clusterhead_i == i) Then
                        If (j is_neighbor RN_i) Then
                        FSend_j NEW_ CH (i, Member_i,"0");
                        Else
                        Begin
                        Send_RNi ELECTION (i, k) / k □ RN_i;
                        Send_j NEW_ CH (i, Member_i,"1");
                        End
                Else
                        Begin
                        Send_k Member (i, j) / clusterhead_i == k;
                        Send_j NEW_ CH (i, Member_i,"2");
                        End
End;
```

Figure 3. New link detection algorithm

In the rest of this paper, some simulations and results are presented in order to show the feasibility of our proposition.

V. SIMULATION AND RESULTS

The aim of the following section is to study the performance of our proposition by a simulation work. Several simulations were achieved on a 100m*1000m grid while considering different network sizes varying between 10 and 40 mobile nodes, a transmission range varying between 1 and 250 m and a node speed between 500 and 3000 m/s. The simulation parameters used are listed in Table III.

TABLE III. SIMULATION PARAMETERS

Parameter	Meaning	Value
N	Number of nodes	10-40
X*Y	Network size	1000m*1000m
TR	Transmission range	1m-250m
S	Node speed	500-3000 m/s
T	Data traffic	CBR (Constant Bit Rate): data payload=512 bytes Rate= 4 packets/s
B	Node bandwidth	2 Mbps
Run Time	Time of simulation	100 sec

 ISBN: 978-1-61208-284-4

To measure the performance of the proposed clustering algorithm, we consider the following performance parameters:

- The node lifetime: the duration from the beginning until the node runs out of its battery power.
- The number of clusters, also known as number of CHs, defining the number of logical partitions formed in the network with the mobile nodes.
- The Packet Data Ratio (PDR): the ratio between the number of data packets received by the destination node and the number of data packets transmitted by the source node. It represents the reliability of packet transmissions in a network.

Fig. 4 depicts the numbers of CHs with varying number of mobile nodes in the network. Obviously, formed clusters number depends of the number of existing nodes. However, such as depicted by the figure, this dependence is linear since we constructed clusters optimally while trying to have some equality between numbers of nodes per cluster. For instance, with thirty nodes five clusters were constructed while with forty five nodes six clusters were created.

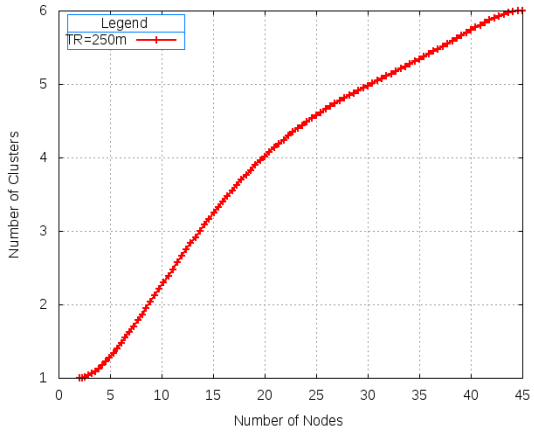

Figure 4. Number of formed clusters with varying number of nodes in the network

The second simulation scenario depicted in Fig. 5 shows the evolution of node lifetime according to the number of nodes in the network and for two different transmission ranges: 10m and 125m.

We observe that node lifetime decreases slightly when the number of nodes increases. In fact, it ranges between 98s and 80s when there are twenty more nodes. This can be explained by the fact that the increase of the nodes number leads to a larger amount of traffic which contribute to a faster depletion of nodes battery. Regarding transmission range modification, we observe a low variation of values. Hence, transmission range has a small impact on node lifetime.

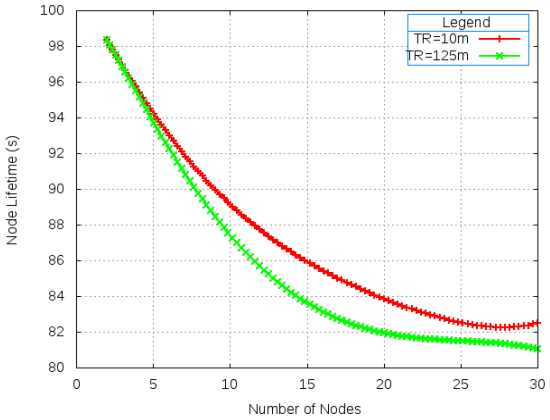

Figure 5. Node lifetime with varying number of nodes in the network

The third simulation scenario evaluates the number of clusters according to the transmission range variation and for different network densities (N=20, N=30 and N=40 nodes). As depicted by Fig. 6, we notice a large number of clusters for small transmission range because this latter leads to limited node visibility and consequently to a small clusters covering area. Moreover, we obtain a reduced number of clusters for large transmission range because in such case clusters cover larger areas. Note that this result is substantially the same for all three considered network densities. In fact, the number of clusters varies in a small interval almost [4, 8] for 20 nodes, [5, 13] for 30 nodes and [6, 13] for 40 nodes.

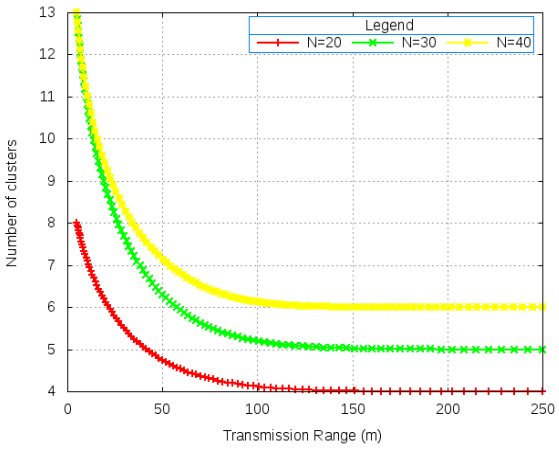

Figure 6. Number of clusters with varying number of nodes in the network

Fig. 7 depicts a simulation scenario considering the variation of a node lifetime with respect to a varying transmission range. We notice that node lifetime decreases as transmission range increases. This may be explained by the fact that when the transmission range increases, node visibility increases too leading to more exchanged traffic that contributes to lower the node lifetime.

 ISBN: 978-1-61208-284-4

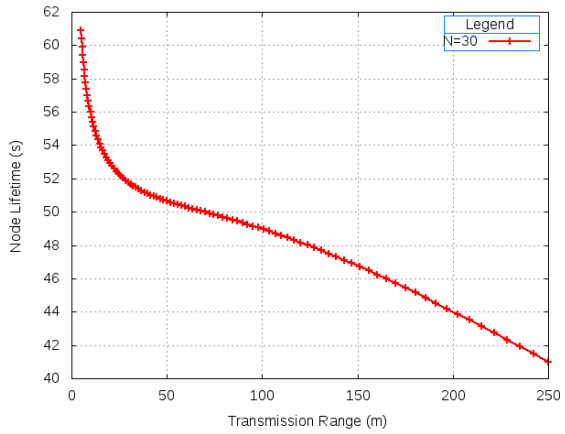

Figure 7. Node lifetime with varying transmission ranges

Fig. 8 depicts the variation of the number of clusters with respect node speed. We notice that this number is stable. In fact, with increase in speed, cluster members change clusters frequently; however, the number of clusters shows a low variation in value.

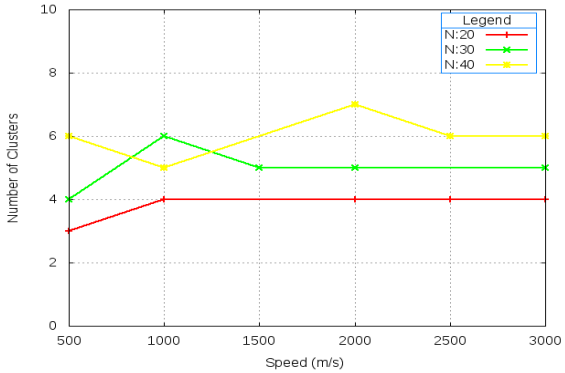

Figure 8. Number of clusters with varying nodes speed

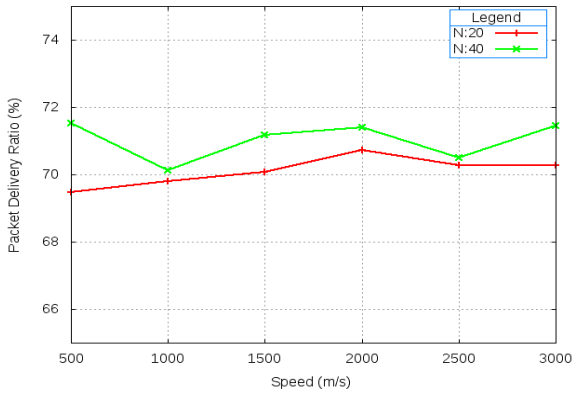

Figure 9. PDR with varying node speed

The last simulation depicted in Fig. 9 is concerned with the variation of the PDR when varying speed of nodes in the network. Since we have a normal environment without attackers target, nodes receive almost all generated data packets from the source node.

The results obtained demonstrate that our mobility based clustering algorithm selects optimal cluster heads and consequently achieves stability, reliability, and low maintenance. This is all the more interesting that our algorithm dealt with some specific initial constraints needed, as explained previously, for our trust management process enforcement.

VI. CONCLUSION AND FUTURE WORK

The dynamic topology of MANET, as well as the limited node capability and bandwidth pose significant problems for wide networks. Generally, and in order to resolve this problem, several propositions based on clustering techniques have been addressed. The major drawback of these techniques is that they do not take into account the mobility issue in an optimal way. This drawback has led us to propose a mobility-based clustering algorithm to form and maintain more stable clusters. The proposed clustering algorithm is based on two phases: the setting up and the maintenance. The first phase forms the clusters then elects CHs with the minimum weight computed through two parameters: mobility and residual energy. The second phase maintains the organization of clusters in the presence of mobility due to the displacement of a node, the failure of a node, or the arrival of a new node.

Several simulations were conducted in order to evaluate our algorithm performances in terms of number of CHs, nodes lifetime and PDR. These latter were evaluated with varying speeds, nodes number and transmission range.

In further work, we aim to establish a trust process based on Watchdog, delegation and the proposed clustering algorithm.

REFERENCES

[1] Z. Xing, L. Gruenwald, and K. Phang, “A Robust Clustering Algorithm for Mobile Ad-hoc Networks”, Handbook of Research on Next Generation Networks and Ubiquitous Computing, IGI Global, Editor Samuel Pierre, Chapter 18, December 2008.

[2] A. Nassuora and A. Hussein, “CBPMD: A New Weighted Distributed Clustering Algorithm for Mobile Ad hoc Networks (MANETs)”, American Journal of Scientific Research ISSN, 1450-223X, Issue 22, 2011, pp. 43-56.

[3] I. Shayeb, A. Hussein, and A. Nassuora, “A Survery of clustering schemes for mobile Ad-hoc Network (MANET)”, American Journal of Scientific Research ISSN, 1450-223X, Issue 20, 2011, pp. 135-151.

[4] M. Elhdhili, L. Azzouz, and F. Kamoun, “Lowest weight: reactive clustering algorithm for ad hoc networks”, In Proceedings of the 11th IFIP International Conference on Personal Wireless Communications, pp. 135–146, September 2006, Spain.

[5] A. Ephremides, J. Wieselthier, and D. Baker, “A design concept for reliable mobile radio networks with frequency hoping signaling”, In Procceedings of the IEEE 75, January 1987, pp. 56-73.

[6] A. K. Parekh, “Selecting routers in ad hoc wireless networks”, In Proceedings of the SBT/IEEE International Telecommunications Symposium, August 1994, pp. 420-424.

[7] P. Basu, N. Khan, and C. Little, “A mobility based metric for clustering in mobile ad hoc networks”, In Proceedings of IEEE ICDCS workshop on wireless networks an mobile computing, April 2001, pp. 413-418, Phoenix, AZ.

[8] M. Chatterjee, S. Das and D. Turgut, “WCA: A weighted clustering algorithm for mobile ad hoc networks”, Journal of Cluster Computing, vol. 5, no. 2-4, April 2002, pp. 193-204.

 ISBN: 978-1-61208-284-4