On the Effect of Group Mobility to Data Replication in Ad Hoc Networks

Jiun-Long Huang and Ming-Syan Chen, Fellow, IEEE
Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan, ROC
E-mail: jluang@arbor.ee.ntu.edu.tw, mschen@cc.ee.ntu.edu.tw

Abstract

The growth in wireless communication technologies attracts a considerable amount of attention in mobile ad-hoc networks. Since mobile hosts in an ad-hoc network usually move freely, the topology of the network changes dynamically and disconnection occurs frequently. These characteristics make a mobile ad-hoc network be likely to be separated into several disconnected partitions, and the data accessibility is hence reduced. Several schemes are proposed to alleviate the reduction of data accessibility by replicating data items. However, little research effort was elaborated upon exploiting the group mobility where the group mobility refers to the phenomenon that several mobile nodes tend to move together. In this paper, we address the problem of replica allocation in a mobile ad-hoc network by exploring group mobility. We first analyze the group mobility model and derive several theoretical results. In light of these results, we propose a replica allocation scheme to improve the data accessibility. Several experiments are conducted to evaluate the performance of the proposed scheme. The experimental results show that the proposed scheme is able to not only obtain higher data accessibility but also produce lower network traffic than prior schemes.

Key words: Mobile data service, replica allocation, data accessibility, mobile computing, ad-hoc networks
1 Introduction

The advance in wireless communication enables users to access information systems from anywhere at any time via various mobile devices such as notebooks, tablet PCs, personal digital assistants (PDAs) and GPRS-enabled cellular phones. Service providers are establishing a number of mobile services including weather forecasting, stock information dissemination, location-dependent query, and route guidance, to name a few. To provide such services, researchers have encountered and are endeavoring to overcome challenges in various research areas including mobile data management [1], wireless network infrastructure [4], location-dependent data management [18], pervasive computing [31], and so on.

A mobile ad-hoc network (referred to as a MANET) is a self-organizing, rapidly deployable network which consists of wireless nodes without infrastructure. All nodes in a MANET are capable of moving actively and can be connected dynamically. Due to the lack of infrastructure, mobile nodes of a MANET also function as routers which discover and maintain routes, and forward packets to other nodes. A number of standards have been developed to support MANETs including IEEE 802.11 [17] and Bluetooth [8]. Example applications of MANETs include digital battlefield communications, personal area networks [32] and sensor networks. Most prior researches on MANETs focus on the development of routing protocols [22][30] to overcome the problems caused by node mobility. In addition, several cache frameworks are proposed to improve the performance of data access in MANETs [6][21][25][39].

With the advance of technology, a mobile device is equipped with a small storage to store data items. In a collaborative work scenario, several mobile devices may work together and access data items stored in other devices. Since each node in a MANET can move freely, the topology and connectivity of the network often change dynamically. In addition, disconnection often occurs and causes frequent network division. Both characteristics will separate the network topology into several disconnected partitions, and hence decrease the accessibility of data items. Consider the scenario in Fig. 1a, where there are six mobile devices ($M_1, M_2, \ldots, M_6$), and $M_1$ and $M_6$ store data items $D_1$ and $D_2$, respectively. Assume that the link between $M_1$ and $M_4$ disconnects due to the movement of mobile devices. Then, the network is divided into two partitions, $P_1 = \{M_1, M_2, M_3\}$ and $P_2 = \{M_4, M_5, M_6\}$. The data access to $D_2$ from a mobile devices in $P_1$ (e.g., $M_1$) will fail since $P_1$ and $P_2$ are disconnected. Similarly, the
(a) Without replication

(b) With replication

Figure 1: The effect of data replication in a MANET

data access to $D_1$ from mobile devices in $P_2$ will also fail.

Data replication is a promising technique to improve data accessibility and system performance in distributed database systems. However, in the literature of mobile computing, only a few works addressed the issues of data replication in MANETs. Authors in [36] and [37] addressed the problem of the update of replicas in a MANET. However, each node is expected to replicate all data items in the MANET whose practicability may need further justifications since the memory space is a scarce resource of mobile devices. Note that with the global topology of a fixed network, the replica allocation problem can be transformed into a minimum K-median problem which is NP-hard [14][29]. Clearly, the problem of replica allocation in a MANET is even more complicated since the network topology changes frequently. Authors in [9] proposed the scheme DCG to address the problem of replica allocation in a MANET. The experiments in [9] show that data replication can greatly improve the data accessibility for a partitioned network. In addition, scheme E-DCG was proposed in [11] to extend scheme DCG by considering the periodic update of data items.

In reality, the moving behavior of mobile users is usually regular and follows some mobility patterns [28][35]. Group mobility refers to the scenario where several mobile nodes tend to move together. In practice, group mobility usually occurs in collaborative works [12][19]. For example, a group of visitors visiting an art gallery with the same guide has similar movement behavior. In addition, in military, a team of soldiers moves together to accomplish a task. As a result, a number of research works have been elaborated upon the impact of group mobility [2][26][27][33][34]. Although reducing the degradation of data accessibility to some extent, both scheme DCG and scheme E-DCG have the following two
major drawbacks.

1. *Generation of a large amount of traffic.*

   In essence, because of requiring the global network connectivity, both schemes are *centralized* and require all data nodes to broadcast their information to all other nodes, which will undesirably cause a significant amount of network traffic. In this paper, we call this phenomenon as *blind flooding*. This situation is more severe in a MANET due to the low network bandwidth.

2. *Negligence of group mobility.*

   Consider the scenario shown in Fig. 1b. Suppose that $M_1$ and $M_3$ are in the same mobility group and $M_2$ is in another. Also assume that both groups are of different moving behaviors. Therefore, $M_1$ and $M_2$ tend to disconnect in the near future and only allocating $D'_2$ in $M_2$ will lead to an unsuccessful access to $D_2$ from $M_1$. As a result, replica allocation schemes without considering group mobility are not able to allocate replicas effectively.

   In view of this, we address in this paper the problem of replica allocation by exploring group mobility. The underlying group mobility model is assumed to be RPGM (standing for Reference Point Group Mobility model). By analyzing the characteristic of the employed group mobility model, several theoretical results are derived. In light of these results, we propose a replica allocation scheme (referred to as scheme DRAM) to allocate replicas by considering group mobility. To avoid blind flooding, scheme DRAM takes a *bottom-up* approach without requiring the global network connectivity. In scheme DRAM, each mobile node first exchanges its motion behavior with some neighbors. The coverage of the information exchange is limited by a predetermined parameter. Then, a *decentralized* clustering algorithm is proposed to cluster mobile nodes with similar motion behavior into mobility groups. Hence, clusters which are likely to connect with one another later will be merged into an allocation unit to save the aggregate storage cost. Finally, data items are replicated according to the resulting allocation units. Moreover, scheme DRAM maintains the mobility groups in an *adaptive* manner which keeps the number of information broadcasts as small as possible and hence reduces the generated network traffic. These characteristics distinguish our work from others. To evaluate the performance of the proposed schemes, several experiments are conducted. The experimental results
show that scheme DRAM is able to not only achieve higher data accessibility but also produce less network traffic than prior schemes.

The rest of this paper is organized as follows. Section 2 describes the employed mobility and system models. The proposed replica allocation scheme DRAM is described in Section 3. The performance study of DRAM is given in Section 4, and finally, Section 5 concludes this paper.

2 Preliminaries

2.1 Mobility Models

Several mobility models have been proposed to model the mobility behavior of mobile nodes [5][15], and Waypoint mobility model [16] is a widely-used one. Waypoint mobility model breaks the entire movement of a mobile node into repeating pause and motion periods. In a pause period, a mobile node will stay at the current location for a certain time period. When changing to a new movement period, one node will move toward a new random-chosen destination at a speed uniformly distributed between $[v_{Min}, v_{Max}]$. After arriving the destination, the mobile node will enter another pause period.

In real cases, mobile nodes may collaborate and hence move as a group instead of independently. RPGM is proposed in [12] to model this kind of team collaboration behavior. RPGM and its variations [19][20][34], are widely used to model the group mobility in a MANET [12][26][27][33][34]. In RPGM, all mobile nodes are divided into several mobility groups and all mobile nodes within the same mobility group are of similar moving behavior. Suppose that the time is divided into several slots, $T(1)$, $T(2)$, · · ·. Each time slot is of equal time interval $\delta$. Each node is assigned to a virtual reference node. The movement of a reference node in a time slot is called a global motion vector. The global motion vectors of all reference nodes in the same mobility group are the same. The location of a mobile node is uniformly distributed in the circle centered by the location of the corresponding reference node with radius $\epsilon$. For each mobile node, the vector from the position of the corresponding reference node to the position of the mobile node is called a random motion vector.

Fig. 2 shows an example movement of the mobile nodes (i.e., $M_1$, $M_2$ and $M_3$) of a mobility group which follows RPGM model in a motion period. In each time slot, each black node represents a mobile
node and the gray node with equal number represents the corresponding reference node. The dotted circle centered by a reference node is the possible area where the mobile node will actually appear. Let $P^N_i(k)$ and $P^R_i(k)$ represent the positions of the mobile node $M_i$ and the corresponding reference node in time $T(k)$, respectively. The global and random motion vectors of $M_3$ are $\overrightarrow{G M_3}(k)$ and $\overrightarrow{R M_3}(k)$, respectively. We then have

$$P^R_i(k + 1) = P^R_i(k) + \overrightarrow{G M_i}(k)$$

and

$$P^N_i(k + 1) = P^R_i(k + 1) + \overrightarrow{R M_i}(k + 1).$$

2.2 System Model

Suppose that there are $m$ mobile devices, $M_1, M_2, \cdots, M_m$. Each device stores and acts as the original host of several data items. There are $n$ data items, $D_1, D_2, \cdots, D_n$. Similar to [9], all data items are of equal size, and each data item is held by a particular mobile host as its original host. In addition to original data items, each mobile host has an extra memory space to store $S_{Rep}$ replicas. Similar to [11], each data item $D_i$ is updated by its original host periodically with period $\tau_i$. The access frequency of mobile host $M_i$ to data item $D_j$ is assumed to be $f_{ij}$ which is known in advance [9][10][11]. We assume that each node is equipped with a GPS device, and hence, the position of the node is always available. It is possible that not all mobile nodes are equipped with GPS devices, and under such case, scheme APS [23][24] can be employed for the mobile nodes without GPS devices to obtain their locations.

In this paper, we take RPGM as the group mobility model and the movement of each group follows
a waypoint model. The speed of a group within a motion period will be a constant which follows a uniform distribution over \([v_{Min}, v_{Max}]\). We also assume that the lengths of pause and motion periods are two exponential distributions with mean \(\mu_{Pause}\) and \(\mu_{Motion}\) time slots, respectively. For a mobile node \(M_i\) in time \(T(k)\), we have a global motion vector \(\vec{GM}_i(k)\) and random motion vector \(\vec{RM}_i(k)\).

Let \(|\cdot|\) represent the length of a vector, and then we have \(v_{Min} \times \delta \leq |\vec{GM}_i(k)| \leq v_{Max} \times \delta\) in each motion period and \(|\vec{GM}_i(k)| = |\vec{RM}_i(k)| = 0\) in each pause period.

3 Design of Decentralized Replica Allocation with Group Mobility

In this section, by considering group mobility we propose a decentralized algorithm, scheme DRAM (standing for Decentralized Replica Allocation with group Mobility), to produce effective replica allocation efficiently. Section 3.1 gives an overview of scheme DRAM. Two major phases of scheme DRAM, the allocation unit construction and replica allocation phases, are described in Sections 3.2 and 3.3, respectively.

3.1 An Overview

Since the network connectivity changes frequently, scheme DRAM is executed periodically with period \(r\) time slots to adapt the replica allocation according to the network connectivity. This time period is referred to a relocation period. The tasks for each mobile node in a relocation period consist of two major phases: the allocation unit construction phase and the replica allocation phase.

In the allocation unit construction phase, all mobile nodes in the network are divided into several disjoint allocation units. The formal definition of an allocation unit is as follows.

![Figure 3: The state transition diagram of mobile nodes in scheme DCG](image)
Table 1: Description of symbols

<table>
<thead>
<tr>
<th>Description</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of mobile nodes</td>
<td>m</td>
</tr>
<tr>
<td>Number of data items</td>
<td>n</td>
</tr>
<tr>
<td>i-th mobile node</td>
<td>M_i</td>
</tr>
<tr>
<td>j-th data item</td>
<td>D_j</td>
</tr>
<tr>
<td>Access frequency of M_i to D_j</td>
<td>f_{ij}</td>
</tr>
<tr>
<td>The k-th time slot</td>
<td>T(k)</td>
</tr>
<tr>
<td>Position of M_i in T(k)</td>
<td>P_N^i(k)</td>
</tr>
<tr>
<td>Position of the reference point of M_i in T(k)</td>
<td>P_R^i(k)</td>
</tr>
<tr>
<td>Global motion vector of M_i in T(k)</td>
<td>GM_i(k)</td>
</tr>
<tr>
<td>Random motion vector of M_i in T(k)</td>
<td>RM_i(k)</td>
</tr>
<tr>
<td>Update period of D_i</td>
<td>τ_i</td>
</tr>
<tr>
<td>Maximal length of random motion vector</td>
<td>ϵ</td>
</tr>
</tbody>
</table>

**Definition 1:** An *allocation unit* is a set of mobile nodes which share their storage and do not store repeated data item unless all data items have been allocated in this allocation unit.

Then, in the following replica allocation phase, the replicas of all data items are allocated in accordance with the access frequencies of the data items and the derived allocation units of the previous allocation unit construction phase.

Clearly, the more mobile nodes each allocation unit contains, the higher data accessibility we have. However, an *unstable* allocation unit which will be separated into disconnected partitions later will not take the advantage of storage sharing which will in turn cause severe degradation of data accessibility. Although setting a shorter relocation period can reduce the degradation in data accessibility, this approach will produce more network traffic. Therefore, constructing *large* and *stable* allocation units is important to achieve high data accessibility.

In the allocation construction phase, scheme DRAM constructs the allocation units in a bottom-up manner by considering the group mobility. In addition to producing less traffic than prior schemes by avoiding blind flooding, scheme DRAM is able to construct more stable allocation units since mobile nodes in each resulting allocation unit are likely connected later. Moreover, scheme DRAM maintains the mobility groups in an adaptive manner which keeps the number of information broadcasts as small as possible and hence reduces the generated network traffic. The details of scheme DRAM are described in the following subsections. For better readability, a list of symbols used is shown in Table 1.
3.2 Allocation Unit Construction Phase

Figure 3 shows the state transition diagram of mobile nodes in scheme DRAM. The tasks of mobile nodes in these states are described in the following subsections.

3.2.1 INITIAL State

Before describing the task of nodes the INITIAL state, we first define a broadcast zone as follows.

Definition 2: The broadcast zone of a mobile node \( M_i \) is a set of mobile nodes whose distances to \( M_i \) are smaller than or equal to a predefined \( TTL \).

When a mobile node wants to join the MANET, it will enter the INITIAL state. In addition, a mobile node in the CLUSTER-MASTER state will enter the INITIAL state when there is no member of its cluster within its broadcast zone. A mobile node in the CLUSTER-MEMBER state will also enter the INITIAL state when its mobility group master is not within its broadcast zone. A detailed description of the CLUSTER-MASTER and CLUSTER-MEMBER states will be given in Section 3.2.3.

A mobile node in the INITIAL state will broadcast an info messages to all mobile nodes in its broadcast zone. Unlike scheme E-DCG, each info message contains a time-to-live which is a predetermined constant \( TTL \). When a mobile host \( M_i \) receives an info message of \( M_j \), \( M_i \) will accept this info message and forward this message to all its neighbors if the distance between \( M_i \) and \( M_j \) is smaller than or equal to \( TTL \). Otherwise, \( M_i \) will discard the message. Therefore, only mobile nodes connected to \( M_j \) with distance smaller than or equal to \( TTL \) (i.e., those mobile nodes within the broadcast zone of \( M_j \)) will receive the info message of \( M_j \). The employment of \( TTL \) can avoid blind flooding and produce less network traffic.

Let an actual motion vector \( \overrightarrow{AM_i}(x, y) \) represent the motion of \( M_i \) from time \( T(x) \) to \( T(y) \) (i.e., the vector from \( P_i^N(x) \) to \( P_i^N(y) \)). Suppose that \( T(k - 1) \) and \( T(k) \) are within the same motion period, and consider the movement scenario of \( M_i \) as shown in Fig. 4. Since \( 0 \leq |\overrightarrow{RM_i}(k)| \leq \epsilon \) always holds for all mobile host \( M_i \), we have:

\[
|\overrightarrow{AM_i}(k - 1, k)| \geq |\overrightarrow{GM_i}(k - 1)| - 2\epsilon,
\]
Assume that $|\overrightarrow{GM_i(k-1)}| > 2\epsilon$, and hence $|\overrightarrow{AM_i(k-1)}| > 0$ when $T(k-1)$ and $T(k)$ are in the same motion period. Since the speed of each mobile node is zero only in pause periods, each mobile node $M_i$ is in the same pause period from $T(x)$ to $T(y)$ if and only if $P_i^N(x) = P_i^N(x + 1) = \cdots = P_i^N(y)$. Therefore, the pause and motion periods of each mobile node can be recognized from its historical locations.

To achieve this, each mobile node $M_i$ maintains a list of its historical locations (i.e., $P_i^N(k)$) called a position list. For each time slot, $M_i$ will retrieve its current location, denoted as $P_i^N(k_1)$, from its GPS device and insert it into the position list. Owing to the limitation in storage, the capacity of a position list is limited. Suppose that the maximal capacity of a position list is $S_{List}$, the current time stamp is $T(k_1)$, and the largest time stamp among all elements in the position list is $T(k_2)$, where $k_2 < k_1$. If $P_i^N(k_1) = P_i^N(k_2)$, $P_i^N(k_1)$ is discarded since $T(k_1)$ and $T(k_2)$ are in the same pause period. Otherwise, $P_i^N(k_1)$ is inserted into the position list. If the position list is full, the element with the smallest time stamp will be removed.

In the method to store and process historical location information mentioned above, only a small portion of historical location information (the location information of the last period) is required by scheme DRAM. Hence, it is unnecessary to store all historical location information. In addition, owing to the employment of $S_{List}$, the volume of stored historical location information is limited and can be controlled by mobile nodes.
<table>
<thead>
<tr>
<th>time-stamp ((k))</th>
<th>2</th>
<th>6</th>
<th>7</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>location ((P_{iN}^N(k)))</td>
<td>(3,3)</td>
<td>(4,3.5)</td>
<td>(5,4)</td>
<td>(4,4.5)</td>
<td>(3,5)</td>
<td>(2,5.5)</td>
<td>(1,6)</td>
<td>(0,6.5)</td>
<td>(-1,7)</td>
</tr>
</tbody>
</table>

Current time-stamp: 15

Figure 5: An example position list

For each mobile node, an info message contains the following fields which can be obtained from its position list.

- **id**: The host id.
- **\(t_s_{\text{Now}}\)**: The time stamp of the current relocation period.
- **\(P_{iN}^N(t_s_{\text{Now}})\)**: The location of \(M_i\) in time \(T(t_s_{\text{Now}})\).
- **\(t_s_{\text{Pause Start}}, t_s_{\text{Pause End}}\)**: The time stamps of the start and end time of the latest pause period.
- **\(t_s_{\text{Motion Start}}, t_s_{\text{Motion End}}\)**: The time stamps of the start and end time of the latest motion period.
- **\(\overrightarrow{AM}_i(t_s_{\text{Motion Start}}, t_s_{\text{Motion End}})\)**: The actual motion vector of the latest motion period.

**Example 1**: Consider an example position list shown in Fig. 5. This list records the positions of a mobile host \(M_i\) from \(T(2)\) to \(T(15)\). The motion of \(M_i\) can be divided into two pause periods (i.e., from \(T(2)\) to \(T(5)\) and from \(T(7)\) to \(T(9)\)) and two motion periods (i.e., from \(T(5)\) to \(T(7)\) and from \(T(9)\) to \(T(15)\)). Since \(P_{iN}^N(9)\) is not in the position list, we have \(P_{iN}^N(9) = P_{iN}^N(7) = (5,4)\) and \(\overrightarrow{AM}_i(9,15) = (-1-5,7-4) = (-6,3)\). As a result, the content of the info message broadcast by \(M_i\) in \(T(15)\) is: \(\{i; 15; (-1,7); 7; 9; 9; 15; (-6,3)\}\).

### 3.2.2 ZONE-MASTER and ZONE-MEMBER States

After broadcasting info messages, the mobile nodes are classified into two groups by the lowest-id clustering algorithm proposed in [7]. Each node in the INITIAL state whose host id is the smallest one among all nodes within its broadcast zone (obtained from the id field of all received info messages) is selected as the master of its broadcast zone and hence enters the ZONE-MASTER state. On the other hand, the other nodes enter the ZONE-MEMBER state. A node \(M_i\) in the ZONE-MEMBER state
will join the node $M_j$ in the ZONE-MASTER state with the smallest id among the nodes within the broadcast zone of $M_i$ by sending a \textit{join} messages to $M_j$.

Each node $M_i$ in the ZONE-MASTER state then clusters its member nodes (i.e., those node which send the \textit{join} messages to $M_i$) into clusters, and all nodes within a cluster are expected to have similar motion behaviors. The employed clustering algorithm is as follows. First, all nodes are clustered by the periods of the latest pause and motion periods recorded in \textit{info} messages since all mobile nodes in the same cluster are of equal records of latest pause and motion periods (i.e., have the same $t_{Start}^{Pause}$, $t_{Start}^{End}$, $t_{Start}^{Motion}$ and $t_{End}^{Motion}$ in their \textit{info} messages). However, several mobile nodes within different mobility groups (i.e., of different motion behaviors) may be placed into the same cluster especially when the length of a motion period is larger than the length of position lists. To achieve better results, the master node should re-cluster the resulting clusters again by considering motion vectors. With the scenario shown in Fig. 4, we have the following lemmas.

\textbf{Lemma 1:} \textit{The expectation of $\overline{AM}_i(k - 1, k)$ is equal to $\overline{GM}_i(k - 1)$}.

\textit{Proof:} Suppose that the polar coordinate representations of the random motion vectors in $T(k - 1)$ and $T(k)$ in Fig. 4 are $\overline{RM}_i(k - 1) = (r_1, \theta_1)$ and $\overline{RM}_i(k) = (r_2, \theta_2)$, respectively. Also let the density function of $r_1, r_2, \theta_1$ and $\theta_2$ be $f_{r_1}, f_{r_2}, f_{\theta_1}$ and $f_{\theta_2}$, respectively. We have

\[
f_{r_1}(a) = \frac{d}{da} \text{Prob}\{r_1 \leq \epsilon\} = \frac{d}{da} \frac{2\pi \times a^2}{2a \times 2\pi \epsilon^2} = \frac{a}{\epsilon^2} \quad \text{and} \quad f_{\theta_1}(b) = \frac{1}{2\pi}.
\]

Similarly, we also have $f_{r_2}(c) = \frac{2c}{\epsilon^2}$ and $f_{\theta_2}(d) = \frac{1}{2\pi}$.

Since $r_1, \theta_1, r_2$ and $\theta_2$ are independent, the joint density function of $r_1, \theta_1, r_2$ and $\theta_2$ is

\[
f_{r_1, \theta_1, r_2, \theta_2}(a, b, c, d) = f_{r_1}(a) \times f_{\theta_1}(b) \times f_{r_2}(c) \times f_{\theta_2}(d) = \frac{a \times c}{\pi^2 \epsilon^4}.
\]
Let the locations of $P_i^R(k-1)$ and $P_i^R(k)$ be $(x_1, y_1)$ and $(x_2, y_2)$, respectively. Clearly, the Cartesian coordinate of $\overrightarrow{GM}_i(k-1)$ is $\langle x_2 - x_1, y_2 - y_1 \rangle$. Suppose that $\overrightarrow{AM}_i(k-1, k) = \langle \alpha, \beta \rangle$. We have

$$E[r_2 \cos \theta_2 - r_1 \cos \theta_1] = \int_0^\epsilon \int_0^\epsilon \int_0^{2\pi} \int_0^{2\pi} \frac{a \times c}{\pi^2 \epsilon^4} (r_2 \cos \theta_2 - r_1 \cos \theta_1) d\theta_1 d\theta_2 dr_1 dr_2$$

$$= \int_0^\epsilon \int_0^\epsilon \frac{a \times c}{\pi^2 \epsilon^4} \left( r_2 \int_0^{2\pi} \cos \theta_2 d\theta_2 - r_1 \int_0^{2\pi} \cos \theta_1 d\theta_1 \right) dr_1 dr_2$$

$$= 0.$$

Then, we can derive the following equation:

$$E[\alpha] = E[\text{the x coordinate of } P_i^R(k) - \text{the x coordinate of } P_i^R(k-1)]$$

$$= E[x_2 + r_2 \cos \theta - (x_1 - r_1 \cos \theta)]$$

$$= x_2 - x_1 + E[r_2 \cos \theta_2 - r_1 \cos \theta_1]$$

$$= x_2 - x_1.$$

Similarly, we can also show that $E[\beta] = y_2 - y_1$, and finally, we have

$$E[\overrightarrow{AM}_i(k-1, k)] = E[\langle \alpha, \beta \rangle]$$

$$= \langle E[\alpha], E[\beta] \rangle$$

$$= \langle x_2 - x_1, y_2 - y_1 \rangle$$

$$= \overrightarrow{GM}_i(k-1).$$

From Lemma 1, we have the following observation. If the length of position lists (i.e., $S_{List}$) is large enough and the length motion period is long enough, the global motion vector of each mobile node can be approximated. However, it is impractical to store too many previous positions of a mobile node. Therefore, we propose a vector clustering algorithm below. To facilitate our presentation, we denote the global motion vector of $M_i$ from $T(k_1)$ to $T(k_2)$ (i.e., the vector from $P_i^R(k_1)$ to $P_i^R(k_2)$) as $\overrightarrow{V}_i(k_1, k_2)$. For $M_i$, if $T(k_1)$ and $T(k_2)$ are in the same motion period and $T(k_2) > T(k_1)$, we have $\overrightarrow{V}_i(k_1, k_2) = (k_2 - k_1) \times \overrightarrow{GM}_i(k_1)$. In addition, we represent motion vectors as polar coordinates (i.e.,
length-angle forms) and $T(k_1)$ and $T(k_2)$ are assumed to be in the same motion period and $k_2 > k_1$.

**Lemma 2:** For a mobile host $M_i$, the maximum absolute value of the difference between the angles of $\overrightarrow{AM_i}(k_1, k_2)$ and $\overrightarrow{V_i}(k_1, k_2)$ is

$$\left| \sin^{-1}\left( \frac{\epsilon}{(k_2 - k_1) \times v_{Min} \times \delta - \epsilon} \right) \right|,$$

where $v_{Min}$, $\delta$ and $\epsilon$ represent respectively the minimum speed of mobile nodes, the length of a time slot and the maximal length of random vectors.

**Proof:** Consider the scenario shown in Fig. 6. Let $\theta$ be the absolute value of the difference between the angles of $\overrightarrow{AM_i}(k_1, k_2)$ and $\overrightarrow{V_i}(k_1, k_2)$. $\theta$ is maximal when $P_i^N(k_1) = P_1$ and $P_i^N(k_2) = P_2$ or $P_3$. Since $\theta \geq 0$, we have

$$\sin(\theta) \leq \frac{\epsilon}{|\overrightarrow{V_i}(k_1, k_2)| - \epsilon} = \frac{\epsilon}{(k_2 - k_1) \times |\overrightarrow{GM_i}(k_1)| - \epsilon} \leq \frac{\epsilon}{(k_2 - k_1) \times v_{Min} \times \delta - \epsilon}$$

Finally, we have

$$0 \leq \theta \leq \left| \sin^{-1}\left( \frac{\epsilon}{(k_2 - k_1) \times v_{Min} \times \delta - \epsilon} \right) \right|.$$ 

Q.E.D.
Lemma 3: For a mobile host $M_i$, the maximal absolute value of the differences between the lengths of $\overrightarrow{AM}_i(k_1, k_2)$ and $\overrightarrow{V}_i(k_1, k_2)$ is $2\epsilon$.

Proof: The distance between $P^R_i(k_1)$ and $P^R_i(k_2)$ is

$$|\overrightarrow{V}_i(k_1, k_2)| = (k_2 - k_1) \times |\overrightarrow{GM}_i(k_1)|.$$ 

$\overrightarrow{AM}_i(k_1, k_2)$ is maximal when $P^N_i(k_1) = P_5$ and $P^N_i(k_2) = P_6$, and therefore we have

$$|\overrightarrow{AM}_i(k_1, k_2)| - |\overrightarrow{V}_i(k_1, k_2)| \leq |\overrightarrow{V}_i(k_1, k_2)| + 2\epsilon - |\overrightarrow{V}_i(k_1, k_2)| = 2\epsilon.$$ 

Similarly, $\overrightarrow{AM}_i(k_1, k_2)$ is minimal when $P^N_i(k_1) = P_1$ and $P^N_i(k_2) = P_4$. Again, we have the following equation:

$$|\overrightarrow{AM}_i(k_1, k_2)| - |\overrightarrow{V}_i(k_1, k_2)| \geq |\overrightarrow{V}_i(k_1, k_2)| - 2\epsilon - |\overrightarrow{V}_i(k_1, k_2)| = -2\epsilon.$$ 

Finally, we have $0 \leq |\overrightarrow{AM}_i(k_1, k_2)| - |\overrightarrow{V}_i(k_1, k_2)| \leq 2\epsilon$. Q.E.D.

For vectors $v_i = (r_i, \theta_i)$ and $v_j = (r_j, \theta_j)$, $v_j$ is said to be “a neighbor in angle with maximal difference $\theta$ of $v_j$” if $|\theta_i - \theta_j| \leq \theta$. Similarly, $v_j$ is said to be “a neighbor in length with maximal difference $\text{dist}$ of $v_j$” if $|r_i - r_j| \leq \text{dist}$. For each mobile node $M_i$, $\overrightarrow{AM}_i(k_1, k_2)$ is a neighbor in angle with maximal difference $\theta$ of $\overrightarrow{V}_i(k_1, k_2)$ where $\theta$ is equal to the result in Lemma 2. Similarly, according to Lemma 3, $\overrightarrow{AM}_i(k_1, k_2)$ is the neighbor in length with maximal difference $2\epsilon$ of $\overrightarrow{V}_i(k_1, k_2)$.

In a mobility group, the number of the neighbors in angle with maximal difference $\theta$ of $\overrightarrow{V}_i(k_1, k_2)$ is maximal among all actual motion vectors of all group members. Similarly, the number of the neighbors in length with maximal difference $2\epsilon$ of $\overrightarrow{V}_i(k_1, k_2)$ is maximal among all actual motion vectors of all group members. With the result of Lemma 1 and the above observations, we have the following two heuristics.

1. In a mobility group, an actual motion vector is close to the global motion vector if it has the
maximal number of neighbors in angle with maximal difference $\theta$.

2. In a mobility group, an actual motion vector is close to the global motion vector if it has the maximal number of neighbors in length with maximal difference $2\epsilon$.

In accordance with the above heuristics, we develop algorithm VectorCluster which is an efficient node clustering algorithm comprising two major procedures, ClusterByAngle and ClusterByLength. The algorithmic form of algorithm VectorCluster is as follows.

Algorithm VectorCluster($\mathcal{V}$)
\textbf{Input:} A set of vectors $\mathcal{V}$.
\textbf{Output:} A set of clusters of the vectors $\mathcal{C}$.
\textbf{Note:} The input vectors are represented by polar coordinate forms, and each mobile host is represented by the actual motion vector in its info message.
1: Cluster the nodes by the periods of the time of the latest pause and motion periods in their info messages, and store the resulting clusters in $\mathcal{C}$.
2: Calculate the value of $\theta$ by the result in Lemma 2.
3: Re-cluster each cluster $C_i$ in $\mathcal{C}$ by invoking procedure ClusterByAngle($C_i, \theta$), and replace $C_i$ in $\mathcal{C}$ by the resulting clusters.
4: Re-cluster each cluster $C_i$ in $\mathcal{C}$ by invoking procedure ClusterByLength($C_i, 2\epsilon$), and replace $C_i$ in $\mathcal{C}$ by the resulting clusters.
5: \textbf{return} $\mathcal{C}$;

Procedure ClusterByAngle($\mathcal{V}, \theta$)
\textbf{Input:} A set of vectors $\mathcal{V}$ and an angle $\theta$.
\textbf{Output:} A set of clusters of the vectors $\mathcal{C}$.
1: Reset $\mathcal{C}$ to empty set.
2: Calculate the number of neighbors in angle with difference $\theta$ for each vector $v_i$ in $\mathcal{V}$.
3: \textbf{while} ($\mathcal{V}$ is not empty) \textbf{do}
4: Let $v$ be the vector with the largest number of neighbors in angle in $\mathcal{V}$. Insert the set containing $v$ and its neighbors in angle into $\mathcal{C}$.
5: Remove $v$ and its neighbors in angle from $\mathcal{V}$.
6: \textbf{return} $\mathcal{C}$;

Procedure ClusterByLength($\mathcal{V}, \text{dist}$)
\textbf{Input:} A set of vectors $\mathcal{V}$ and a distance $\text{dist}$.
\textbf{Output:} A set of clusters of the vectors $\mathcal{C}$.
1: Reset $\mathcal{C}$ to empty set.
2: Calculate the number of neighbors in length with difference $\text{dist}$ for each vector $v_i$ in $\mathcal{V}$.
3: \textbf{while} ($\mathcal{V}$ is not empty) \textbf{do}
4: Let $v$ be the vector with the largest number of neighbors in length in $\mathcal{V}$. Insert the set containing $v$ and its neighbors in length into $\mathcal{C}$.
5: Remove $v$ and its neighbors in length from $\mathcal{V}$.
6: \textbf{return} $\mathcal{C}$;

After executing the algorithm VectorCluster, each zone master will select one cluster master for each resulting cluster. Then the selected mobile nodes will enter the CLUSTER-MASTER state, and
the other nodes will enter the CLUSTER-MEMBER state.

**Example 2:** Consider the scenario shown in Fig. 7. Clustering by the time of latest pause and motion periods (i.e., line one in Algorithm VectorCluster) will form two clusters: \{M_1, M_2, \ldots, M_{10}\} and \{M_{11}\}. M_{11} is separated from other nodes since it has different latest pause and motion periods.

Then, \{M_1, M_2, \ldots, M_{10}\} will be re-clustered by angle. The actual motion vectors of \{M_1, M_2, \ldots, M_{10}\} are drawn in Fig. 8 in a counterclockwise order. Assume that \(v_{Min} = 5, v_{Max} = 7\) and \(\epsilon = 2\). According to Lemma 2, we have \(\theta = |\sin^{-1}(\frac{2}{5x3-2})| = 0.1545\). The number of neighbors in angle with maximal difference 0.1545 for each vector is then calculated. \(v_1\) is a neighbor of \(v_3\) since the absolute value of the difference between angles of them is \(|0.42 - 0.56| = 0.14 < 0.1545\). On the other hand, \(v_5\) is not a neighbor of \(v_3\) since the absolute value of the difference between angles of them is \(|0.9 - 0.56| = 0.34 > 0.1545\). Since the actual motion vector of \(M_7\) (i.e., \(v_7\)) is of the maximal number of neighbors, \(M_7\) and its neighbors in angle form a cluster. The rest actual motion vectors are processed analogously. The results of procedure ClusterByAngle on \{M_1, M_2, \ldots, M_{10}\} are \{M_1, M_2, M_3, M_4\} and \{M_5, M_6, \ldots, M_{10}\} as shown in Fig. 8a.

All resulting clusters of procedure ClusterByAngle are re-clustered again by procedure ClusterByLength. From Lemma 3, we have \(dist = 2\epsilon = 4\). Similarly, the number of neighbors in length with maximal difference 4 for each motion vector is calculated. \(v_5\) is a neighbor of \(v_7\) since the absolute value of the difference between the lengths of them is \(|15.7 - 15.1| = 0.6 < 4\). On the other hand, \(v_6\) is not a neighbor of \(v_5\) since \(|31.2 - 15.1| = 16.1 > 4\). Therefore, \{M_5, M_6, \ldots, M_{10}\} is re-clustered into
(a) The results of procedure ClusterByAngle

(b) The results of procedure ClusterByLength

(c) The results of algorithm VectorCluster

Figure 8: An example of motion vector clustering on \{M_1, M_2, \ldots, M_{10}\} and \{M_5, M_7, M_{10}\} by procedure ClusterByLength as shown in Fig. 8b. The result of algorithm VectorCluster on \{M_1, M_2, \ldots, M_{10}\} is shown in Fig. 8c, and finally, these 11 mobile nodes are clustered into four clusters: \{M_1, M_2, M_3, M_4\}, \{M_5, M_7, M_{10}\}, \{M_6, M_8, M_9\} and \{M_{11}\}.

3.2.3 CLUSTER-MASTER and CLUSTER-MEMBER States

The tasks of cluster masters and members consist of two steps, cluster maintenance and cluster merge, which are described below.

**Cluster Maintenance:** Mobile nodes which enter the CLUSTER-MASTER and CLUSTER-MEMBER states in current allocation unit construction phase will skip the cluster maintenance step and execute the cluster merge step in this allocation unit construction phase. The other mobile nodes will execute cluster maintenance step before executing cluster merge step. Since the clusters are constructed by considering group mobility, each cluster is likely to be connected in the near future. These clusters are maintained in the following adaptive fashion.

Each cluster member first sends a status message to its cluster master. The contents of status messages are similar to those of info messages. If the mobile node and its cluster master are disconnected, the mobile node will leave the current cluster and enter the INITIAL state. At the same time, the cluster master will remove the mobile hosts which do not send status messages to it from its member list.

The cluster master then checks whether the moving behaviors recorded in received status messages are similar to one another by the following procedure. The cluster master first clusters the motion behaviors stored in received status messages by the clustering algorithm proposed in Section 3.2.2. If
each cluster member keeps similar motion behavior to that it used to have, the result of the execution of the clustering algorithm is one cluster which contains all cluster members. If some cluster members change their motion behaviors, the result will consist of several clusters. Let the dominating cluster of the resulting clusters be the cluster with the most nodes among all resulting clusters. Those mobile nodes not in the dominating cluster will then receive reject messages from the cluster master, and enter the INITIAL state. Note that only mobile nodes with different moving behavior from the majority of the original cluster will not be in the dominating cluster. If the original cluster master is not in the dominating cluster, it will assign one node in the dominating cluster as the new cluster master. The new cluster master will then send reject messages to the mobile nodes which are not in the dominating cluster.

Except the first execution of scheme DRAM, most mobile nodes will be in the CLUSTER-MASTER or CLUSTER-MEMBER state since only mobile nodes disconnected with its cluster master or with different moving behavior from its cluster members will enter the INITIAL state. Therefore, the successive execution of scheme DRAM will produce less network traffic than the first execution because only the first execution of scheme DRAM requires all mobile nodes to broadcast messages. As shown in Section 4, the adaptive cluster maintenance is able to effectively reduce produced network traffic.

**Cluster Merge:** If two mobile nodes, \( M_1 \) and \( M_2 \), tend to connect to each other, replicating one data item in \( M_1 \) also makes this data item accessible for \( M_2 \). Hence, \( M_2 \) can replicate another data item which will be also accessible for \( M_1 \). As mentioned in Section 3.1, merging clusters which tend to be connected in the near future into a big allocation unit makes the mobile nodes within different clusters share their storage and hence improve the data accessibility.

In replica allocation construction, only the resulting clusters which are likely connect with one another until the next execution of scheme DRAM will be merged into the same allocation unit. Initially, each resulting cluster is assigned to be an allocation unit. Assume that the current and next executions of scheme DRAM are in \( T(k) \) and \( T(k + r) \), respectively. Then, we have the following definitions.

**Definition 3:** The *bounding rectangle* of an allocation unit is a rectangle which contains all nodes of the allocation unit. Denote the bounding rectangle of cluster \( C_i \) in \( T(k) \) as \( BR_i(k) \). Let \( BR_i(k) \) be
represented by four values: \(top_i(k), bottom_i(k), left_i(k)\) and \(right_i(k)\). Then, we have

\[
\begin{align*}
top_i(k) &= \max_{\forall M_j \in C_i} \{\text{the y coordinate of } M_j \text{ in } T(k)\} \\
bottom_i(k) &= \min_{\forall M_j \in C_i} \{\text{the y coordinate of } M_j \text{ in } T(k)\} \\
left_i(k) &= \min_{\forall M_j \in C_i} \{\text{the x coordinate of } M_j \text{ in } T(k)\} \\
right_i(k) &= \max_{\forall M_j \in C_i} \{\text{the x coordinate of } M_j \text{ in } T(k)\}
\end{align*}
\]

**Definition 4:** Let \(BR^*_i(k+r)\) be the estimated bounding rectangle of the allocation unit \(C_i\) in \(T(k+r)\). Assume that the latest info message of the master node is in \(T(k)\), we have

\[
\begin{align*}
top^*_i(k+r) &= top_i(k) + r \times y \\
bottom^*_i(k+r) &= bottom_i(k) + r \times y \\
left^*_i(k+r) &= left_i(k) + r \times x \\
right^*_i(k+r) &= right_i(k) + r \times x,
\end{align*}
\]

where

\[
\langle x, y \rangle = \frac{\text{the average of all } \overrightarrow{AM}_i(ts_{Motion\ Start}^i, ts_{Motion\ End}^i) \text{ in } C_i}{ts_{End}^i - ts_{Start}^i}.
\]

**Definition 5:** Two allocation units \(C_i\) and \(C_j\) are said to be **cluster-wise connected** in \(T(k)\) if

1. \(BR_i(k)\) and \(BR_j(k)\) are overlapped, or
2. there exists another allocation unit \(C_x\) so that \(BR_i(k)\) and \(BR_j(k)\) are overlapped with \(BR_x(k)\), respectively.

**Definition 6:** Two allocation units \(C_i\) and \(C_j\) are said to be **potentially cluster-wise connected** in \(T(k+r)\) if

1. \(BR^*_i(k+r)\) and \(BR^*_j(k+r)\) are overlapped, or
2. there exists a \(C_x\) so that \(BR^*_i(k+r)\) and \(BR^*_j(k+r)\) are overlapped with \(BR^*_x(k+r)\), respectively.

We observe that two allocation units \(C_i\) and \(C_j\) can be **merged** into a new allocation unit if \(C_i\) and \(C_j\) are cluster-wise connected in \(T(k)\) and potentially cluster-wise connected in \(T(k+r)\).
Example 3: Fig. 9 shows the bounding rectangles of cluster $C_1$ and $C_2$ in $T(k)$ and the estimated bounding rectangles of $C_1$ and $C_2$ in $T(k + r)$. Although the moving behaviors of $C_1$ and $C_2$ (i.e., $GM_1(k)$ and $GM_2(k)$) are much different, $C_1$ and $C_2$ are still cluster-wise connected in $T(k)$ and potentially cluster-wise connected in $T(k + r)$. As a result, they should be merged into an allocation unit in order to increase the data accessibility in $C_1$ and $C_2$.

In replica allocation construction, each cluster master will broadcast a merge message containing the cluster master id, the current and estimated bounding rectangles. Note that the merge relation is an equivalence relation since it is reflective, symmetric and transitive. The cluster merge process can be performed by the following procedure.

```
Procedure ClusterMerge($C$)
Input: A set of cluster.
Output: The merged clusters of the input clusters.
1: Calculate $BR_i(k)$ and $BR_i^*(k + r)$ for each cluster $C_i$ in $C$;
2: Reset $R$ to empty set.
3: for $i = 1$ to $|C|$ do
4:   Insert $\{(C_i, C_i)\}$ into $R$. /* Reflective relation */
5:   for $j = i$ to $|C|$ do
6:     if ($BR_i(k)$ and $BR_j(k)$ are overlapped and $BR_i^*(k + r)$ and $BR_j^*(k + r)$ are also overlapped) then
7:       Insert $\{(C_i, C_j), (C_j, C_i)\}$ into $R$. /* Symmetric relation */
6:     end if
5:   end for
4: end for
1: Calculate the transitive closure of the relation $R$ (denoted as $R^*$).
9: return the partitions of $R^*$;
```

3.3 Replica Allocation Phase

In order to compare to [11], the objective of replica allocation phase is to identify data items to be replicated, and the locations to replicate them for each allocation unit in order to maximize the data...
accessibility. The employed replica allocation algorithm is as follows. Let the allocation weight of data item \( D_j \) in allocation unit \( C_x \) in \( T(k) \) (denoted as \( w^x_j(k) \)) be from now on, the expected number of data access from all mobile nodes in \( C_x \) before the next update of \( D_j \). Suppose that the current time stamp is \( T(k) \), \( w^x_j(k) \) can be obtained by the following equation:

\[
  w^x_j(k) = f^x_j \times (U_j - k), \text{ where } f^x_j = \sum_{M_i \in C_x} f_{ij},
\]

and \( U_j \) is the time-stamp of the next update of \( D_j \). Since the update of each data item is periodic, \( U_j \) can be predicted in advance.

All data items are allocated in \( C_x \) according to their allocation weights in \( C_x \) in descendent order. Let the allocation candidate set of \( D_j \) in \( C_x \) be a set of mobile hosts in \( C_x \) where each mobile host in the set is of available storage and with at least one path to the owner of \( D_j \). If the candidate set of \( D_j \) in \( C_x \) is not empty, \( D_j \) will be allocated to \( M_i \) where \( f_{ij} \) is the largest among all mobile hosts in the allocation candidate set of \( D_j \). Otherwise, \( D_j \) will be skipped. The allocation process completes if all mobile hosts in \( C_x \) is full. Each unit master then executes the following replica allocation process as below.

```
Procedure ReplicaAllocation(C)
Input: A set of allocation units C.
1: for (each data item \( D_j \)) do
2:   if (there exists at least one path between the owner of \( D_j \) and \( C_x \)) then
3:     Calculate allocation weight (i.e., \( w^x_j(k) \)) for \( D_j \) in \( C_x \) and insert the identifier of \( D_j \) into \( D \)
4: while (\( D \) is not empty and there are unused space in \( C \)) do
5:   Let \( D_j \) be the data item with largest allocation weight in \( D \).
6:   Calculate the allocation candidate set of \( D_j \) in \( C_x \)
7:   if (the allocation candidate set of \( D_j \) in \( C_x \) is not empty) then
8:     Let \( M_i \) be the mobile host where \( f_{ij} \) is the largest among all mobile hosts in the allocation candidate set of \( D_j \).
9:     Allocate \( D_j \) to \( M_i \)
10: Remove \( D_j \) from \( D \).
```

### 3.4 Complexity Analysis

In algorithm ClusterVector, the complexity of clustering vectors by their time of the latest pause and motion periods is \( O(|V|) \). In procedure ClusterByAngle, to calculate the number of neighbors in angle, these vectors are sorted according their angles and complexity of sorting is \( O(|V| \log |V|) \) where
$|\mathcal{V}|$ is the number of input vectors. After calculation, these vectors are sorted again by their number of neighbors. The clustering process (i.e., the loop in line 3 - 6) requires one scan of these vectors. Therefore, the complexity of procedure ClusterByAngle is $O(|\mathcal{V}| \log |\mathcal{V}|)$. Similarly, the complexity of procedure ClusterByLength is also $O(|\mathcal{V}| \log |\mathcal{V}|)$. As a result, the complexity of algorithm VectorCluster is $O(|\mathcal{V}| \log |\mathcal{V}|)$.

In procedure ReplicaAllocation, the calculation of allocation weight for each data item requires one scan of $f_{ij}$ for each data item $D_j$ and each mobile node $M_i$, and its complexity is $O\left(\frac{m}{|C|}\right)$ where $\frac{m}{|C|}$ is the average number of mobile nodes in an input cluster. The replica allocation process (i.e., the loop in line 3-6) will traverse all data items once, and its complexity is $O(n)$. Finally, the complexity of procedure ReplicaAllocation is $O\left(\frac{m}{|C|} + n\right)$.

### 3.5 Remarks

Authors in [20] proposed two pattern recognition-based algorithms to recognize mobility groups, and devised a service replication algorithm to provide continuous multimedia streaming services. Different from this paper, the work in [20] is based on RVGM (Reference Velocity Group Mobility) which is a variation of RPGM. In addition, the mobility recognition algorithms proposed in [20] are centralized, and similarly to scheme E-DCG, will be prone for having much network traffic.

Authors in [38] stated that although being able to achieve high data accessibility, storage sharing among neighboring mobile nodes also increases query delay. Hence, authors in [38] proposed scheme RN to balance the tradeoffs between data accessibility and query delay based on the following two concepts:

1. Each mobile node shares only part of its storage with its neighbors.

2. A mobile node $M_i$ only cooperates with its neighbors which tend to be *directly* connected with $M_i$ in the future.

The work in [38] is indeed complementary to the work in this paper. When data accessibility and query delay are required to be considered together, it is easy to integrate the above concepts into scheme DRAM to balance the tradeoffs between data accessibility and query delay.
4 Performance Evaluation

4.1 System Model

Although the notion of exploiting group mobility to data replication was mentioned in [13], the performance of scheme DRAM on several parameters was yet to be investigated. Consequently, we implemented an event-driven simulator in C++ with SIM [3] to evaluate the performance of scheme DRAM on various parameters. Scheme E-DCG is also implemented for comparison purposes. We assume that there are 120 mobile nodes in a $50m \times 50m$ flatland and each node owns 20 data items. The system parameters are shown in Table 2. Similar to [9], the access frequency for each data item, $f_{ij}$, is determined as a positive value based on the normal distribution with mean $0.025(1 + 0.01i)$ and standard derivation 0.0025. For both schemes, the revised RPGM described in Section 2.2 is taken as the underlying group mobility model of the simulation. The simulation is run for 3000 seconds.

Similar as [11] and [21], data accessibility is employed as the measurement of the performance of these schemes. When one mobile node $M_i$ issues a data request, the request is successful if the required data item is stored in $M_i$ or in another mobile node connected to $M_i$. Therefore, the data accessibility is defined as below:

$$\text{Accessibility} = \frac{\text{Number of successful requests}}{\text{Number of issued requests}}.$$  

We also use the produced network traffic to evaluate the cost of the employed schemes. Similar to [9], traffic is defined as the total hop counts of message transmission (e.g., info messages, join messages . . .) of all schemes.

4.2 The Effect of Relocation Period

In first experiment, we investigate the effect of the varied relocation period in data accessibility and traffic of all schemes. Fig. 10 shows the effects of scheme E-DCG and DRAM with the relocation period ranging from 200 to 800.

We observe from Fig. 10a that the data accessibility increases as the value of relocation period decreases. It can be explained that a shorter relocation period implies the more executions of relocation schemes, and therefore makes both replica relocation schemes be able to quickly adapt to the
Table 2: System parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>The number of mobile hosts ($m$)</td>
<td>120</td>
</tr>
<tr>
<td>The number of data items ($n$)</td>
<td>2400</td>
</tr>
<tr>
<td>The number of groups</td>
<td>10</td>
</tr>
<tr>
<td>Data size</td>
<td>16K Bytes</td>
</tr>
<tr>
<td>Max. no. of replicas to be stored in $M_i$ ($S_{Rep}$)</td>
<td>100</td>
</tr>
<tr>
<td>The min. speed of mobile nodes ($v_{Min}$)</td>
<td>3 m/sec.</td>
</tr>
<tr>
<td>The max. speed of mobile nodes ($v_{Max}$)</td>
<td>5 m/sec.</td>
</tr>
<tr>
<td>The avg. length of motion period ($\mu_{Motion}$)</td>
<td>5 m</td>
</tr>
<tr>
<td>The avg. length of pause period ($\mu_{Pause}$)</td>
<td>3 m</td>
</tr>
<tr>
<td>The max. length of random vectors ($\epsilon$)</td>
<td>1 m</td>
</tr>
<tr>
<td>Radio Radius ($R_{Radius}$)</td>
<td>5 m</td>
</tr>
<tr>
<td>Relocation period ($r$)</td>
<td>600 sec.</td>
</tr>
<tr>
<td>The update period for each data item ($\tau_i$)</td>
<td>500 sec.</td>
</tr>
<tr>
<td>Time-to-live of info messages ($TTL$)</td>
<td>10 m</td>
</tr>
</tbody>
</table>

mobility behavior of mobile nodes by frequently generating allocation units according to the network connectivity. We also observe that scheme DRAM outperforms scheme E-DCG in data accessibility. This result shows the importance of the consideration of group mobility. Since scheme E-DCG only considers the current network connectivity, the generated allocation units are unstable and may be partitioned soon. Hence the effect of scheme E-DCG is limited. On the contrary, the generated allocation units of scheme DRAM are stable since the generated allocation units are formed with mobility groups. Therefore, the data accessibility of scheme DRAM is higher than that of scheme E-DCG. In addition, the performance gain of scheme DRAM over scheme E-DCG in data accessibility increases as the relocation period decreases. The reason is that each execution of scheme DRAM can obtain better replica allocations than that of scheme E-DCG, and hence, a short relocation period strengthens the advantage of scheme DRAM over scheme E-DCG.

Fig. 10b shows the produced network traffic of scheme DRAM and E-DCG with the relocation period varied. We observe that the produced network traffic increases as the relocation period decreases. The reason is that with the same time interval, a shorter relocation period implies more executions of the relocation schemes, and hence implies more exchanged messages. It is also observed that the produced network traffic of scheme DRAM is smaller than that of scheme E-DCG. Due to the natural of blind flooding, the amount of network traffic produced by scheme E-DCG is rather large. On the contrary, scheme DRAM employs $TTL$ to avoid blind flooding in the exchange of $info$ messages and adopts a
decentralized and hierarchical approach to form allocation units. Hence, the amount of network traffic produced by scheme DRAM is less sensitive on the number of mobile nodes than that produced by scheme E-DCG.

In addition, the increment of the produced network traffic of scheme DRAM is less than that of scheme E-DCG when the value of relocation period decreases. It is because that scheme DRAM adaptively maintains the mobility groups in the cluster maintenance step without requiring each mobile node to broadcast its information in each relocation period as scheme E-DCG. Therefore, the successive executions of scheme DRAM will produce less network traffic than the first execution. Since a short relocation period implies frequent executions of relocation schemes, the network traffic reduction of scheme DRAM over scheme E-DCG increases as the value of relocation period decreases.

4.3 The Effect of the Number of Mobile Nodes

This experiment investigates the effect of the number of mobile nodes. The data accessibility and produced traffic of both schemes with the number of mobile nodes varied are shown in Fig. 11. The number of mobile nodes ranges from 40 to 200.

As observed in Fig. 11a, the data accessibility increases as the number of mobile nodes increases. It is because that with the same number of mobility groups, more mobile nodes imply that each mobility group is of more mobile nodes. Since the network connectivity of the mobile nodes within one mobility group is stable, more mobile nodes can share their storage by constructing large allocation units, and
as a consequence, increase the data accessibility. It is also shown that scheme DRAM outperforms scheme E-DCG in data accessibility. It is because that scheme DRAM can take advantage of group mobility to obtain higher data accessibility than scheme E-DCG by constructing larger and more stable allocation units than scheme E-DCG.

As observed in Fig. 11b, the amounts of produced traffic of all schemes increase as the number of mobile nodes increase. In addition, the amount of produced traffic of scheme E-DCG increases drastically as the number of mobile nodes increases. This is due to the characteristic of blind flooding employed in scheme E-DCG that the amount of produced network traffic increases exponentially as the number of mobile nodes increases. On the other hand, the amount of produced traffic of scheme DRAM is less sensitive than that of scheme E-DCG. The reason is that the TTL in info messages can effectively reduce the number of exchanged messages in the INITIAL state. In addition, the adaptive cluster maintenance in scheme DRAM is able to reduce the number of broadcast messages and therefore reduce the amount of produced network traffic. This result shows the scalability of scheme DRAM over scheme E-DCG.

### 4.4 The Effect of the Number of Mobility Groups

We investigate in this experiment the performances of all schemes under cases with different degree of group mobility. The effects in the accessibility and traffic for both schemes with the the number of mobility groups varied are shown in Fig. 12. The number of mobility groups is setting from 1 to 120.
In the case with 120 mobility groups, we set $\epsilon = 0$ so that this case degenerates to the case of waypoint mobility model.

We observe that the data accessibilities of all schemes are equal to 100% when the number of groups is set to one. That is because that in the case with one mobility group, the data accessibility is dependent on the initial network connectivity and the effect of the maximal length of random vectors. In this experiment, since the initial network topology is connected and the maximal length of random vectors is relatively smaller than radio radius, the network topology will not be partitioned and all mobile nodes are connected throughout the simulation, thereby having complete data accessibilities for both schemes. When the number of groups is set to two, the data accessibilities are reduced to around 55% since each mobility groups contains about half mobile nodes and the network topology is highly likely to be separated into two equally-sized partitions. These results show that the data accessibilities of scheme DRAM and scheme E-DCG are similar when the number of mobility groups is small.

When the number of mobility groups is larger than two, the data accessibilities of all schemes decrease as the number of mobility groups increases. It is because that with the same number of mobile nodes, the smaller number of mobile groups indicates that more mobile nodes are of similar moving behavior. Hence, more mobile nodes can share their storage by constructing large allocation units, and hence, increase the data accessibility. Moreover, as shown in Fig. 12a, scheme DRAM outperforms scheme E-DCG especially when the number of mobility groups is large (from four to 20 in this experiment). The larger number of mobility groups indicates that the movement behavior of all mobile nodes are less regular. Therefore, the allocation units obtained by scheme E-DCG are more unstable,
and hence, the data accessibility of scheme E-DCG degrades. On the other hand, scheme DRAM reduces the effect of the increment of the number of mobility groups by considering group mobility when constructing allocation units. As a result, the degradation in data accessibility of scheme DRAM is less significant than that of scheme E-DCG. This result shows that scheme DRAM outperforms scheme E-DCG when the number of mobility groups is medium. However, the performance gain of scheme DRAM over scheme E-DCG diminishes greatly when the number of mobility groups is large. Under these cases, since the network topology changes quickly and is highly likely to be partitioned into several disconnected partitions, the effect of replication diminishes significantly.

As observed in Fig. 12b, the amount of traffic produced by all schemes increases as the number of mobility groups decreases. When the number of mobility groups is large, the MANET tends to be separated into many disconnected partitions since fewer mobile nodes are of similar moving behavior. Therefore, the amount of produced traffic of all schemes is small since many mobile nodes disconnect with others when the MANET is separated into several partitions. In addition, we also observe that scheme DRAM produces less network traffic than scheme E-DCG. Due to adaptive procedures of scheme DRAM, the successive executions of scheme DRAM are able to execute in an adaptive manner, and hence produce less traffic than the first execution.

When the number of mobility groups is small, the MANET is often separated into several partitions, and each partition contains more mobile nodes than that in the case with many mobility groups. Due to the message exchange among adjacent mobile nodes, the produced traffic of all schemes are larger than that in the case with many mobility groups. With the employment of blind flooding, the amount traffic produced by scheme scheme E-DCG exponentially increases as the number of mobility groups increases. On the other hand, in scheme DRAM, due to the employment of TTL in info messages, the amount of message exchange in INITIAL state is greatly reduced. Therefore, the decrease of the number of mobility groups only slightly increases the amount of produced traffic.

4.5 The Effect of the Number of Replicas Per Node

In this subsection, we investigate the effect of the number of replicas per node which is set from 5 to 120. Since the number of replicas per node does not affect the produced traffic of all schemes, only
accessibility is shown in this experiment. Fig. 13a shows the effect of scheme E-DCG and DRAM in data accessibility. As shown in Fig. 13a, the data accessibility increases as the number of replicas per node increases. With a larger number of replicas per node, each mobile node has more space to store replicas, and each allocation unit has more shared storage. Moreover, the increment in data accessibility decreases when the number of replicas per node is large enough (i.e., 20 in this experiment) since data items with high access frequencies have mostly been replicated. Due to the consideration of group mobility, scheme DRAM utilizes the shared space better than scheme E-DCG by generating more stable allocation units, and hence, leads to better data accessibility than scheme E-DCG does.

4.6 The Effect of Update Period

In this subsection, we investigate the effect of update period which is set from 100 seconds to 500 seconds. The effect of scheme E-DCG and DRAM in data accessibility with the length of update period varied is shown in Fig. 13b. Similarly, only accessibility is shown in this subsection since the update period does not affect the traffic of all schemes.

As observed in Fig. 13b, the data accessibilities of all schemes decrease as the update period decreases. It is because that a shorter update period implies frequent updates on data items. As a result, a short update period makes each replica invalid soon, thereby reducing the effect of replication. In addition, the improvement of scheme DRAM over scheme E-DCG decreases as the update period decreases. This can be explained by the fact that quick invalidations of replicas incurred by short update period
will reduce the effect of data replication. As a consequence, although scheme DRAM is able to generate better replica allocation than scheme E-DCG, due to quick invalidation of replicas, the superiority of scheme DRAM over scheme E-DCG diminishes significantly as the update period decreases.

### 4.7 Effect of the Precision of Location Information

In this subsection, we investigate the effect of the precision of location information on accessibility, and the experimental result is shown in Fig. 14a. We assume that a GPS is likely to produce random errors in location information, and the error of location information is defined as the maximal distance between the actual location of a mobile node and the location reported by the GPS device of the mobile node. Intuitively, the smaller the error is, the more precise the location information is. Since scheme E-DCG does not rely on location information, its performance is not affected by precision of location information. On the other hand, the performance of scheme DRAM degrades as the error of information increases since scheme DRAM relies on location information to form stable allocation units.

However, as shown in Fig. 14a, the performance degradation of scheme DRAM is limited even the error is up to three meters. This phenomenon results from the effect of precision lists. Since a precision list stores multiple locations of the corresponding mobile node in the latest motion period, according to Lemma 1, the expectation of the motion vectors in the position list is equal to the actual motion vector of the mobile node. Therefore, the aggregated location information stored in the precision list is close
to the corresponding actual motion vector. As a consequence, the motion vectors in info messages are close to the actual motion vectors, and hence, the performance degradation of DRAM caused by the error of location information is limited.

4.8 Effect of Packet Loss Rate

Since the wireless communication is prone to packet errors, packet loss is common in MANETs. We investigate in this subsection the effect of packet loss rate on data accessibility. Packet loss rate is setting from 0% to 20% and the corresponding experimental result is shown in Fig. 14b. It is intuitive that data accessibilities of both schemes degrade as packet loss rate increases. In addition, as shown in Fig. 14b, the superiority of scheme DRAM over scheme E-DCG diminishes when the pack loss rate is high. Since scheme DRAM relies on the exchange of location information among neighboring mobile nodes, in the case of high packet loss rate, one node may miss some location information of its neighbors, and therefore recognize some connected nodes as unconnected. Such situation results in smaller allocation units which lead to lower degree of storage sharing and lower data accessibility. The experimental result shows that both schemes are vulnerable on packet loss. The degradation of accessibility caused by packet loss can be reduced by some techniques such as acknowledge and retransmission which enable reliable transmission at the cost of increasing produced network traffic.

4.9 The Effect of the Value of Time-to-Live

Fig. 15 shows the experimental results with the values of time-to-live (TTL) and simulation time varied. The values of TTL and simulation time are set from five meters to 20 meters and from 600 seconds to 3000 seconds, respectively. We use “DRAM- n” to indicate the scheme DRAM with TTL = n. It is obvious that the values of TTL only slightly affect data accessibilities. Although the smaller TTL value indicates that some mobility groups may be partitioned into several clusters, they are likely to be merged into the same allocation unit in the procedure of cluster merge. Therefore, the value of TTL does not significantly affect the accessibilities.

Fig. 15b shows the produced network traffics of all schemes with simulation time varied. Due to the employment of blind flooding, scheme E-DCG produces more traffic than all other schemes. We
also observe that DRAM-based schemes with smaller \(TTL\) values produce less traffic than that with larger \(TTL\) values in the 600-th second. It is because that when the relocation period is set to 600 seconds, most mobile nodes are in INITIAL states before the 600-th second, and hence, the produced traffic before the 600-th second is mainly caused by the broadcasting of \(info\) messages. Since the value of \(TTL\) indicates the ranges of the \(info\) message broadcasting in INITIAL states, and as a result, DRAM-based schemes with smaller \(TTL\) values produce less traffic than that with larger \(TTL\) values. After the 600-th second, most mobile nodes are in CLUSTER-MASTER and CLUSTER-MEMBER states. Therefore, most mobile nodes perform the procedure of cluster maintenance instead of \(info\) message broadcasting in each relocation period. DRAM-based schemes with smaller values of \(TTL\) indicate smaller clusters, and therefore, in the procedure of cluster maintenance, they produce more traffic (i.e., transmission of \(status\) messages) than DRAM-based schemes with larger values of \(TTL\). As a result, DRAM-based schemes with smaller \(TTL\) produce less traffic in the first relocation period, and produce more traffic in the successive relocation periods than DRAM-based schemes with larger \(TTL\). As shown in Fig. 15b, the traffic produced by DRAM-based schemes with smaller \(TTL\) is smaller than that produced by DRAM-based schemes with larger \(TTL\), and they are getting more and more close when the time approaches the 3000-th second.
5 Conclusion

We explored in this paper the problem of replica allocation in a MANET with group mobility. We first analyzed the employ group mobility model and derived several theoretical result. In light of these results, we proposed a replica allocation scheme DRAM to allocate replicas by considering group mobility. To evaluate the performance of the proposed schemes, several experiments were conducted. The experimental results showed that scheme DRAM cannot only obtain higher data accessibility but also produce less traffic than prior schemes.

Acknowledgements

The work was supported in part by the National Science Council of Taiwan, R.O.C., under Contracts NSC93-2752-E-002-006-PAE.

References