Computer Communications 32 (2009) 189–202

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

A distributed group mobility adaptive clustering algorithm for mobile ad hoc networks ☆

Yan Zhang *, Jim Mee Ng, Chor Ping Low

School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore

ARTICLE INFO

Article history:
Received 18 February 2008
Received in revised form 28 September 2008
Accepted 2 October 2008
Available online 17 October 2008

Keywords:
MANET
Clustering algorithm
Group partition
Group mobility
Spatial dependency
Ad hoc networks

ABSTRACT

This paper proposes a distributed group mobility adaptive (DGMA) clustering algorithm for mobile ad hoc networks (MANETs) on the basis of a revised group mobility metric, linear distance based spatial dependency (*LDSD*), which is derived from the linear distance of a node's movement instead of its instantaneous speed and direction. In particular, it is suitable for group mobility pattern where group partitions and mergence are prevalent behaviors of mobile groups. The proposed clustering scheme aims to form more stable clusters by prolonging cluster lifetime and reducing the clustering iterations even in highly dynamic environment. Simulation results show that the performance of the proposed framework is superior to two widely referenced clustering approaches, the Lowest-ID clustering scheme and the mobility based clustering algorithm MOBIC, in terms of average clusterhead lifetime, average resident time, average number of clusterhead changes, and average number of cluster reaffiliations.

1. Introduction

A mobile ad hoc network (MANET) consists of a number of wireless hosts that communicate with each other through multi-hop wireless links in the absence of fixed infrastructure. Some envisioned that MANETs may possibly contain thousands of mobile nodes as well as connections, such as in mobile military networks. The network complexity of such a huge network is in exponential growth as the number of mobile hosts increases, thus a flat network structure poses scalability problems with respect to routing and mobility management [1]. If each node had a complete view of the entire network topology, routing table would explode to an immense size. Clustering is hence vital for efficient resource utilization and load balancing in large-scale dynamic networks such as ad hoc networks and sensor networks. A cluster is a connected graph composed of a clusterhead (CH) and possibly some ordinary nodes. Using a cluster structure, topology information is aggregated and routing messages are only exchanged at higher hierarchical level normally constructed by CHs, which are responsible for the formation of clusters and maintenance of the network topology whereas local group members only have the view of a fraction of the total network. Clustering algorithms are used to partition networks into multi-level hierarchical organization. By dividing the network into non-overlapped clusters, intra-cluster routing is administered by the CH and inter-cluster routing can be done in a reactive manner by CHs and gateways.

However, clustering in MANETs is quite different from that in static networks such as sensor networks, since MANETs consist mostly of mobile devices. This makes clustering more complicated as it must be performed in a fully distributed, real-time, and mobility-adaptive fashion. Having stable clusters in MANETs makes dynamic topology appear less dynamic and large networks seem smaller.

Several clustering strategies have been proposed to increase the stability, routing performance, scalability, bandwidth utilization, and resource allocation efficiency. However, most of the clustering schemes are not mobility-aware or assume low mobility which cannot reflect the distinct dynamic character of MANETs. Mobility is characterized as an inherent phenomenon of MANETs, hence, discussing clustering algorithms by assuming low mobility becomes meaningless. Apart from this, it has been observed that in many application scenarios, such as military operations and searching and rescue scenario, mobile nodes are moving in a similar pattern in a number of groups and such a mobility pattern is known as group mobility [2]. For group mobility, group membership does not change frequently and thus it is more efficient to elect a CH or a number of CHs to represent each mobile group to maintain a relatively stable group structure while noting that

☆ A preliminary version of this paper is published in the Proc. of IEEE International Conference on Communications (IEEE ICC 2008) [8].

* Corresponding author. Tel.: +65 67906528.
E-mail addresses: zh0003an@ntu.edu.sg (Y. Zhang), ijmng@ntu.edu.sg (J.M. Ng).

0140-3664/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2008.10.002

topology changes may still happen with group partitions or mergence. Therefore, our algorithm attempts to capture the group mobility pattern and uses this information to form stable clusters.

To resolve the deficiency in existing clustering schemes, we propose a distributed group mobility adaptive (DGMA) clustering algorithm for MANETs [8]. The motivation of DGMA is to provide an optimal clustering method for nodes with group mobility pattern as described before. The formation of clusters is determined by the mobility pattern of group members to ensure maximum cluster stability. DGMA is superior to other clustering schemes in many aspects.

Firstly, in DGMA, speed is no longer a constraint on the usefulness of the algorithm since it is a mobility metric based algorithm that can adapt to high speed environment. The mobility metric expressed with spatial dependency in DGMA is based on physical location changes where "micro" movement is ignored and only "macro" location change is considered.

Secondly, DGMA is a distributed algorithm and is executed at each node with its sole knowledge of local information. Furthermore, the commonly used assumption for distributed clustering algorithm that nodes do not move while the cluster formation is in progress [3] is removed in DGMA. DGMA is a real-time distributed clustering algorithm.

Thirdly, DGMA is adaptive to group partitions. As shown in our simulation studies, we modeled mobile groups' regular partitioning in a similar manner as in the real world by using the Reference Region Group Mobility (RRGM) [4] model. Based on our best knowledge, only one previous work has mentioned clusters formation under group mobility [5]; however, the assumption for their algorithm is that no network partition would happen that deviates from many practical application scenarios.

Lastly, we propose a linear distance based spatial dependency (*LDSD*) as the metric for clustering. The spatial dependency [6] in DGMA relies on mobility information from a node's historical record instead of their instant speed. Furthermore, DGMA is simple to implement with a coloring approach for forming clusters. Various colors represent nodes with different characters such as CHs and ordinary members.

In DGMA, changes in positions are measured for nodes periodically. A mobile node should have the knowledge of its physical location. Thus, we assume every node is equipped with some positioning system, such as GPS, to capture location changes in coordinates. Such information is extracted by each individual node but it may not be necessary to share this information locally among its connected neighbors so as to reduce the network communication overhead. On the other hand, Received Signal Strength Indication (RSSI) is a measurement of the power present in a received radio signal. Based on the received signal strength, the receiver could identify the distance to the senders. In DGMA, the RSSI propagation model is applied to identify a node's neighborhood structure. Thus, a mobile node could get the complete view of its neighborhood.

The rest of the paper is organized as follows. Previous work done in the area of clustering is reviewed in Section 2. Section 3 describes the approach we have adopted to calculate the mobility metric for each node and highlights how this mobility metric is used in DGMA. Section 4 presents the details of DGMA. The performance of DGMA and its comparison with the clustering schemes Lowest ID [7] and MOBIC [15] are given in Section 5. Finally, Section 6 concludes this paper.

2. Related work

This section consists of two main parts. We first review several existing clustering algorithms in MANETs. Next, a group mobility model used to implement DGMA is described.

2.1. Clustering schemes

From wireless networks to current MANETs, a number of clustering schemes have been proposed in the past years. Some of them have been described in details in the recent survey papers [9,10] on clustering schemes for ad hoc network. These clustering schemes are categorized based on their objectives including energy efficient [11,12,25], low-maintenance [13,26], mobility-aware [3,5,14–16] and weight based [17–19] etc.

In the prior work, the Lowest-ID and its variants are the most commonly referred scheme in many research papers. In Lowest-ID, each node is assigned a distinct ID. If a node has a lower id than all of its directly connected neighbors, it becomes a CH; otherwise, it is an ordinary node. When two CHs encounter, the one with higher id should give up its role of CH. The major drawback of this algorithm is that it solely considers the node id without considering the qualification of a node being elected as a CH, and certain nodes are prone to drain out of power.

A variety of clustering schemes are variations of Lowest-ID by adding some featured constraint factors such as transmission power, energy consumption, and mobility speed. For example, Multi-hop clustering based on neighborhood benchmark [27], Maximum connectivity clustering (MCC), Least Clusterhead Change Algorithm [20] and Mobility Metric Based Clustering (MOBIC) [15], all of these are originated from Lowest-ID with certain annexed conditions. In [27], the author proposed a very innovative clustering metric called Benchmark to qualify connectivity and link stability. In MCC, CH election is based on degree of connectivity instead of node id such that CHs are mobile nodes with "local highest node degree". This algorithm suffers from dynamic network topology with frequent changes of CHs. LCC has improved the performance of Lowest ID by importing an additional rule on the clustering process: when a non-CH node in cluster i moves into another cluster j, no CHs in cluster i and j will be changed. MOBIC is a mobility-aware clustering scheme. Instead of using the node's ID, MOBIC uses the relative mobility, M_Y^{rel}, at a node Y with respect to X, as mobility metric as shown by Eq. (1):

$$M_Y^{rel}(X) = 10\log_{10}\frac{RxPr_{X\to Y}^{new}}{RxPr_{X\to Y}^{old}} \tag{1}$$

where the received signal power, RxPr, is used for estimating the distance between nodes X and Y, and M_Y^{rel}, indicates if two nodes are moving apart or closer together. By calculating the variance of a mobile node's relative mobility with each of its neighbors, the aggregate local relative mobility of this node can be estimated. A small value of variance indicates that the mobile node is relatively less mobile than its neighbors [15]. Consequently, a mobile node with the smallest relative mobility among its neighborhood takes a CH role.

It is not uncommon to see mobile nodes moving with similar speed and direction as on expressways. The relative mobility is able to reflect the similarity in motion between nodes. However, if mobile nodes move in an arbitrary manner, the diversity with other nodes is dominant. Even if most of a node's neighbors are having similar motion but a few others move differently, it still results in dramatic increase in the variance. That means the relative mobility would be greatly affected by nodes with diverse motion. In such a case, the performance of MOBIC may be degraded greatly.

2.2. Group mobility

In this work, group mobility environment is considered as our experiment mobility environment. Currently, there are a number of group mobility models being proposed in the domain of MANETs. Reference Point Group Mobility Model (RPGM) [21] is a well-known

mobility model used for generic scenarios for many ad hoc applications, such as search and rescue in military tasks. In RPGM, a number of mobile nodes surround a leader to construct a group. Group mobility behavior is dependent on the leader. There are some others variation of RPGM model, e.g. Reference Velocity Group Mobility model [22] that uses reference velocity as the mobility metric to define the motion of the group. Zhou et al have proposed the VT model in [24] which uses some "virtual tracks" to model the dynamics of group mobility based on pre-deployed switch stations that group movement is restricted. There are some other group mobility models for specific realistic scenarios as describe by Tracy Camp in [23], e.g. The Column group mobility model, Nomadic Community group mobility model, and Pursue group mobility model. The movement patterns provided by these mobility models can be obtained by changing the parameters associated with the RPGM Model. However, the drawback of RPGM model as well as all its variations is that it cannot describe grouping behaviors such as group partitions and mergence. If without group membership updating, any clustering algorithm applied to a mobile group having constant membership is not meaningful to most group mobility scenarios. Similarly as a generic mobility model with RPGM, Reference Region Group Mobility model (RRGM) proposed in [4] is enhanced with regrouping character, i.e. group partition and mergence. It also shows its advantages in independency of mobile nodes from any group leader and group membership management scheme superior to RPGM.

In RRGM, every group is initially assigned with a destination and the group is then associated with a reference region which is like an enlarged reference point but is used to define the boundary of the group which is proportional to the group size. A reference region is placed in an area between a group's present location and its destination. Each node belonging to the group will randomly pick a location inside the reference region and moves towards to that point. A new reference region will be generated only after the last group member has moved into the region. For nodes arriving earlier than others, they may either pause or travel within the range of reference region in randomness mobility. After stationary at an intermediate location for a period of time waiting for all group members to arrive, the reference region will be relocated. As such, the reference region moves gradually towards the group destination and nodes will finally reach the group's destination by following the path of the reference region. Group destinations are generated regularly in RRGM. If multiple destinations are assigned to a group, such as multiple tasks in different areas are assigned to a team in the disaster recovery scenario, this group will be partitioned into a number of smaller subgroups, each with a new reference region associated with a different destination. Thus a number of the group members will move apart from the other members resulting in group partitioning. After the members from the original group have reached their respective destinations and have completed their tasks, they may join back to the original group by merging their respective reference regions with the original group's reference region. Thus mergence among groups is achieved.

As described, group partitions and mergence are reflected in RRGM model. RRGM can mimic natural group movement without any artificial regulations. Groups can move and partition arbitrarily in fast or slow paces by adjusting parameters, i.e. *Partition_interval*. Moreover, RRGM model can mimic all the scenarios that RPGM can produce by assigning a big number to the parameter of *Partition_interval* in RRGM that group partition is prohibited. Thus we select RRGM model as the mobility environment to implement our DGMA algorithm.

3. Mobility metrics

Bai et al. introduced the mobility metric *Spatial dependency* (*SD*) to capture the similarity of instant velocities and directions between a node pair that are within their communication range [6]. However, we have observed that nodes belonging to the same group are more likely to move within the same region over a period of time to complete their tasks. When nodes are moving within a stationary region, the group is considered to have no movement. For instance, a group of rescuers may occasionally concentrate their searching in a specific area, or a group of people stays together during a networking session in a conference or to engage in a discussion while visiting in an exhibition hall.

All of these scenarios reflect the group mobility characteristics with possibly stationary group while nodes are moving constantly. While moving toward a destination, a node may move around within a region frequently in an uninterrupted trajectory, say to carry out a search operation. As a result, only minor change has been made to the linear distance from a node's starting point. Such oscillating movement of individual's featured in group mobility does not contribute much to the group displacement and the vector summation of individual instant velocity may not reflect the group motion either. Therefore, *SD*, as well as other similar mobility metrics, is not suitable to reflect such a mobility pattern. We hence propose the *linear distance based spatial dependency* (*LDSD*), which is an extension of *SD*, in this paper.

In *LDSD*, the linear distance D is used as the measurement of node displacement. With this term, we can identify points where "macro" changes in location have been made by a node. A node is moving along with a group which is in motion only if a node has changed its physical location by more than a threshold D_{thr}, where in general, D_{thr} is greater than the radius of a group's reference region. Thus, each node has two velocity properties: the instant velocity and the velocity calculated based on D over time ΔT. Let Δx_T and Δy_T be the increment of the linear distance in x and y coordinates since the last significant movement was recorded, and $T = \widetilde{T}_i$ be the moment when the last history information is recorded for node i. To calculate the linear displacement of a node, we have:

$$\Delta x_{\widetilde{T}i} = (x_{ti} - x_{\widetilde{T}i}) \tag{2}$$

$$\Delta y_{\widetilde{T}i} = (y_{ti} - y_{\widetilde{T}i}) \tag{3}$$

where t is the current time and x_{ti}, y_{ti}, $x_{\widetilde{T}i}$, $y_{\widetilde{T}i}$ are the coordinates of the node i at time t and T, respectively. Thus, the linear distance D can be calculated by:

$$D = \sqrt{\Delta x_{\widetilde{T}i}^2 + \Delta y_{\widetilde{T}i}^2} \tag{4}$$

If $D \geqslant D_{thr}$, the most recent speed and direction of a node that will be entered to the history record are:

$$\widetilde{S}_i = \frac{D}{t - \widetilde{T}_i} \tag{5}$$

$$\tilde{\theta}_i = \begin{cases} \varphi \cdot \mathrm{sgn}(\Delta y_{\widetilde{T}i}) & \Delta x_{\widetilde{T}i} > 0 \\ \pi/2 \cdot \mathrm{sgn}(\Delta y_{\widetilde{T}i}) & \Delta x_{\widetilde{T}i} = 0 \\ (\pi - \varphi) \cdot \mathrm{sgn}(\Delta y_{\widetilde{T}i}) & \Delta x_{\widetilde{T}i} < 0 \end{cases} \tag{6}$$

where $\tan\varphi = \left|\frac{\Delta y_{\widetilde{T}i}}{\Delta x_{\widetilde{T}i}}\right|$ and $\tilde{\theta}_i \in (-\pi, \pi)$. The time $\widetilde{T}_i$, the speed $\widetilde{S}_i$, the direction $\tilde{\theta}_i$, and the present location of the node will be recorded in the history cache for node i with

$$x_{\widetilde{T}i} = x_{ti} \tag{7}$$

$$y_{\widetilde{T}i} = y_{ti} \tag{8}$$

$$\widetilde{T}_i = t \tag{9}$$

Hence, at time t, a node's location is compared with the most recent record in the history cache such that a node's "micro" move-

ment trajectory within D_{thr} will not trigger any updating of the movement.

Based on above, each node will periodically check its own mobility information at every time interval T_0. If its linear distance D is greater than D_{thr}, it updates its mobility history information correspondingly. Based on the information in the history cache, a node calculates its *linear distance based total spatial dependency* (*LDTSD*) with respective to its neighbors with the following steps:

Step 1: It exchanges its history information speed $\widetilde{S}_i$ and direction $\tilde{\theta}_i$ with its directly connected neighbors.

Step 2: A node calculates its *relative direction* (*RD*) and *speed ratio* (*SR*) with each of his directly connected neighbors. For example, for nodes i and j, the *RD* of these two nodes is the cosine of the angle between i and j at time t,

$$RD(i,j,t) = \cos(\tilde{\theta}_i(t) - \tilde{\theta}_j(t)) \tag{10}$$

and the *SR* between two nodes i and j is given by

$$SR(i,j,t) = \frac{\min(\widetilde{S}_i(t), \widetilde{S}_j(t))}{\max(\widetilde{S}_i(t), \widetilde{S}_j(t))} \tag{11}$$

Step 3: Calculate spatial dependency *LDSD*,

$$LDSD(i,j,t) = RD(i,j,t) * SR(i,j,t) \tag{12}$$

Thus, if a mobile node moves away from its neighbor, it will have a small *LDSD* between the two nodes. That means their mobility is independent from each other. On the other hand, if a node's movement is closely related to another node or is influenced by the nodes in its neighborhood, such that they are moving in a similar direction and speed, then they are expected to have a higher value for *LDSD*.

Step 4: Assuming that the number of a node's directly connected neighbors is n, after exchanging the *LDSD* with each neighboring node, the node takes the summation of all *LDSD* it has and gets the *LDTSD* by

$$LDTSD(i,t) = \sum_{j=1}^{n} LDSD(i,j,t) \tag{13}$$

A higher *LDTSD* value implies that node i has a larger neighborhood set and it has a similar mobility pattern with the majority of its neighbors. The speed and direction during the recent period may be strongly related to others. Thus, a node with a higher *LDTSD* value is entitled to represent and to reflect the mobility pattern of the group.

It should be noted that the data range for *LDSD* is between $[-1,1]$ and that for relative mobility in MOBIC is $(-\infty,+\infty)$. For a pair of nodes, if their direction and speed are closely correlated, *LDSD* will have a value close to 1 whereas their relative mobility will have a value close to 0. Hence, *LDSD* uses its maximum value to represent high correlation on mobility while relative mobility uses the minimum absolute value. On the other hand, if two peers move in an arbitrary manner or their mobility is obviously uncorrelated, relative mobility will have an extraordinarily large absolute value while *LDSD* will have a value close to 0.

4. Distributed group mobility adaptive clustering algorithm

In this section, we describe the proposed DMGA clustering scheme especially apt for group mobility pattern in MANETs. Our design objectives targets at the following issues for innovative aspects and ubiquitous usage in the MANET clustering domain:

(a) to prolong lifetime of cluster as well as CHs;
(b) adaptable in fast speed mobility scenarios, such as vehicular networks, and
(c) to reduce the frequency of re-clustering iterations.

Fig. 1 displays snapshots of network group topologies with different node densities and illustrates the clusters being formed using DGMA. Note that in the figures, square represents CHs whereas circle represents cluster members or idle nodes. The solid line between two nodes indicates the connectivity between CH and members. Seen from the figures, some cluster members are connected to more than one CH. This phenomenon could easily be found in practical scenarios where these nodes with multiple connections to different CHs are considered as gateways between communication groups.

DGMA is implemented by employing a coloring mechanism that specific colors, red, yellow, and white, are assigned to nodes for representing their attributes in different states. White represents a node either in the initial color or in a distance of at least two hops from the closest cluster head. According to their status, they are named as idle nodes. CHs are colored with red while cluster members are in yellow. Clusters are organized by their CHs in the sense that each CH has a complete view of the entire cluster and cluster members are labeled with their CH's ID. If a cluster member connects with multiple CHs, it may simultaneously belong to multiple clusters. Membership maintenance scheme is executed at each node by creating a cluster *membership_table* which contains two fields: *membership_table.CH* and *membership_table.member*. The *membership_table.CH* keeps all the CHs that a cluster member connects with, while the *membership_table.member* logs all cluster members by a CH. Thus, CHs add in or delete their neighboring cluster members from the *membership_table.member* field consecutively with a certain update sequence, and a member tracks all the connected CHs in its *membership_table.CH* to indicate exactly which clusters it enrolls in. An idle node does not have to maintain its *membership_table* except setting it to blank as long as it turns to idle/white.

As introduced, *LDSD* functions as the mobility metric in DGMA to minimize capturing the instantaneous changes in speed and direction. The local movement within the range of a static cluster would not be captured according the *LDSD* mechanism. Thus it benefits DGMA scheme more adaptive in the fast speed scenarios proved with simulations as shown in Section 5.

We first introduce the terminology used in DGMA.

(1) T: the duration that two red nodes are directly connected.
(2) n_1: used by a red node to show the ratio of the number of one-hop white neighbors to the number of all one-hop neighbors. n_{1_Input} is the input threshold value of n_1.
(3) n_2: used by a white node to show the ratio of the number of one-hop white neighbors to the number of all one-hop neighbors. n_{2_Input} is the input threshold value of n_2.

Every node including CHs, member nodes as well as idle nodes maintain their respective status and clustering affiliations in the sense that no particular centralized pivot is necessary for organizing a whole cluster. Cluster mobility information is archived by CHs so that mobility information could be retrieved by *LDSD* scheme.

DGMA clustering algorithm contains two main processes: the nomination process and the cluster maintenance process. The

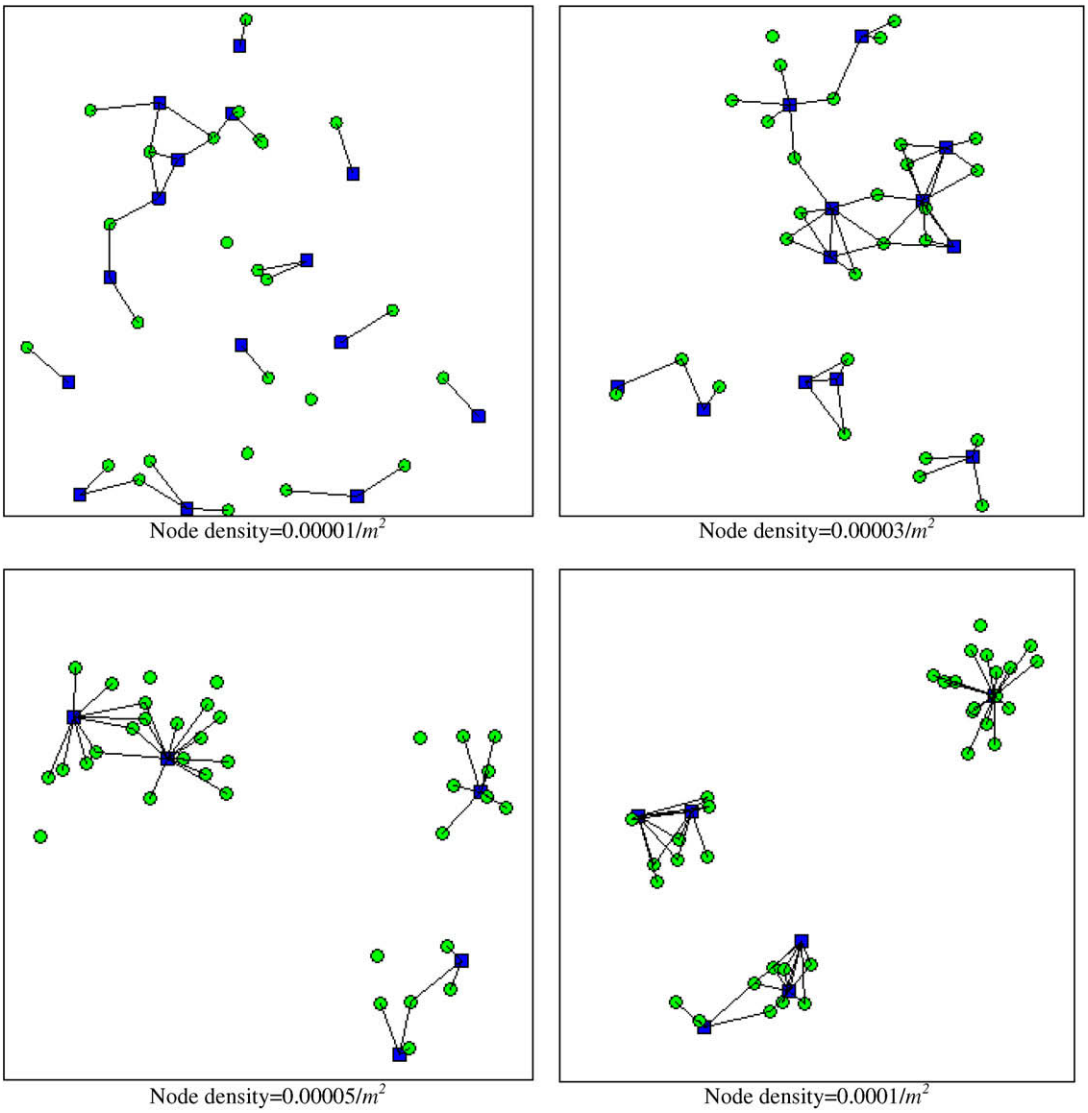

Fig. 1. Example of topologies resulting from DGMA clustering with different node densities.

pseudo-code for these two processes is given in Figs. 2 and 3. First, the nomination process is carried out in five steps:

Step 1: Each node starts in white as the initial color.
Step 2: A white node, if not connecting with any red node, broadcasts a *nomination request* to its directly connected neighbors to trigger a nomination process.
Step 3: Each neighboring white node feeds back the *nomination_request* with its *LDTSD*, the applicant (node A) compares its own *LDTSD* with the received *LDTSD* value of its white neighbors (node B), respectively.
  (a) If $LDTSD(A,t) < LDTSD(B,t)$, nomination process is terminated and node A remains in white.
  (b) If $LDTSD(A,t) > LDTSD(B,t)$, node A continues comparing its *LDTSD* until the last white neighbors.
Step 4: If node A has the largest *LDTSD* among all its white neighbors, it turns to red to form a new cluster labeled with the ID number of node A as the cluster ID. Moreover, the cluster *membership_table* maintained by node A is also updated with those white neighbors who have sent feedback messages to node A.
Step 5: Node A broadcasts an *invitation_message* to its neighbors. The receivers in white change their color to yellow and update their *membership_table* by adding this cluster ID to the *membership_table.CH* field. Note, if a cluster member node affiliates with multiple clusters simultaneously, there would be more than one cluster ID stored in the *membership_table*.

For the maintenance process executed at a node, it is triggered upon any update of mobility information. Different maintenance steps are executed based on the node colors from red, yellow and white separately.

(i) For red nodes, i.e. CHs:
  (1) The red node updates its neighbor information in *membership_table.member*.
  (2) If two red nodes are connected over time *T*, the one with smaller *LDTSD* turns into yellow. Both nodes update their own *membership_table*. For the red node, it adds the yellow node ID into its *membership_table.member* list; while for the yellow node, it

```
PROCEDURE Nomination (μ);
BEGIN
 Precondition:
    Input node: μ
    Neighborhood set of μ: Θ
    membership_ table.CH : list of connected CHs
    membership_ table.member : list of cluster members
 Effect:
    if μ.color is white
       for each v ∈ Θ
          if ∀ v that v.color is white, μ.LDTSD > v.LDTSD is always true
             μ.color = red;
             set u.membership_table.CH to blank;
             update u.membership_table.member with v.ID for all v in Θ;
          endif
       endfor
       if μ.color is red
          for each v ∈ Θ
             if v.color is white
               v.color = yellow;
               add u.ID into v.membership_table.CH
             endif
          endfor
       endif
    endif
END
```

Fig. 2. Protocol specification of nomination process.

resets its *membership_table.member* to blank and adds the red node's ID into *membership_table.CH* list.

(3) If the ratio of white neighbors among its neighborhood set is greater than $n_{1-Input}$, i.e. $n_1 \geqslant n_{1-Input}$, it turns into white and resets its *membership_table.member* to blank.

(ii) For yellow nodes, i.e. cluster member nodes:

1) If a yellow node has the largest *LDTSD* among all connected red neighbors, it turns into white and resets its *membership_table.CH* to blank. For all connected red nodes, they remove the yellow node ID from their *membership_table.member* field. Or,

2) If it fails to find any neighboring red node, it turns into white and resets its *membership_table.CH* to blank.

(iii) For white nodes, i.e. idle nodes:

(1) If the white node can find a red neighbor with a larger *LDTSD* value than itself, it turns to yellow. Both nodes update their own *membership_table*. For the red node, it adds the yellow node ID into its *membership_table.member* list; while for the yellow node, it updates its *membership_table.CH* with the red node ID.

(2) Else, if the ratio of white neighbors among its neighborhood set is greater than $n_{2-Input}$, i.e. $n_2 \geqslant n_{2-Input}$, it triggers a new cluster *Nomination* process.

The correctness and complexity of DGMA algorithm is analytically discussed next.

Lemma 1. *Θ_u is the neighborhood set of a node μ. If both μ and v are white nodes and $v \in \Theta_u$, at the end of nomination process, if μ. color = red, then v. color = yellow.*

Proof. Assume that $v.color \neq yellow$. That means node v may be either red or white. We discuss it in two cases. **Case 1**, if $v.color = red$. Since $\mu.color = red$, and $v \in \Theta_u$, hence $\mu.LDTSD > v.LDTSD$. Similarly, because $v.color = red$, and $v \in \Theta_u$, $\Rightarrow \mu \in \Theta_v$, hence $\mu.LDTSD < v.LDTSD$, which is a contradiction. Note that although in the maintenance process, two red nodes may directly connect temporally to avoid frequent CH re-elections, this would not happen in the initial nomination process as proved. **Case 2**, if $v.color = white$. *Since* $v \in \Theta_u$ and $\mu.color = red$, according to the clustering protocol of the nomination process, $v.color = yellow$ is executed eventually. Therefore, the assumption is incorrect. □

Lemma 2. *DGMA has a worst case processing time complexity of O(n) per node, where n is the number of nodes in the network.*

Proof. In the cluster nomination phase, the worst case happens when all nodes in the network are in the neighborhood set of a node μ assuming that μ is finally selected as the CH. It takes a processing time of at most $2(n-1)$. Similarly, for the phase of cluster maintenance, it takes a processing time of at most $(n-1)$ at each of red and yellow nodes and $2(n-1)$ at white node for the worst case. Therefore, the total processing complexity is $O(n)$. □

Lemma 3. *DGMA has a worst case communication (message exchange) complexity of O(1) at each node and for a network with n nodes, the communication complexity for the worst case is O(n).*

Proof. During every maintenance iteration, each node broadcasts a hello message which contains the information of node id, color and *LDTSD*. Only the newly elected red nodes broadcast one more CH message. Thus, the complexity is exactly $O(1)$ at each node. The number of messages has a upper bound of $O(n)$ in a network of size n. □

```
PROCEDURE Maintenance (μ);
BEGIN
 Precondition:
    Input node: μ
    Neighborhood set of μ: Θ
    membership_ table.CH : list of connected CHs
    membership_ table.member : list of cluster members
 Effect:
    if μ.color is red
       update u.membership_table.member;
       for each v ∈ Θ
          if ∃ v.color is red & connecting_time (μ, v) > T
             if μ.TSD < v.TSD
                μ.color = yellow;
                reset u.membership_table.member to blank;
                update u.membership_table.CH with v.ID;
                update v.membership_table.member with u.ID;
             else
                v.color = yellow;
                reset v.membership_table.member to blank;
                update v.membership_table.CH with u.ID;
                update u.membership_table.member with v.ID;
             endif
          else if n1 ≥ n1-Input
             μ.color = white;
             reset u.membership_table.member to blank;
          endif
       endfor
    endif

    if μ.color is yellow
       for each v ∈ Θ
          if that ∃ v.color is red is false
             μ.color = white;
             reset u.membership_table.CH to blank;
          else if ∀ v that v.color is red, μ.TSD > v.TSD is true
             remove v.ID from u.membership_table.CH;
             if u.membership_table.CH is blank
             μ.color = white;
             endif
          endif
       endfor
    endif

    if μ.color is white
       if ∃ v.color is red & μ.TSD < v.TSD
          u.color = yellow;
          update u.membership_table.CH with v.ID;
          update v.membership_table.member with u.ID;
       else if n2 ≥ n2-Input
          Nomination(μ);
       endif
    endif
END
```

Fig. 3. Protocol specification of maintenance process.

Lemma 4. *For a connected graph G(V,E), |V| ⩾ 2, there exists at least one CH.*

Proof. Assume $|V| = 2$, for $V = \{A,B\}$, if $max.LDTSD(A,B) = A$, A becomes CH; otherwise B is CH. Thus, we can assume for $G = (V,E)$, $|V| = n$, there exists at least one CH. If $|V| = n + 1$, we assume node i is CH for $G' = \{1,2,3,\ldots,i,\ldots,n\}$ which is a sub graph of G, thus $max.LDTSD(G'_j | j = 1,2,\ldots,n) = i$. For the vertex $n + 1$, we discuss it in two cases. **Case 1**, if node i and node $n + 1$ are directly connected neighbors, and $max.LDTSD(i,n+1) = n + 1$, we can derive $max.LDTSD(G''_j | j = 1,2,\ldots,n,n+1) = n + 1$ and the $(n + 1)$-th node becomes CH. **Case 2**, if node i and node $n + 1$ are not directly connected neighbors, node i is still the CH. So the statement holds in either case. □

Deduction: For a complete graph after the cluster nomination process, there is exactly one CH.

Proof. In a complete graph, $G = (V,E)$, each vertex has a degree of $n - 1$, and $|E| = n(n-1)/2$, which means every pair of vertices is adjacent. As proved in Lemma 4, there must have at least one CH which is also applied for a complete graph. We assume there exist two CHs, say node A and B. Since both A, $B \in V$, A and B is an adjacent pair of vertices in V. The node with larger *LDTSD* is assigned as CH. □

Observation: After the cluster nomination process is completed, if a white node is surrounded by only yellow nodes without any other white or red nodes, this white node is in two hops distance to its CHs through its directly connected yellow nodes. (Note: If more than 2 red nodes are in two hops distance from it, this white node functions like a gateway node.)

Lemma 5. *After the cluster nomination process is completed, the maximum distance from a cluster member to its corresponding CH is two hops.*

Proof. Consider a two-hop connection $A \sim B \sim C$ which means node A and B, B and C are directly connected neighbor pairs but A and C are not. Assume *LDTSD.A* > *LDTSD.B* > *LDTSD.C*, after cluster nomination process, A becomes red node and B turns to yellow, whereas node C still keeps as white according to the nomination process. Thus, the maximum connection within this cluster is two. If there is a connection more than two hops, say $A \sim B \sim C \sim D$, if node A is CH, node B would be yellow. According to Lemma 4, two connected white nodes can not co-exist after the cluster nomination process is completed and the one with greater *LDTSD* will be elected as CH. Thus for this case, either node C or node D will finally turn to red. The maximum distance is still within two hops distance. Similarly, Assume a series of connections $A \sim B \sim \ldots \sim C \sim D$ in which the maximum distance from a member to a CH is two hops. When another node, E, connects with one of the node in the node series, if a yellow node is connected, the distance from node E to a CH is two; else if a white node is connected, one of the white node will turn to a CH. Thus, by deduction, the statement is true for any number of nodes in G. □

5. Simulation results and discussions

Our experiment is set up in a *C++* environment which is developed by ourselves. The mobility model is generated with RRGM mobility generator developed in *C++* and it is as an input to the clustering algorithm module which is also developed by *C++*. We compare the performance of different clustering algorithm, i.e. DGMA with MOBIC and the Lowest-ID schemes. Four independent sets of experiments have been conducted with each experiment repeated for 20 times and the averaged value is returned as the result. The simulation is executed for 1500 s. 40 identical mobile nodes are deployed in a terrain of 1500 m by 1500 m. We will discuss our simulation results by first introducing the experiment environment that includes the specifications of mobility model and the configuration of DGMA. Comparison and analysis are then followed to show the performance gain when DGMA scheme is used.

5.1. Mobility model

In this paper, we verify our proposed clustering algorithm within a general group mobility environment. For example, in disaster

recovery of a geographical area as introduced in [21,23], multiple rescue teams may be deployed within a region with boundaries. According to the functions or assignments, each team may be formed with different roles, such as rescue personnel team, medical team and psychologist team, etc. They may form a larger group when a vital task is assigned. Alternatively, they may also split into several smaller teams if multiple tasks are assigned to the same team to carry out immediately. Because RRGM inherits the character of previous generic group mobility model, such as RPGM, plus additional regrouping character, RRGM model is adaptive to such group mobility environment with the phenomenon of group partition and mergence.

The parameters for RRGM model are listed in the Table 1. In RRGM, all nodes are clustered in one group initially. When the simulation starts, all nodes spread out into a number of groups which is specified by the "NUMBER-OF-NEW-DESTINATIONS". After that, group partitions may occur every "PARTITION-TIME-INTERVAL" depending on the partitioning preconditions defined in the RRGM model, while group mergence may happen at any moment.

The random waypoint mobility model is also implemented. Initially, the node is randomly placed at a point P1 within the region chosen from a uniform distribution in x and y coordinates, respectively. Then, a destination point P2 is chosen from a uniform distribution similarly with P1 and the node moves along a straight line from P1 to P2 with constant velocity randomly selected from the range of (minimum speed, maximum speed). Once the node reaches P2, a new destination point, P3, is generated from a uniform distribution and the process repeats.

5.2. Configurations of DGMA

Table 2 shows the control parameters for DGMA. As illustrated in Section 3, D_{thr}, the "CLUSTER-MOBILITY-DISTANCE", is specified for calculating the mobility metric. This value indicates the distance threshold over which a node updates its mobility information in its history cache. For instance, if D_{thr} is 100 m, a mobile node will periodically measure its physical linear distance every 1 s with respect to the spot where it has last updated its mobility information. If D is greater than 100 m, the node updates its history information derived from the linear distance of its displacement. The "CLUSTER-MAINTAINANCE-n1", which is to be used by a CH, represents the ratio between the number of neighboring white nodes and the total number of direct neighbors. If the ratio is greater than n_1, the CH has to turn into an ordinary node. In our experiment, 0.5 is assigned as default to this parameter, which means if more than 50% of a cluster's members have left its cluster, the corresponding CH will change its role to an ordinary node in the sense that it is no longer eligible to represent the nodes in this region. "CLUSTER-MAINTAINANCE-n2", which is to be used by a white node, represents the ratio between the number of neighboring white nodes and the number of all neighboring nodes. If the ratio is over n_2, then the cluster-nomination process will be triggered. "CLUSTER-MAINTAINANCE-red-red" defines the duration of two connected red nodes (CHs), after which they have to compete for the role of CH.

Table 1
Mobility parameters.

Parameters	Default value
NUMBER-OF-NEW-DESTINATIONS	2
GROUP-DENSITY	0.0001 node/m^2 (1 node in 100 m $\times$ 100 m)
AVERAGE-NODE-SPEED	10.0 M/S
TRANSMISSION-RANGE	200 M
PARTITION-TIME INTERVAL	50.0 S

Table 2
Clustering parameters.

Parameters	Default value
CLUSTER-MOBILITY-DISTANCE	100 m
CLUSTER-MAINTAINANCE-n1	0.5
CLUSTER-MAINTAINANCE-n2	0.5
CLUSTER-MAINTAINANCE-red-red	50 s

5.3. Performance evaluations of DGMA

We simulate and compare DGMA clustering scheme with MOBIC and the Lowest-ID clustering scheme by investigating cluster stability with respect to the nodes' *transmission range*, *speed*, *group partition interval* and *node density*. For DGMA, both of the control parameters n_1 and n_2 are varied from 0.3 to 0.5, and are represented as DGMA-0.3, DGMA-0.4, DGMA-0.5, respectively. Our results show that DGMA has better performance than the other two schemes in all investigated aspects, including *Mean lifetime of CHs*, *Mean resident time*, *Average number of CH changes*, and *Average number of cluster reaffiliations*. These performance metrics studied reflect the stability of clusters from different views.

- *Mean lifetime of CHs*, also known as cluster lifetime, is calculated by cumulating the time duration of all nodes being CHs and dividing it by the total number of CHs during one simulation run. A longer *Mean lifetime* implies a longer period of time a cluster may exist.
- *Mean resident time* of a node is the average time interval that a node affiliates with a certain cluster [14]. It reflects the average duration that an ordinary node stays within the range of a certain cluster. Its value depends on the dynamic character of mobile nodes and the CH with which a node affiliates.
- *Average number of CH changes* indicates the number of CH elections normalized over the simulation period. A larger value of this metric denotes frequent CH replacement. It has a close relationship with the approach used in selecting CHs and the mobility pattern. The aim for clustering algorithm is to reduce this value so that a cluster may last for a longer period of time so as to increase a cluster's stability.
- *Average number of cluster reaffiliations* is the summation of the number of a node joining a cluster and the number of CH election events in which a node becomes a CH over all nodes and with the result normalized over the simulation period. As a power set of *Average number of CH changes*, this metric reflects the overall overhead cost in the maintenance of a network's hierarchical structure by using the clustering algorithm. It has a global view on all network members such that for a more dynamic network, more cluster reaffiliations would be generated.

5.3.1. Transmission range

We vary the node's transmission range from 100 to 300 m with an increment of 50 m each time. As shown from Fig. 4(a)–(d), with the increasing of the transmission range, when both n_1 and n_2 are greater than 0.4, the *mean lifetime of CHs* grows fast and the duration has been improved by 30–60% when compared with the Lowest-ID. Lowest-ID has similar mean lifetime of CHs as DGMA-0.3. MOBIC initially has a long CH lifetime, but it decays gradually when the transmission range is enlarged.

A similar steadily increasing trend can be found in *mean resident time* for all of these three clustering schemes, in which DGMA has the fastest growing rate that the average resident time is prolonged by about 40–50% in the best case when compared with the Lowest-ID and MOBIC as illustrated in the Fig. 4(b).

MOBIC and Lowest-ID have the average number of CH changes in the range between that of DGMA-0.3 and DGMA-0.4 as illustrated by Fig 4(c). DGMA-0.5 distinctly has less CH changes than others. However, Lowest-ID exceeds DGMA-0.4 at transmission range 250 m and becomes closely to DGMA-0.5 at 300 m.

A longer transmission range is equivalent to larger group coverage and results in less cluster reaffiliations as illustrated by Fig. 4(d). It shows that MOBIC performs worse than DGMA and Lowest-ID, and that the latter is comparable to DGMA-0.4.

In general, when transmission range is expanded, more mobile nodes are covered by a CH. If the selection of a CH is insensitive to the node's movement within the cluster, for instance, by using the lowest ID and *LDSD*, the CH lifetime would be prolonged. When the radius is large enough, the lifetime of a CH will approach constant unless new members join the cluster. If the CH of a cluster is persistent, the corresponding *mean resident time* should also be extended and less cluster reaffiliations occur.

As shown in Fig. 4(a), when the transmission range is expanded, the CH lifetime under the Lowest-ID and DGMA are prolonged. This shows that both the Lowest-ID and DGMA are insensitive to the local movement of neighboring nodes. The possibility for a CH turning into a member is low if the cluster membership rarely changes. However, for MOBIC, since it relies on instant speed as its mobility metric, even though there are very few cluster membership changes, their relative mobility will still change as long as the neighboring nodes keep moving. With a larger transmission range and more nodes within a cluster, the aggregated relative mobility of a node may change more often as the relative mobility of a larger set of nodes needs to be considered. As a result, CHs may be replaced more frequently and the average lifetime is shortened.

5.3.2. Speed

The speed is varied from as low as 5–20 m/s for high speed mode in our simulation. When the speed is low, both the Lowest-ID and MOBIC perform well; however, when the speed increases, Lowest-ID drops fast in the mean lifetime while MOBIC only has a minor decrease as shown in Fig. 5(a). For the mean resident time, MOBIC performs similar to DGMA-0.3 but is worse than DGMA-0.4 and DGMA-0.5 while Lowest-ID performs worst with mean lifetime ~50% of DGMA-0.5 as shown in Fig. 5(b).

As node mobility become more dynamic, more CH changes and cluster reaffiliations occur as illustrated in Fig. 5(c) and (d). It has been observed that MOBIC has the highest number of average cluster reaffiliations while Lowest-ID has the fastest growing rate. DGMA-0.4's and DGMA-0.5's performance surpass the others. Although MOBIC is adaptive to change in speed, its performance is even worse than Lowest-ID in terms of *Mean resident time* and *Number of cluster reaffiliations*. As shown in Fig. 5(a)–(d), DGMA is more adaptive to speed changes.

From the simulation results, it can be observed that as mobile nodes increase in speed, a CH may be connected to another CH more frequently resulting in more CH re-elections. This in turn causes a decrease in both CH lifetime and cluster member resident time and an increase in number of cluster reaffiliations. Fig. 5(a) shows that the CH lifetime of MOBIC drops slower than the other two schemes despite its absolute value is not the best when the speed is lower than 15 m/s. This reflects that MOBIC is rather adaptive to highly dynamic environment. MOBIC achieves this by defining a CH contention interval (= 10.0 s in our simulation) to avoid frequent CH replacement when two CHs are directly connected. For the Lowest-ID, CH re-election will occur whenever two CHs are directly connected, resulting in a sharp decrease in CH lifetime

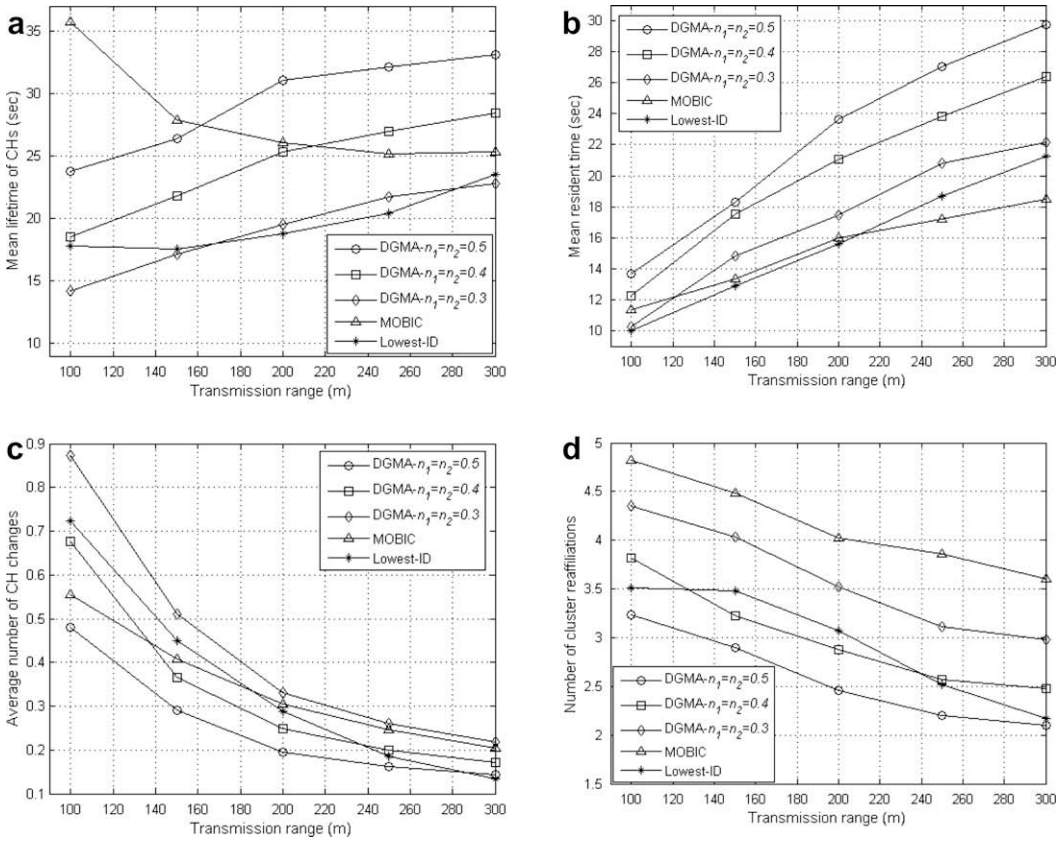

Fig. 4. Performance metrics vs. transmission range with group mobility. (a) Mean lifetime of CHs vs transmission range. (b) Mean resident time vs transmission range. (c) Average number of CH changes per sec. vs transmission range. (d) Average number of cluster reaffiliations per sec. vs transmission range.

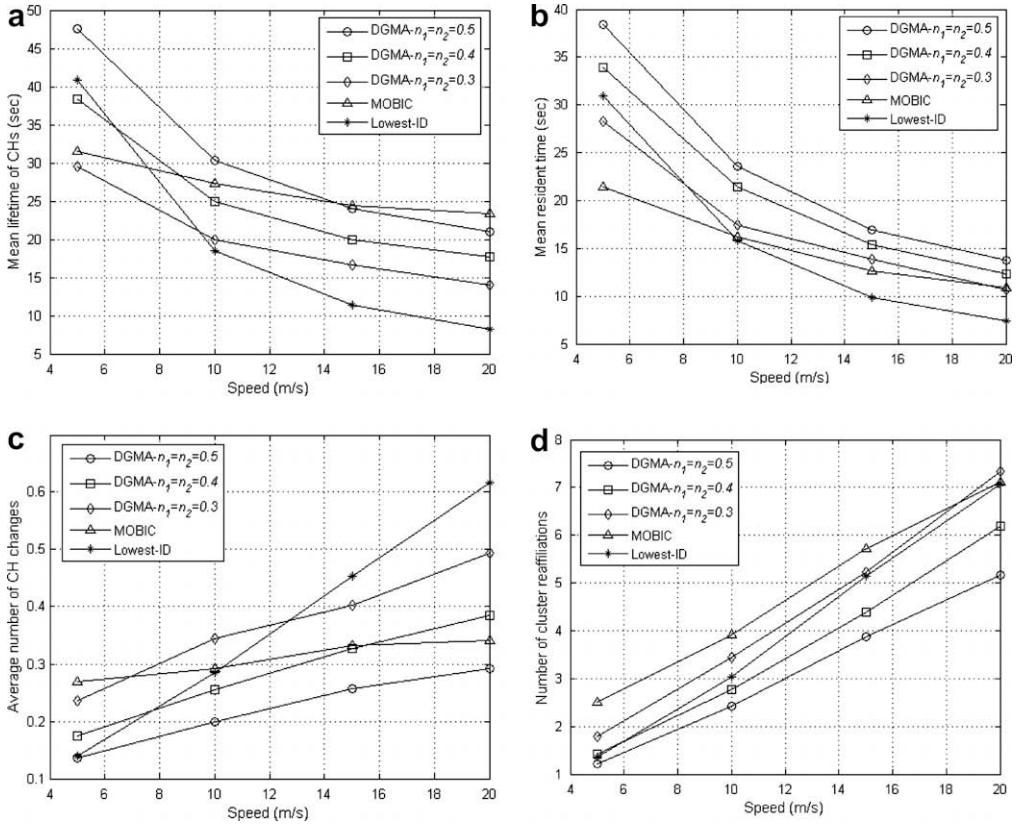

Fig. 5. Performance metrics vs. speed with group mobility. (a) Mean lifetime of CHs vs. speed. (b) Mean resident time vs. speed. (c) Average number of CH changes per sec. vs. speed. (d) Average number of cluster reaffiliations per sec. vs. speed.

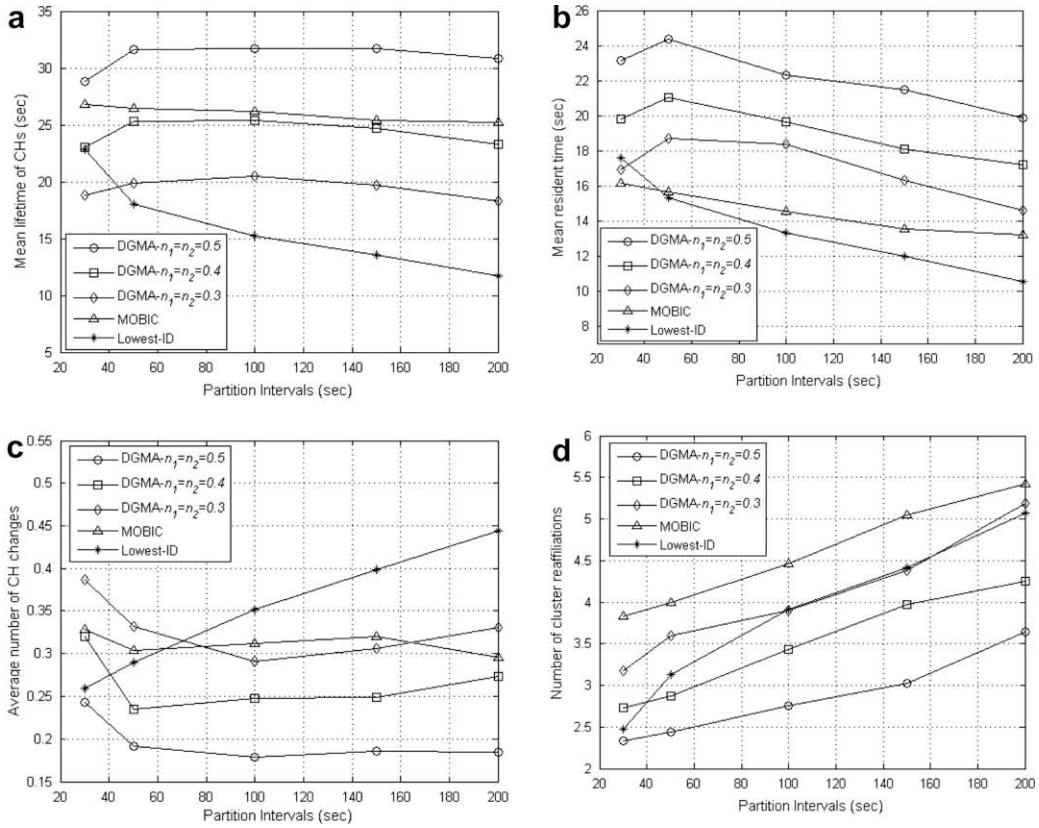

Fig. 6. Performance metrics vs. partition intervals with group mobility. (a) Mean lifetime of CHs vs. partition intervals. (b) Mean resident time vs. partition intervals. (c) Average number of CH changes per sec. vs. partition intervals. (d) Average number of cluster reaffiliations per sec. vs. partition intervals.

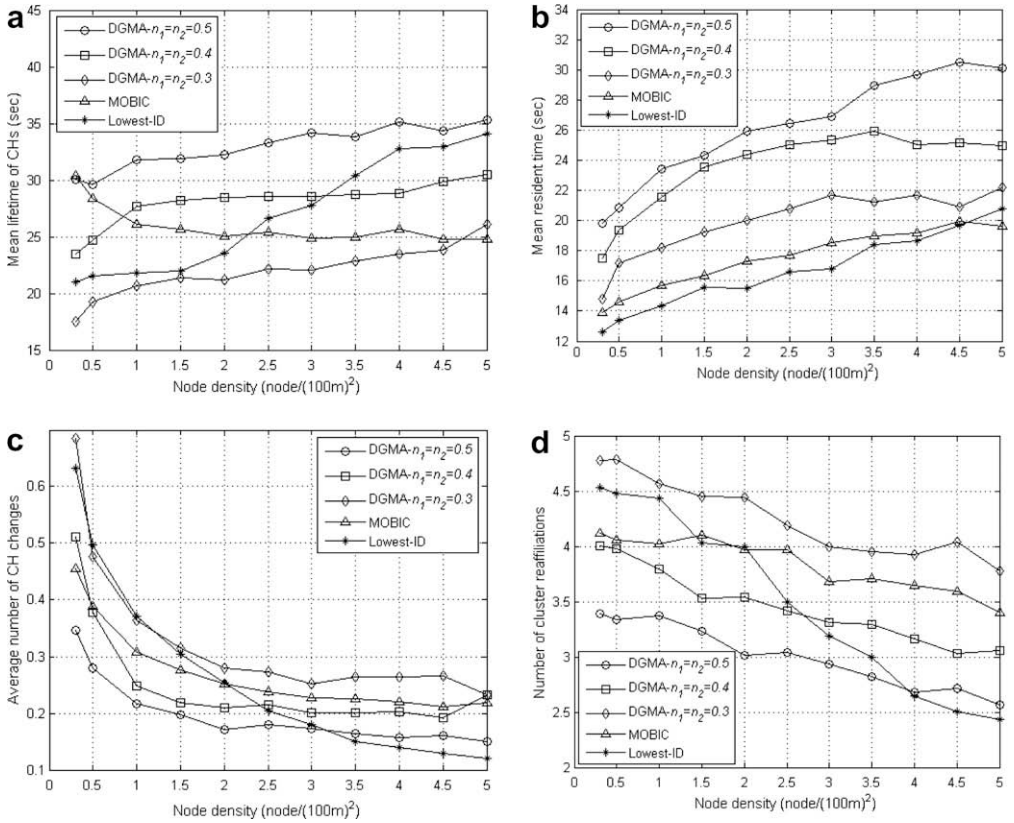

Fig. 7. Performance metrics vs. node density with group mobility. (a) Mean lifetime of CHs vs. node density (nodes/(100 m)^2). (b) Mean resident time vs. node density (nodes/(100 m)^2). (c) Average number of CH changes per sec. vs. node density (nodes/(100 m)^2). (d) Average number of cluster reaffiliations per sec. vs. node density (nodes/(100 m)^2).

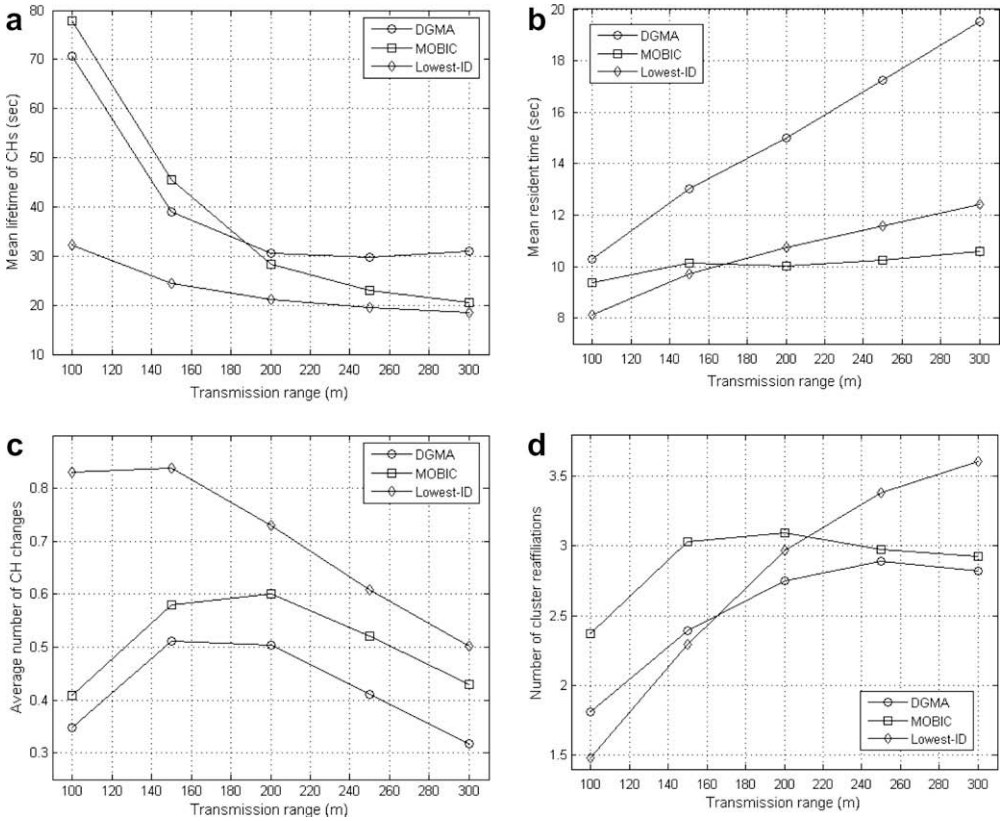

Fig. 8. Performance metrics vs. transmission range with RWP mobility. (a) Mean lifetime of CHs vs. transmission range. (b) Mean resident time vs. transmission range. (c) Average number of CH changes per sec. vs. transmission range. (d) Average number of cluster reaffiliations per sec. vs. transmission range.

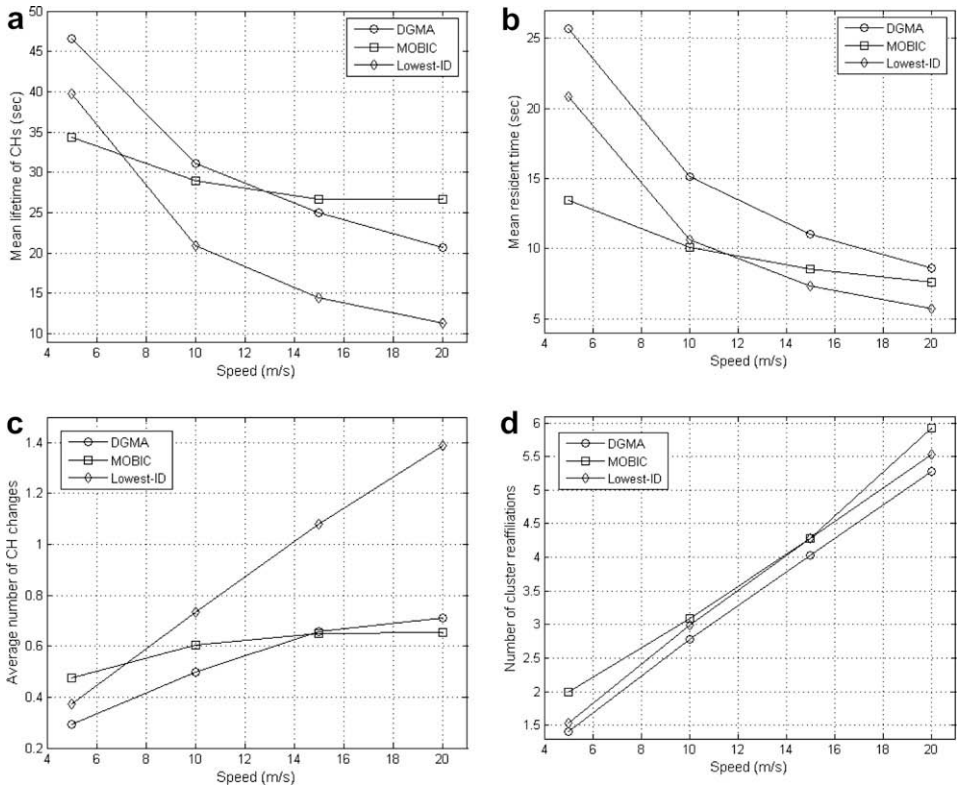

Fig. 9. Performance metrics vs. speed with RWP mobility. (a) Mean lifetime of CHs vs. speed. (b) Mean resident time vs. speed. (c) Average number of CH changes per sec. vs. speed. (d) Average number of cluster reaffiliations per sec. vs. speed.

at high speed. Despite of having a CH contention time for DGMA, the CH lifetime drops a bit faster than MOBIC because besides the CH contention time, DGMA also depends on the composition of a CH's neighborhood in CH replacement. Thus, although MOBIC has a longer *Mean lifetime of CHs*, its mean resident time and average number of cluster reaffiliations are worse than that of DGMA as some CHs may not be a good representative for a cluster, which may have negative impact on routing. Another negative aspect of considering CH contention interval only in MOBIC is that it may create more co-existing CHs, which causes more cluster reaffiliations as shown in Fig. 5(d).

5.3.3. Partition interval

Group partitioning interval is an important metric to evaluate a clustering scheme in a dynamic environment. Unfortunately, no existing work that we are aware of has considered this before. We introduce this new metric in this paper to show network partition may affect the clustering performance.

We set the partition intervals to 30, 50, 100, 150 and 200 s, respectively. According to the RRGM model, when the interval is short, group partitioning will happen more often and smaller groups will be formed frequently. On the other hand, a longer partition interval may cause group partition to happen less frequent so that mobile nodes may stay in a big group for a longer while.

For DGMA and MOBIC, when the partition interval is 30 s, most CH replacements are resulted from group partitioning instead of changes of other mobility metrics. When the partition interval is beyond 50 s, the lifetime becomes stable since fewer new groups are generated regularly. MOBIC has a similar mean lifetime of CHs to DGMA-0.4 and both are not affected much by group partitions. As seen from Fig 6(a), both DGMA and MOBIC can keep the lifetime in a steady range. This also can be reflected by the *Average number of CH changes* in Fig. 6(c) where the curves of DGMA and MOBIC are constant with small fluctuation. However, for Lowest-ID, the average CH lifetime declines very fast to ~50% as the partition interval increases from 30 to 100 s, and the average number of CHs changes also grows dramatically. When the partition interval is long, small groups may merge together to form a large group which increase the chance of a low ID node encountering another node with a lower ID. This causes the *mean lifetime of CHs* of Lowest-ID scheme to decline when the partition interval becomes longer.

On the other hand, when more large groups have been formed with the partition intervals become longer, a large group may possible contain multiple CHs. Thus, for a member node, it may hop among different clusters which decreases the *Mean resident time* and increases the *Number of cluster affiliations* as illustrated in Fig. 6(b) and (d), respectively. DGMA has ~50% longer *Mean resident time* than MOBIC and Lowest-ID.

From the simulation results, Lowest-ID performs worst and is affected strongly by the partitioning intervals. The performance of MOBIC is reasonably well and DGMA shows the best performance with predictably steady results which are not affected much by the group partitioning.

5.3.4. Node density

Performance metrics are investigated at different node density from 0.3 node to 5 nodes over an area of 100 m × 100 m. As observed from Fig. 7(a), both DGMA and Lowest-ID appear in an increasing trend when the node density increases. DGMA levels up gradually with a small increment each time in contrast to Lowest-ID's rapid growth. When the density is high, group radius and the number of clusters become smaller. Ultimately, most cluster members are covered within the transmission range of a CH. Thus the *mean lifetime of CH* of Lowest-ID increases dramatically since the node with the smallest ID will remain as the CH of the group. However, for DGMA and MOBIC, even all mobile nodes in a group are covered by a CH, CH re-elections may still happen

if their mobility metrics change. When the density is over 2 nodes/(100 m)^2, the CH lifetime of MOBIC approaches constant in the sense that density change does not have much impact on CH lifetime subsequently from this point. Before this point, from 0.3 to 1.5 nodes/(100 m)^2, with the transmission range fixed at 200 m, the increase of node density makes more mobile nodes to be under the coverage of a CH. This cause changes in the relative mobility of MOBIC and resulting in a decrease in CH lifetime.

In general, when a group is stable and a cluster physical coverage is small, the CH should have a longer lifetime. The lifetime of CH in Lowest-ID scheme improves rapidly and it becomes steady and is close to DGMA-0.5 after 4 nodes/$(100\,m)^2$. CH's lifetime under MOBIC does not fluctuate much and is between DGMA-0.3 and DGMA-0.4. Note that the *Average number of CH changes* also displays an analogous performance among all schemes as shown in Fig. 7(c).

For the *mean resident time*, all three schemes show an increase in resident time with higher density as a result of a smaller cluster size; however, both MOBIC and Lowest-ID appear to perform worse than DGMA-0.3 as illustrated in Fig. 7(b).

MOBIC makes less cluster reaffiliations than DGMA-0.3 but more than DGMA-0.4 and DGMA-0.5 as shown in Fig. 7(d). On the other hand, Lowest-ID displays a fast reduction rate and finally surpasses DGMA-0.5 in the number of cluster reaffiliations when density increases.

From the simulation results, DGMA is superior to MOBIC as the node density increases. Although Lowest-ID performs closely to DGMA-0.5 when mobile nodes are highly concentrated, it is not suitable for sparsely distributed mobile nodes.

5.4. Random mobility

Although DGMA is originally designed to support group mobility, in this section, we present our investigation on the performance of DGMA for entity mobility by using the random waypoint (RWP) mobility model for a broader understanding on the performance of DGMA. We still compare the performance of DGMA with the other two schemes, MOBIC and the Lowest-ID, by varying the node transmission range and node speed separately. Similar to the previous experiments, we set the default average speed to 10 m/s, the transmission range to 200 m, and the pause time to zero. Other parameters are set with the default values as discussed in Section 5.

It is easy to understand that with random mobility and when the transmission range is extended, more mobile nodes are included in the range of a cluster and the cluster's membership changes more frequently. Therefore, competition to become a CH will occur more frequent and the CH lifetime is shortened as shown in Fig. 8(a). MOBIC performs better than DGMA by ~10% when the transmission range is less than 200 m, but after which MOBIC fells fast and eventually achieves only about 60% of the CH lifetime under DGMA. In general, DGMA has about 50% longer mean CH lifetime than the Lowest-ID. Cluster members in DGMA have an obvious longer resident time than the other two as illustrated by Fig. 8(b).

From Fig. 8(c), it can be seen that there are less CH replacements when the transmission range is 100 m because the transmission range is not large enough and many nodes turn to be a CH without any members. With the increase of the transmission range, more mobile nodes are included in a cluster, and this leads to an increase in the number of CH changes. When the transmission range is large than 200 m, the number of clusters decreases thus the number of CH changes declines too. DGMA has less cluster reaffiliation events than MOBIC and Lowest-ID because of less CH re-elections as shown in Fig. 8(d).

Fig. 9(a)–(d) shows the performance metrics by varying the speed from 5 to 20 m/s. With a higher speed, the possibility of encountering another CH increases, and the CH lifetime is hence shortened. However, for DGMA and MOBIC, where CH contention interval has been defined, the CH lifetime drops slower than Lowest-ID. This can also be verified by the number of CH changes in Fig. 9(c). For a cluster member, it changes its cluster frequently if it moves fast, as reflected in Fig. 9(b) where DGMA has the longest *Mean resident time* among the three schemes. In Fig. 9(d), all three schemes have a linear increasing trend in the *Number of cluster reaffiliations* with DGMA having the least reaffiliations; however, the difference is not significant as the cluster members are equally dynamic no matter which clustering scheme is selected. The node distribution within a cluster and the position of the CHs may cause cluster reaffiliations of each scheme to deviate from each other slightly. Thus, it can be concluded that DGMA outperforms both MOBIC and Lowest-ID and MOBIC is better than Lowest-ID as speed increases for the entity mobility pattern.

From the above simulation results and discussion, we have shown that DGMA is also applicable for the entity mobility environment with a relatively good performance.

6. Conclustion and future work

With the development of MANETs, large-scale MANETs will become one of the future research directions. The routing efficiency and network complexity would be greatly influenced by the number of mobile hosts. Clustering can provide large-scale MANETS with a hierarchical network structure to facilitate network operations. Thus, clustering approach will be an active research arena. In this paper, we proposed a distributed clustering scheme for group mobility in MANETs as well as a novel group mobility metric, linear distance based spatial dependency. Based on the new mobility metric, it ensures the maximum cluster stability. As we have expected, the clusters formed by this scheme is more stable in terms of longer cluster lifetime, longer mean residence time for cluster members and with a smaller number of reaffiliation events. The results reflect that DGMA scheme outperforms Lowest-ID algorithm as well as MOBIC in many aspects.

Group partition is an inherited group mobility behavior. With the DGMA clustering scheme, proper group partition prediction algorithms can be developed based on the mobility information of the cluster heads. Efficient group routing protocols for mobile ad hoc is also a challenging topic based on this hierarchical cluster structure generated by DGMA.

References

[1] L. Kleinrock, F. Kamoun, Hierarchical routing for large networks, Computer Networks 1 (1977) 155–174.
[2] X. Hong, M. Gerla, G. Pei, C. Chiang, A group mobility model for ad hoc wireless networks, in: Proc. of ACM International Workshop on Modeling and Simulation of Wireless and Mobile Systems (MSWiM), Seattle, Washington, United States, August 1999, pp. 53–60.
[3] S. Basagni, Distributed clustering for ad hoc networks, in: Proc. of the 1999 International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN'99) IEEE Computer Society, Washington, DC, 1999, pp. 310–316.
[4] J.M. Ng, Y. Zhang, A mobility model with group partitioning for wireless ad hoc networks, in: Proc. of the IEEE International Conference on Information Technology and Applications, (ICITA 2005), Sydney, 5 July 2005, pp. 289–294.
[5] I.I. Er, W.K.G. Seah, Mobility-based d-hop clustering algorithm for mobile ad hoc networks, in: Proc. of IEEE Wireless Communications and Networking Conference, IEEE WCNC, Atlanta, USA, March 2004, pp. 2359–2364.
[6] F. Bai, N. Sadagopan, A. Helmy, Important: a framework to systematically analyze the impact of mobility on performance of routing protocols for ad hoc networks, in: Proc. of INFOCOM 2003, San Francisco, April 2003, pp. 825–835.
[7] A. Ephremides, J.E. Wieselthier, D.J. Baker, A design concept of reliable mobile radio networks with frequency hopping signaling, in: Proc. of the IEEE, vol. 75, no. 1, 1987, pp. 56–73.

[8] Y. Zhang, J.M. Ng, A distributed group mobility adaptive clustering algorithm for mobile ad hoc networks, in: Proc. of IEEE International Conference on Communications, 19–23 May 2008 (ICC 2008), Beijing, China.
[9] Y. Yu, P.H.J. Chong, A survey of clustering schemes for mobile ad hoc networks, IEEE Communications Surveys & Tutorials 7 (1) (2005) 32–48.
[10] D. Wei, H.A. Chan, A survey on cluster schemes in ad hoc wireless networks, in: Proc. of the 2nd International Conference on Mobile Technology, Applications and Systems, 2005, Guang Zhou, China, 2005, pp. 1–8.
[11] R. Palit, E. Hossain, P. Thulasiraman, Mobility-aware proactive low energy clustering in ad hoc mobile networks, in: Proc. of IEEE Global Communications Conference 2004 (GlobeCom'04), vol. 6, Dallas, TX, USA, November–December 2004, pp. 3426–3430.
[12] O. Younis, S. Fahmy, HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks, IEEE Transactions on Mobile Computing 3 (4) (2004) 366–379.
[13] G. Chen, F.G. Nocetti, J.S. Gonzalez, I. Stojmenovic, Connectivity based k-hop clustering in wireless networks, in: Proc. of the 35th Annual Hawaii International Conference on System Sciences, HICSS, Hawaii, 7–10 January 2002, pp. 188–191.
[14] C. Bettstetter, R. Krausser, Scenario-based stability analysis of the distributed mobility-adaptive clustering (DMAC) algorithm, in: Proc. of the 2nd ACM International Symposium on Mobile Ad hoc Networking & Computing, CA, USA, 2001, pp. 232–241.
[15] P. Basu, N. Khan, T.D.C. Little, A mobility based metric for clustering in mobile ad hoc networks, in: Proc. of IEEE ICDCS 2001 Workshop on Wireless Networks and Mobile Computing, Phoenix, AZ, April 2001, pp. 413–419.
[16] A.B. McDonald, T.F. Znati, A mobility-based framework for adaptive clustering in wireless ad-hoc networks, IEEE Journal on Selected Areas in Communications 17 (8) (1999) 1466–1487.
[17] S.K. Dhurandher, G.V. Singh, Weight based adaptive clustering in wireless ad hoc networks, in: Proc. of 2005 IEEE International Conference on Personal Wireless Communications, (ICPWC 2005), New Delhi, India, 2005, pp. 95–100.
[18] C. Johnen, L.H. Nguyen, Self-Stabilizing weight-based Clustering Algorithm for Ad hoc sensor Networks, Algorithmic Aspects of Wireless Sensor Networks, Springer, Berlin/Heidelberg, 2006, pp. 83–94.
[19] J. Lian, G. Agnew, K. Naik, A variable-degree based clustering algorithm for networks, in: Proc. of IEEE International Conference in Computer Communication and Networking, Dallas, TX, USA, November 2003, pp. 465–470.
[20] C.C. Chiang, H.K. Wu, W. Liu, M. Gerla, Routing in clustered multihop, mobile wireless networks with fading channel, in: Proc. of IEEE Singapore Conference on Networks (IEEE SICON'97), Singapore, 1997.
[21] X. Hong, M. Gerla, G. Pei, C. Chiang. A group mobility model for ad-hoc wireless networks, in: Proc. of the 2nd ACM International Workshop on Modeling and Simulation of Wireless and Mobile Systems, Seattle, WA, August 1999, pp. 53–60.
[22] K.H. Wang, B. Li, Group mobility and partition prediction in wireless ad-hoc networks, in: Proc. of the IEEE International Conference on Communications, New York City, NY, USA, April 2002, pp. 1017–1021.
[23] T. Camp, J. Boleng, V. Davies, A survey of mobility models for ad hoc network research, Wireless Communication & Mobile Computation (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications 2 (5) (2002) 483–502.
[24] B. Zhou, K. Xu, M. Gerla, Group and swarm mobility models for ad hoc network scenarios using virtual tracks, in: Proc. of the IEEE Military Communication Conference, Monterey, California, 2004, pp. 289–294.
[25] F. Tashtarian, A.T. Haghighat, M.T. Honary, H. Shokrzadeh, A new energy-efficient clustering algorithm for wireless sensor networks, in: Proc. of Software, Telecommunications and Computer Networks, SoftCOM 2007, 27–29 September 2007, pp. 1–6.
[26] V. Gayathri, E. Sabu, T. Srikanthan, Size-restricted cluster formation and cluster maintenance technique for mobile ad hoc networks, International Journal of Network Management 17 (2) (2007) 171–194.
[27] Stephen S. Yau, Wei Gao, Multi-hop clustering based on neighborhood benchmark in mobile ad-hoc networks, Mobile Networks and Applications 12 (5) (2007) 381–391.