A Robust Clustering Algorithm for Mobile Ad Hoc Networks

Zhaowen Xing
University of Oklahoma
School of Computer Science
200 Felgar Street, Room 133 EL
Norman, OK 73019
U.S.A.
Phone: 1+ (405) 325-6803
Email: zhaowenxing@ou.edu

Le Gruenwald *
University of Oklahoma
School of Computer Science
200 Felgar Street, Room 144 EL
Norman, OK 73019
U.S.A.
Phone: 1+ (405) 325-3498
Email: ggruenwald@ou.edu

K.K. Phang
Faculty of Computer Science and Information Technology
University of Malaya, 50603 Kuala Lumpur
Malaysia
Phone: 60 + (603) 7967-6367
Email: kkphang@um.edu.my

Chapter submitted to

Handbook of Research on Next Generation Networks and Ubiquitous Computing

A book edited by Prof. Samuel Pierre

December 2008

A Robust Clustering Algorithm for Mobile Ad Hoc Networks

ABSTRACT

To mimic the operations in fixed infrastructures and to solve the routing scalability problem in large Mobile Ad Hoc Networks (MANET), forming clusters of nodes has been proven to be a promising approach. However, when existing weighted clustering algorithms calculate each node's weight, they either consider only one metric or rely on some metrics collected from extra devices. This often leads to a higher rate of re-clustering. This chapter presents a robust weighted clustering algorithm, called PMW (Power, Mobility and Workload), to form and maintain more stable clusters. In PMW, the weight of each node is calculated by its power, mobility and workload, which can be easily collected and computed locally and cover the major factors that cause re-clustering. Clustering overhead of PMW is analyzed. The simulation results confirm that PMW prolongs lifetime of MANETs and has a lower cluster head change rate and re-affiliation rate than other existing algorithms.

INTRODUCTION

A mobile ad hoc network (MANET) is a collection of battery-powered mobile nodes (or hosts) connected by relatively lower bandwidth wireless links. Each node has an area of influence called cell, only within which others can receive its transmissions. Due to no fixed infrastructures, all nodes can move freely, the network topology may change rapidly and unpredictably over time, and nodes have to form their own cooperative infrastructures. Thus, each node operates as an autonomous end system and a router for others in the network.

A MANET is of interest because there is no prior investment for fixed infrastructures, it can be easily deployed in a short time, and end users can access and manipulate data anytime and anywhere. Examples of MANET applications (Chlamtac et al., 2003) include law enforcement operations, automated battlefield applications, natural disaster recovery situations where the communication infrastructures have been destroyed, self-organizing sensor networks for data collecting, interactive lectures or conferences for data exchanging without pre-installed infrastructures. However, these MANET applications cannot be realized without efficient routing protocols. Current routing protocols over the flat network structure, in which each node participates as a peer, acts as a router and maintains individual routes to other nodes (Shiflet et al., 2004), are either proactive or reactive. Proactive routing protocols store route information in tables and update them periodically, while reactive routing ones search the routes on demand by flooding neighbors with the route requests. However, when the network size is large or the mobility of nodes is high, both the routing protocols over the flat network structure cannot work efficiently (Hong et al., 2002).

One promising way that can solve the routing scalability problem is to divide a MANET into clusters first, and then develop a routing protocol on top of the clustered MANET. A clustered MANET consists of cluster heads and cluster members, where a cluster head (like a mobile support station in a cellular mobile network) manages its clusters, coordinates intra/inter-cluster communication and so on. A cluster member is a node that belongs to a cluster and is not a cluster head.

Many weighted clustering algorithms have been proposed to elect cluster heads, form clusters and maintain clusters (Basagni, 1999; Basu et al., 2001; Chatterjee et al., 2002; Kim, 2006; Liu et al., 2005; Sheu and Wang, 2006). However, when calculating the weight utilized to determine whether a node is eligible to be a cluster head, these algorithms either consider only one metric (like mobility or power of nodes) (Basu et al., 2001; Kim, 2006; Sheu and Wang, 2006) or rely on some metrics collected from extra devices (such as locations of nodes read from Global Positioning Systems) (Chatterjee et al., 2002). This often leads to a higher possibility of re-clustering and, consequently, quality of service cannot be provided.

This chapter presents a robust weighted clustering algorithm, called PMW (Power, Mobility, Power), to form and maintain more stable clusters in MANETs. In PMW, in order to take into account the major factors that frequently cause re-clustering and in order to avoid being dependent on any extra device, the weight of a node is calculated by three parameters: power, mobility and workload. The results of our simulation show that PMW balances the power usage, prolongs the lifetime of clustered MANETs and has a lower cluster head change rate and re-affiliation rate than MOBIC (Basu et al., 2001). The rest of this chapter is organized as follows. Section 2 reviews the related background. Section 3 presents our approach to build stable clusters based on power, mobility and workload. Section 4 analyzes the message overhead and time complexity of PMW. Section 5 evaluates PMW and MOBIC by using the NS-2 simulator. Section 6 discusses some direction of future research. Finally, Section 7 concludes the chapter.

BACKGROUND

Several weighted clustering algorithms have been proposed and surveyed in (Yu and Chong, 2005). Here we briefly review some of the newly published approaches and others, which are already reviewed in (Yu and Chong, 2005), but are strongly related to our algorithm PMW. By considering the system parameters that are utilized to calculate the weight of each node, these approaches are categorized as mobility-only-based (Basu et al., 2001; Kim, 2006), power-only-based (Sheu and Wang, 2006) and combination-based (Basagni, 1999; Chatterjee et al., 2002; Liu et al., 2005).

Mobility-only-based

Mobility of nodes triggers re-clustering and makes networks unstable, thus, it becomes the key attribute in the weight computation in the mobility-only-based clustering algorithms.

In MOBIC (Basu et al., 2001), in order to form stable clusters, the Relative Mobility (*RM*) metric is introduced and calculated as the logarithm of ratio of received signal strengths (RSS): $10 * \log_{10} \frac{rss_1}{rss_2}$, where rss_1 and rss_2 are read from the RSS indicator when two successive HELLO messages, which are sent by the same neighbor, are received. For each node, the variance of RMs among its neighbors with respect to 0 (not the exact mean) is calculated as the aggregate local mobility metric. The nodes with the lowest aggregate local mobility among their neighbors are elected as cluster heads. Unfortunately, it is possible that some elected cluster heads may almost run out of power, thus, the re-election has to be invoked soon.

In ACT (Average Connection Time) (Kim, 2006), to overcome the negative effects caused by nodes moving fast or moving back and forth, the ACT of each node with its neighbors during a time period is introduced as the major parameter to form and maintain the clusters, and nodes having the largest ACT value become cluster heads. However, the ACT is similar to the cumulative time in WCA (Chatterjee et al., 2002) or elapsed time (Liu et al., 2005), it can not accurately reflect the current level of the battery power because a node may have connected with its neighbors too long and it may almost run out of power.

Power-only-based

A node with a higher remaining power level is, of course, a better candidate for the cluster head; so battery power is the only system parameter applied to calculate the weight of each node in power-only-based clustering algorithms.

Because nodes with higher battery power have a higher priority to become cluster heads, it is possible that nodes with the least battery power being left out and claiming themselves as cluster heads. In Sheu's Stable Cluster Algorithm (SCA) (Sheu and Wang, 2006), Sheu et al. set up a battery power level threshold, define nodes whose battery level is below the threshold as bottlenecks, count the number of neighbors that are bottlenecks for each node, and elect nodes with the largest number of bottlenecks as cluster heads. By taking the detour in the election, nodes with the least battery power are kept from becoming cluster heads, thus, the clusters become more stable. Unfortunately, because the mobility of nodes is not considered in the election, the possibility of re-clustering is still high when elected cluster heads have high mobility.

Combination-based

Each node is assigned with a weight, which is calculated by considering more than one system parameters like node degree, remaining power, roaming speed, and so on (Yu and Chong, 2005). In DCA (Distributed Clustering Algorithm) (Basagni, 1999), each node is assumed to have a different weight, nodes with the biggest weighs are elected as cluster heads, and 1-hop neighbors of elected cluster heads join the cluster as ordinary nodes. However, the calculation of nodes' weights is not discussed (Basu et al., 2001).

In WCA (Weighted Clustering Algorithm) (Chatterjee et al., 2002), to determine whether a node *v* is suited for being a cluster head, the weight of each node (W_v) is calculated by a formula as shown below that consists of four system parameters: sum of distance with all neighbors (D_v), average running speed (M_v), cumulative time of serving as a cluster head (P_v) and degree difference of nodes (Δ_v), where $\Delta_v = |d_v - \delta|$, in which d_v is the number of neighbors and δ is the ideal number of neighbors that a cluster head can handle. Unfortunately, how to choose δ is not discussed (Yu and Chong, 2005).

$$\begin{aligned}&W_v = f_1 * \Delta_v + f_2 * D_v + f_3 * M_v + f_4 * P_v,\\ &\textit{where}, f_1, f_2, f_3 \textit{ and } f_4 \textit{ are weighting factors}\\ &\textit{and} \sum_{i=1}^{4} f_i = 1\end{aligned}$$

The values of f_1, f_2, f_3 and f_4 are varied based on different applications. The nodes with the lowest weights are elected as cluster heads. However, how to normalize these parameters is not addressed explicitly. The global positioning system (GPS), the accuracy of which is not ideal for fine computing and the operations of which would drain the limited power of the node quickly, has to be applied to obtain the coordinates of each node for computing the running speed. The cumulative time of a node already serving as a cluster head cannot accurately reflect the current level of battery power because a busy node may almost run out of power and it has never been a cluster head.

In Liu's Group Management (Liu et al., 2005), the resource (*R*) richness and elapsed time of a node being a cluster head (leader) are integrated to evaluate a node's suitability for being a cluster head. The resources are CPU load (*L*), memory (*M*), battery (*B*) and bandwidth (*BW*). The elapsed time (*ET*) is the time between now and the last time a node is a cluster head. The weight of a node *v* is calculated by the following formulas, and nodes with the highest weights are elected as cluster heads.

$$W_v = f_1 {*} R_v + f_2 {*} ET_v, \text{ where } f_1 + f_2 = 1, \text{ and}$$
$$f_1 {*} R_v = f_{11} {*} L_v + f_{12} {*} M_v + f_{13} {*} B_v + f_{14} {*} BW_v,$$
$$\text{in which, } f_1 = f_{11} + f_{12} + f_{13} + f_{14}$$

However, how to normalize these parameters is not addressed explicitly either. The use of elapsed time in cluster head determination has the same disadvantage as the use of cumulative time in WCA (Chatterjee et al., 2002). In addition, re-clustering may be frequently triggered since the mobility of nodes is not considered in the election.

THE PROPOSED PMW APPROACH

Being inspired by MOBIC (Basu et al., 2001) and WCA (Chatterjeet al., 2002) and considering a new system parameter, called "Power Decreasing Rate (*PDR*)", we propose a weighted clustering algorithm, called PMW, to build a stable backbone in MANETs. Although our proposed clustering algorithm is also combination-based, it can become mobility-only-based if we tune the weighting factors accordingly. In other words, our approach can build a more stable backbone for MANETs by forming clusters.

The Basis of Our Algorithm

To capture the mobility of nodes, we do not consider the absolute roaming speed, which is actually applied in WCA (Chatterjee et al., 2002). This is because it is easy to calculate the speed's quantity but it is hard to predict the direction of movement. Without the direction, the speed's quantity alone is not appropriate to justify whether a node is a good candidate or not. For instance, the speeds of two nodes are small, but they both move in the opposite directions. As time goes, they will be out of each other's transmission range and get disconnected from each other. Also the utilization of GPS is opted out due to the following reasons:

- When a GPS is utilized, every node must be equipped with one, which incurs high hardware costs. In addition, these GPSs consume the limited battery power of nodes.
- GPS does not work indoor because buildings shield the satellite signals (Basu et al., 2001; Bruning et al., 2007).
- The accuracy of a typical civilian GPS is in the range of 6-12 meters (Zidek et al., 2006), thus, the returned results could be the same when two GPSs are located within 10 meters.

- If data cannot be read from the GPS and no replacement can be found, then the whole system has to wait or fail.

Instead, two mobility metrics, Relative Mobility (*RM*) (Basu et al., 2001) and Mobility Prediction (*MP*), are introduced to monitor the mobility of nodes and applied to determine whether a node is suitable to be a cluster head as follows:

- For each node j ($1 \le j \le N$ for N nodes in the network), after receiving two successive HELLO messages from every 1- hop neighbor i ($1 \le i \le n$ if there are n neighbors), the RM_{ij} is calculated by the formula (3.1). rss_{ij1} and rss_{ij2} are the received signal strength (RSS) that are read from the RSS indicator when the first and second HELLO message from the same neighbor are received, respectively. Based on the value of RM_{ij}, we can say that if RM_{ij} is equal to 1, then the node j and its neighbor i either do not move at all or move with the same speed in the same direction; if RM_{ij} is less than 1, then they move close to each other; otherwise, they move away from each other.

$$RM_{ij} = \frac{rss_{ij1}}{rss_{ij2}} \qquad (3.1)$$

- For each node j, to take into account the mobility of all n 1-hop neighbors, MP_j is calculated as the standard deviation of $RM_{1j}, RM_{2j}, \ldots, RM_{nj}$ shown in the formula (3.2). However, for the stability of elected clusters, we prefer RM_{ij} to be equal to or less than 1 because we want cluster heads not to move away from their members. Thus, in the MP_j calculation the mean of RM_{ij} ($1 \le i \le n$) is 1 instead of the actual mean. A node j with a lower MP_j means that it stays closer to its neighbors, thus, it is a better candidate for the cluster head among its neighbors.

$$MP_j = \sqrt{\frac{\sum_{i=1}^{n} (RM_{ij} - \overline{RM_{ij}})^2}{n}}, \qquad (3.2)$$
$$where\ \overline{RM_{ij}} = 1$$

When dealing with the limited battery power, we consider not only the Remaining Power (RP) of each node but also its Power Decrease Rate (*PDR*) as the workload because nodes with heavier workload consume more energy, so that we can balance the power usage and prevent cluster heads from running out of power quickly. In other words, for each node j, the PDR_j is considered because the RP_j represents only the current state of power level and the power will run out soon if this node usually has a heavy workload (for instance, it provides service as a server and relays packets for many neighbors). The PDR_j at time interval [t_1, t_2] is calculated by using the formula (3.3), where rp_{j1} and rp_{j2} are the remaining power at time t_1 and t_2, respectively.

$$PDR_j = \frac{rp_{j1} - rp_{j2}}{t_2 - t_1} \qquad (3.3)$$

A node with a lower *PDR* indicates that it was not busy at least during the interval [t_1, t_2]. However, when a node had a busy work history, it most likely would be busy in the future as well. Since the larger the time interval is, the more accurate the *PDR* is in indicating a node's workload history, during the initial election, each node saves a copy of its initial remaining power and initial time as rp_{j1} and t_1, such that a more accurate *PDR* can be calculated in the future re-election.

Based on the above analysis about power, mobility and workload, it is obvious that a node *j* is the best candidate for a cluster head among all its neighbors if its RP_j is the highest, its MP_j is the lowest and its PDR_j is the lowest. In other words, a node with the highest weight is the best candidate for a cluster head when we combine these three metrics together as the weight, which is calculated in formula (3.4). Since these metrics have different units, we apply the inversed exponential function to normalize MP_j and PDR_j and bound their values between 0 and 1. RP_j is left out because it is the remaining power level in percentage and its value is already between 0 and 1.

$$W_j = f_1 * e^{-MP_j} + f_2 * RP_j + f_3 * e^{-PDR_j} \quad (3.4)$$

In formula (3.4), $RP_j = rp_{j2}$, the weighting factors f_1, f_2 and f_3 are set according to the different scenarios in the applications, and $f_1 + f_2 + f_3 = 1$. When we let $f_2 = f_3 = 0$, that is, we take away the effect of power and workload, our algorithm turns into a mobility-only-based approach just like MOBIC (Basu et al., 2001).

Cluster Formation

Cluster formation involves the following four steps, and messages used in cluster formation and maintenance are summarized in Table 1.

Table 1. Messages Used in PMW

Message	Description
HELLO(my_ID, my_W, CH_ID, my_RP, other_CH)	To notify neighbors about my ID, my weight, my cluster head's ID, my remaining power and any other neighboring cluster heads.
WEIGHT(my_ID, my_W)	To notify neighbors about the value of my weight.
CLUSTERHEAD(my_ID, CH_ID)	To notify neighbors about my role: I am a cluster head, that is, my ID is same as my cluster head's ID.
JOIN(my_ID, CH_ID)	To notify neighbors that I am going to join the cluster whose cluster head's ID is CH_ID. If a cluster head broadcasts a JOIN message, then it informs its members about its resignation and joins a cluster at the same time.

Step 1: Each node *j* periodically broadcasts (this broadcast interval is predefined (Basu et al., 2001) a HELLO message with the same transmission power. In the mean time, the remaining power level rp_1 is recorded at the initial election time t_1; for each HELLO message received from neighbor *i*, rss_{ij} is recorded. If only one HELLO message from a neighbor is received after *j* broadcasts three successive HELLO messages, then this neighbor is excluded from the weight calculation (Basu et al., 2001).

Step 2: Immediately after each node *j* receives two successive HELLO messages from all the 1-hop neighbors, it records the remaining power level rp_{j2} and the time t_2. And then node *j* calculates the values of RM_j, MP_j, PDR_j and W_j using the formulas 3.1, 3.2, 3.3 and 3.4 defined above, respectively. When the weight is re-calculated, the saved remaining power rp_{j1} at time t_1 is applied in the calculation of PDR_j.

Step 3: Each node j broadcasts the value of W_j to all its neighbors in a WEIGHT message, and waits for their WEIGHT messages.

Step 4: Upon receiving the weights from all 1-hop neighbors, the nodes with the highest weight declare themselves as cluster heads among their 1-hop neighbors by broadcasting a CLUSTERHEAD message, but, no two cluster heads should be 1-hop neighbors. All 1-hop neighbors of elected cluster heads just join them as their members by broadcasting JOIN messages. If two nodes have the same weight, then the node with a smaller ID becomes the cluster head (Basu et al., 2001). If it happens that a node is 1-hop neighbor of two or more cluster heads, then it joins the cluster head with the highest weight and works as a gateway for these cluster heads.

Cluster Maintenance

Because every node can roam and has limited battery power in a MANET, the links between members and cluster heads can be broken, and the links between two cluster heads can be generated (Xue et al., 2006). Consequently, clusters need be re-clustered. In other words, leaving clusters, joining clusters, merging clusters, and re-electing cluster heads are normal re-clustering operations in a clustered MANET. However, these operations should be performed only on demand to reduce the overhead of computation and communication, and to provide consistent quality of service.

In order to detect the link breaks and new link establishments, each node periodically broadcasts a HELLO message, which contains the ID and weight of itself, its cluster head's ID, its remaining power and any other neighbor which is a cluster head (a Boolean variable). Being a cluster head, it has to periodically monitor its remaining power level so that it will resign when the remaining power drops below a predefined threshold. Also each node keeps recording the values of RSS from the last two HELLO messages and re-calculating its weight in case of future re-elections.

Relying on these two periodical operations, cluster maintenance can be done by the following recovery:

- from the link break between a member and its cluster head: after three successive broadcast intervals (*BI*) (Wang and Kim, 2007), if no HELLO message is received from a member, the cluster head will just remove this member from its neighbor and member lists. On the other hand, if a member does not receive a HELLO message from the cluster head after three successive *BI*s, it removes the cluster head from its neighbor list, and joins another cluster head with the highest weight if any. If no other cluster head is available from its neighbor list, this member declares itself as a cluster head.
- from the link establishment because two cluster heads become 1-hop neighbors: if a cluster head has become a 1-hop neighbor of another cluster head for a predefined Cluster Contention Interval (*CCI*) (Basu et al., 2001), then the one with the smaller weight resigns and joins the other. The members of the resigned cluster head cannot join the new cluster head because of non-1-hop neighbors. They have to join other cluster heads with the highest weight or declare themselves as cluster heads instead.

- from the link break because a cluster head resigns: if the current remaining power of a cluster head is less than the Low Power Threshold (*LPT*), and if there exists a member whose power is higher than *LPT,* and this member has no other neighbor that is a cluster head, then the cluster head resigns and triggers a cluster head re-election. But, this re-election is limited to inside the old cluster. That is, the resigned cluster head goes through each member's profile, which is periodically updated after receiving a HELLO message, and finds a replacement that has the highest weight. After the new cluster head is elected, the resigned cluster head joins its cluster. If a member cannot join the new cluster head because they are not 1-hop neighbors, it has to join others or declare itself as a cluster head.

ANALYSIS OF PMW

In this section, we analyze PMW with respect to the message overhead per time step per node and time complexity per network topology change. These terms are defined below. The approach used is inspired by the theoretical analysis in (ER and Seah, 2005).

The price of clustering is that extra time is consumed and additional messages are incurred to form and maintain clusters. The consequence of these additional messages is called message overhead (the more messages are transmitted, the more traffic is in the network and the more battery power of nodes is consumed). Since bandwidth and battery power of each node are limited in MANETs, message overhead is an important metric for evaluating the performance of a clustering algorithm. We analyze the message overhead by analyzing the overhead due to the HELLO protocol and the overheads due to cluster formation and maintenance. In the mean time, the time complexity per network topology change is also computed.

To simplify the analysis, the continuous runtime is divided into discrete time steps, which are the duration between the time when a message is sent and the time when the message is received and processed by a receiver (Bettstetter and Konig, 2002). The random waypoint mobility model with zero pause time is assumed. The following definitions are used in the analysis (ER and Seah, 2005):

- N: the number of nodes in the MANET;
- m: the average number of cluster members in a cluster; $m = \Theta(1)$ because all clusters should have a maximum size constrain to avoid overburdening cluster heads (Banerjee and Khuller, 2001).
- f_{hello}: the number of HELLO messages broadcast by a node per time step; $f_{hello} = \Theta(1)$ because f_{hello} is proportional to average node speed s and inversely proportional to the transmission radius R, and both s and R are less than or equal to some constants (Sucec and Marsic, 2004).
- f_{link}: the average frequency of network topology changes occurred per time step; $f_{link} = \Theta(N)$ (Sucec and Marsic, 2004).
- T: the number of time steps taken by the algorithm after a network topology change to re-establish a valid cluster structure (or called re-clustering);
- M: the number of messages (or called packets) exchanged between nodes after a network topology change to re-establish a valid cluster structure;
- L: the total evaluation time in terms of time steps.

In terms of the average number of messages transmitted by PMW per time step per node, and the number of time steps needed to re-establish a valid cluster structure after a topology change, the following claims are made.

Claim 1: the *message overhead of PMW is O(1) packet transmissions per time step per node.*

Claim 2: *the time complexity of PMW is* $T \leq 2$ *per network topology change.*

Both claims are proved in the following subsections.

Hello Protocol Overhead

In order to discover its neighborhood and compute its weight, each node broadcasts HELLO messages periodically. Thus, the HELLO protocol introduces an overhead of $f_{hello}*N$ packets per time step for all nodes.

Cluster Formation Overhead

Immediately after each node calculates its weight, it broadcasts a WEIGHT message in one time step. After receiving WEIGHT messages from all its 1-hop neighbors, each node either becomes a cluster head by broadcasting a CLUSTERHEAD message or joins some cluster by broadcasting a JOIN message in one time step. Thus, for all nodes, they broadcast $2N$ messages in 2 time steps, that is, cluster formation overhead is N messages per time step.

Cluster Maintenance Overhead

From the discussion of cluster maintenance, it is obvious that every network topology change is detected by relying on the periodical HELLO messages, and each cluster head resignation is verified by periodically checking the remaining power level. Once a network topology change or a cluster head resignation occurs, relating nodes have to take respective actions to re-establish a valid cluster structure. Because of these actions, cluster formation and maintenance overheads incur, which are investigated in the following four subsections.

Link Break between a Member and Its Cluster Head

Since the cluster structure is still valid when a link break occurs between nodes from different clusters or nodes that are members from the same cluster, there is no action. Only a link break between a member and its cluster head triggers the re-clustering.

The cluster head removes this member from its neighbor and member lists, so no message is necessarily transmitted. On the other hand, this member removes this cluster head from its neighbor list as well, and joins another cluster head with the highest weight if any. This case is done by broadcasting a JOIN message in one time step. If no other cluster head is available from its neighbor list, this member declares itself as a cluster head, and broadcasts a CLUSTERHEAD message in one time step. Thus, we have: $T = 1$ and $M = 1$ for one of this kind of link breaks.

Link Establishment Because Two Cluster Heads Become 1-hop Neighbors

When a cluster head becomes a 1-hop neighbor of another cluster head for a predefined duration of *CCI* interval, the one with the smaller weight resigns and joins the other. The resigned cluster head has to broadcast a JOIN message to inform all its members in one time step. After receiving the JOIN message from their cluster head, each member of this resigned cluster head has to do re-clustering, which is the same as the case in subsection "Link Break between a Member and Its Cluster Head". To summarize, $T = 2$ and $M = m + 1$ for one of this kind of link establishments.

Link Break Because a Cluster Head Resigns

When the current remaining power of a cluster head is less than the threshold *LPT*, and there exists one of its members that can be a new cluster head, this cluster head resigns and triggers a cluster head re-election.

Once a new cluster head is elected, the resigned cluster head broadcasts a JOIN message to inform all its members in one time step. After receiving the JOIN message from its cluster head, each member of this resigned cluster head has to do re-clustering, which is also the same as the case in subsection "Link Break between a Member and Its Cluster Head". In short, $T = 2$ and $M = m + 1$ for one of this kind of link breaks.

Total Cluster Maintenance Overhead

Since $M = 1$ in the case of "link break between a member and its cluster head" and $M = (m+1)$ in the case of "link establishment because two cluster heads become 1-hop neighbors", the total number of messages transmitted per network topology change due to link state changes is $(m+2)$. And the average network topology changes occurred per time step is f_{link}. Therefore, there are totally $f_{link}*(m+2)$ messages per time step due to link state changes.

Since a node cannot become a cluster head any more once it resigns due to its lower remaining power, there are at most N cluster head resignations in an evaluation. An evaluation period consists of L time steps, thus, the average number of cluster head resignations per time step should be N/L. Therefore, the total number of messages is $N(m+1)/L$ per time step due to the cluster head resignation.

In summary, total cluster maintenance overhead is $f_{link}*(m+2) + N(m+1)/L$ messages per time step.

Total Message Overhead

To summarize, the message overhead of PMW (O_{PMW}) is the sum of the overhead due to HELLO protocol, the overhead due to cluster formation and the overhead due to cluster maintenance, that is,

$$O_{PMW} = f_{hello}*N + N + f_{link}*(m+2) \; + \; N(m+1)/L$$

Since $f_{hello} = \Theta(1)$, $f_{link} = \Theta(N)$, $m = \Theta(1)$, L is an integer and $L > 1$, given some constants c_1, c_2 and c_3, we have: $f_{hello} \leq c_1$, $f_{link} \leq c_2 * N$, $m \leq c_3$ and $(1/L) < 1$. Therefore, O_{PMW} can be expressed as follows:

$$\begin{aligned}
&O_{PMW} = f_{hello}*N + N + f_{link}*(m+2) + N(m+1)/L \\
&\Rightarrow O_{PMW} \leq f_{hello}*N + N + f_{link}*(m+2) + N(m+1) \\
&\Rightarrow O_{PMW} \leq c_1 * N + N + c_2 * N * (c_3 + 2) + N * (c_3 + 1) \\
&\Rightarrow O_{PMW} \leq (2 + c_1 + 2c_2 + c_3 + c_2 c_3) * N \\
&\Rightarrow O_{PMW} = O(N)
\end{aligned}$$

After dividing O(N) by the number of nodes N, the message overhead of PMW is O(1) per time step per node and Claim 1 is proved.

$T = 1$ for a link break between a member and its cluster head, and $T = 2$ for both a link establishment and a link break due to the resignation of some cluster head, therefore, the convergence time is at most 2 time steps per topology change, as per Claim 2.

PERFORMANCE EVALUATION

The performance of PMW and MOBIC (Basu et al., 2001) is evaluated via NS-2 simulator with clustering framework (Basagni et al., 2006). Most of our simulation parameters are the same as the ones in (Basu et al., 2001) listed in Table 2 along with others. Since mobility is the major cause of re-clustering, the weighting factors $f_1 = 0.8$, $f_2 = 0.15$ and $f_3 = 0.05$ are used. The initial power level of each node is randomly distributed between 20% and 100%.

Table 2. Simulation Parameters

Parameter	**Value**
Number of nodes (*N*)	50
Network size	670m x 670m
Maximum speed of node movement	1, 10, 20, 30m/s
Transmission range (*TR*)	10m – 250m
Pause time (*PT*)	0s, 30s
Broadcast interval (*BI*)	1s
Cluster contention interval (*CCI*)	3s
Low power threshold (*LPT*)	30%
Simulation time	200s

To measure the stability of a clustered MANET, we consider the following metrics:

- The lifetime of the network: the duration from the beginning until any node runs out of its battery power (Choi and Woo, 2006; Sheu and Wang, 2006).
- The cluster head change rate (per second): the total number of cluster heads is divided by the total simulation time (Basu et al., 2001).

- The re-affiliation (joining a cluster and becoming a member) rate (per second): the total number of cluster members is divided by the total simulation time (Choi and Woo, 2006).

All metrics in the following figures are collected from an average value of 50 simulation runs in 50 different scenarios, which are randomly generated using the random waypoint model (built-in in NS-2). To better mimic a real wireless network, 25 constant bit rate (CBR) connections are randomly generated by the traffic-scenario generator. Each source sends a 512-byte packet through UDP at a rate of one packet per second.

In Figures 1, the lifetime of the network decreases as node mobility and transmission range increase in both MOBIC and PMW. In Figure 1(a), PMW prolongs the lifetime of the network from 9% to 42% (or 23% on average) better compared to that of MOBIC. In addition, PMW outperforms MOBIC from 2% to 27% (or 15% on average) when the transmission range is beyond 50 meters as shown in Figure 1(b). These promising results show affirmatively the effect of taking into consideration the power and workload in the weight calculation and the forced resignation of a cluster head when its power becomes too low.

Figure 1. Lifetime of the network by varying maximum speed and transmission range, respectively

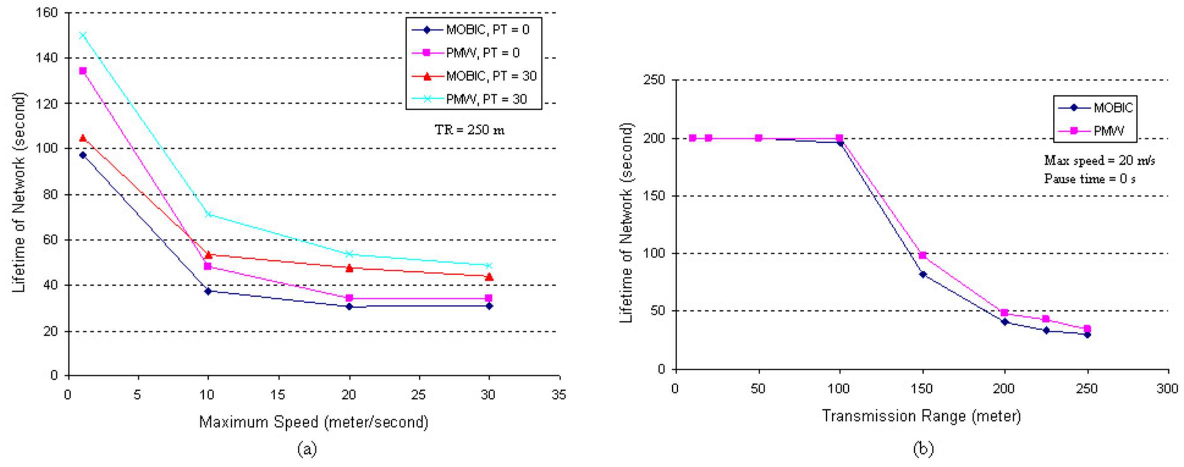
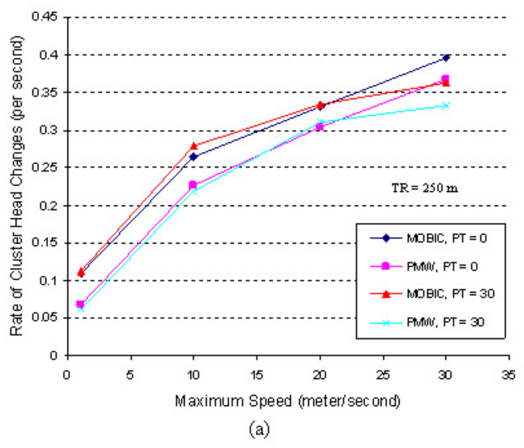


Figure 2(a) shows that the cluster head change rate increases as node speed increases. This is because cluster heads with higher speed are more likely to become 1-hop neighbors, and consequently, the one with a lower weight has to resign. PMW produces from 5 to 12 (or 7 on average) fewer cluster heads than MOBIC for *PT* (pause time) = 0 and *PT* = 30, respectively. In Figure 2(b), the number of cluster heads increases when the transmission range is less than 50 meters. However, when the transmission range becomes larger than 50 meters, the cluster head change rate decreases as more nodes stay together for a longer period of time, and PMW produces from 5 to 13 (or 10 on average) fewer cluster heads than MOBIC. PMW produces fewer cluster heads mainly because nodes with higher power are likely to get elected as cluster heads and, hence, can function as cluster heads for a longer time.

Figure 2. Rate of cluster head changes by varying maximum speed and transmission range, respectively



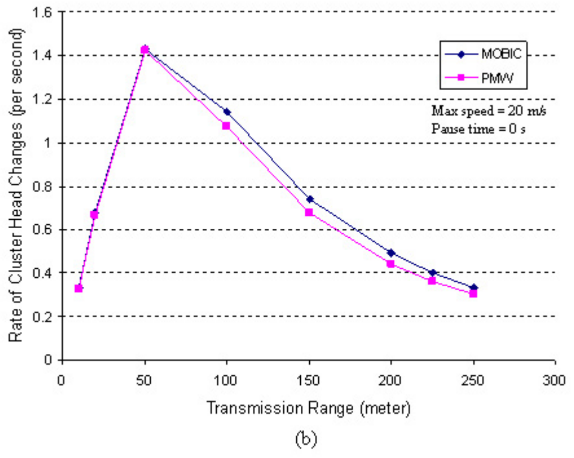
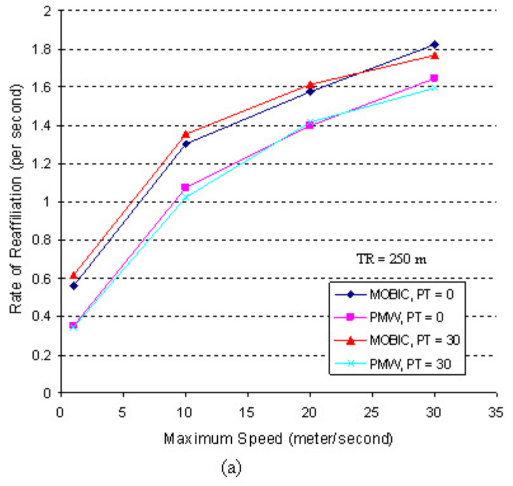
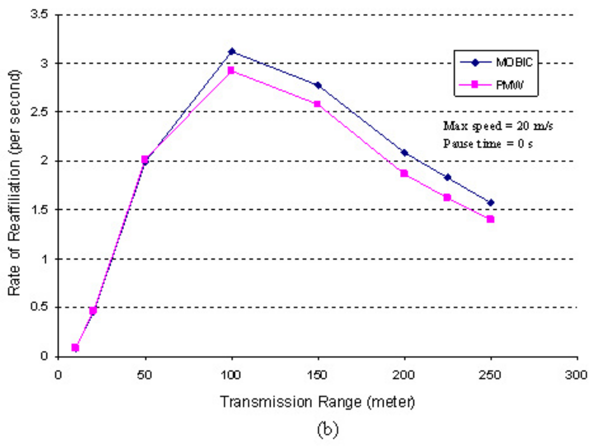


Figure 3(a) shows that the re-affiliation rate increases as the node speed increases. This is because cluster members with higher speeds are likely to get disconnected from their cluster heads and join the others. PMW produces from 34 to 66 (or 44 on average) fewer cluster members than MOBIC for PT (pause time) = 0 and PT = 30, respectively. In Figure 3(b), the re-affiliation rate has the similar trend as in Figure 2(b), and PMW produces from 34 to 43 (or 40 on average) fewer re-affiliations than MOBIC when the transmission range is greater than or equal to 100 meters. The advantage of PMW having a lower re-affiliation rate is mainly attributed to the less likelihood of the resignation of a cluster head due to power exhaustion.

Figure 3. Rate of re-affiliation by varying maximum speed and transmission range, respectively





FUTURE RESEARCH

Since performance of routing protocols over the flat structure suffers as number of nodes grows (Hong et al., 2002; Shiflet et al., 2004), there is the need to develop a reliable routing protocol, which is built on top of a hierarchical structure produced by a light-weight clustering algorithm.

Although replication provides a feasible solution for improving data availability in MANET databases, efficient consistency maintenance of replica remains a challenging problem in a large MANET. Cluster heads are the most stable nodes in the network and can form a virtual backbone, so they can decrease message overhead and help to realize efficient replication service.

Since pessimistic concurrency control techniques limit/block users to access data items and resources (bandwidth, power and storage) in MANET databases are limited, optimistic concurrency control algorithms may efficiently guarantee correctness of concurrent transactions execution in MANET databases, but it is challenging to choose which nodes to maintain the history information that is used for future validation. Again, cluster heads are the best candidates because they are more stable than other nodes, so that the history information is available when it is needed.

CONCLUSION

In this chapter, in order to resolve the mobility and scalability issues in routing protocols in MANETs, we proposed a robust weighted clustering algorithm, called PMW (Power, Mobility, Workload), to form and maintain more stable clusters in MANETs. In PMW, the weight of a node is calculated by three parameters: remaining power, mobility prediction (to check if a node moves along with all its 1-hop neighbors) and workload (represented by power decrease rate because nodes with heavier workload consume more energy). These three metrics are computed locally, independent of extra devices, and cover the major causes of re-clustering. Thus, in PMW, there is low re-clustering overhead during the cluster maintenance. The message overhead incurred due to PMW was claimed as O(1) per time step per node. The simulation results from the NS-2 simulator did confirm that PMW prolonged the lifetime of a clustered MANET and outperformed MOBIC by as much as 23% on average; PMW elected at least 7 fewer cluster heads on average than MOBIC; and PMW produced at least 40 fewer cluster members on average than MOBIC.

Acknowledgement
This material is based upon work supported by (while serving at) the National Science Foundation (NSF) and the NSF Grant No. IIS-0312746. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

REFERENCES:

Basagni, S. (1999). Distributed Clustering for Ad Hoc Networks. *Proceedings of the International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN '99)*, (pp. 310-315). Perth/Fremantle, WA, Australia.

Basagni, S., Mastrogiovanni, M., Panconesi, A., & Petrioli, C. (2006). Localized Protocols for Ad Hoc Clustering and Backbone Formation: A Performance Comparison. *IEEE Transactions on Parallel and Distributed Systems*, *17*(4), 292-306.

Banerjee, S., & Khuller, S. (2001). A Clustering Scheme for Hierarchical Control in Mult-hop Wireless Networks. *In Proceeding of 20th IEEE INFOCOM,* (pp.1028-1037). Anchorage, Alaska.

Basu, P., Khan, N., & Little, T. D. C. (2001). A Mobility Based Metric for Clustering in Mobile Ad Hoc Networks. *In Proceeding of IEEE ICDC,* (pp. 413-418). Phoenix, Arizona, USA.

Bettstetter, C., & Konig, S. (2002). On the Message and Time Complexity of a Distributed Mobility-Adaptive Clustering Algorithm in Wireless Ad Hoc Networks. *In proceeding of the 4th European Wireless*, (pp.128-134). Florence, Italy.

Bruning, S., Zapotoczky, J., Ibach, P., & Stantchev, V. (2007). Cooperative Positioning with MagicMap. *Workshop on Positioning, Navigation and Communication 2007 (WPNC'07)*, Hannover, (pp.17-22). Leibniz University of Hannover, Germany.

Chatterjee, M., Das, S. K., & Turgut, D. (2002). WCA: A Weighted Clustering Algorithm for Mobile Ad Hoc Networks. *Cluster Computing*, *5*(2), 193-204.

Chlamtac, I., Conti, M., & Liu, J. J. N. (2003). Mobile Ad Hoc Networking: Imperatives and challenges. *Ad Hoc Networks Publication*, *1*(1), 13-64.

Choi, W., & Woo, M. (2006). A Distributed Weighted Clustering Algorithm for Mobile Ad Hoc Networks. *Proceedings of the Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services,* (pp. 73-78). Guadeloupe, French Caribbean.

ER, I., & Seah, W. (2005). Clustering Overhead and Convergence Time Analysis of the Mobility-based Multi-hop Clustering Algorithm for Mobile Ad Hoc Netorks. *Proceedings of the 11th International Conference on Parallel and Distributed System,* (pp. 1144 – 1155). Fuduoka, Japan.

Hong, W., Xu, K., & Gerla, M. (2002). Scalable Routing Protocols for Mobile Ad Hoc Networks. *IEEE Networks, 16*(4), 11-21.

Kim, K. (2006). A Novel Factor for Robust Clustering in Mobile Ad Hoc Networks. *IEICE Transactions on Communications*, *E89-B*(4) 1436-1439.

Liu, J., Sailhan, F., Sacchetti, D., & Issarny, V. (2005). Group Management for Mobile Ad Hoc Networks: Design, Implementation and Experiment. *Proceedings of the 6th international conference on Mobile Data Management*, (pp.192-199). Ayia Napa, Cyprus.

Sheu, P., & Wang, C. (2006). A Stable Clustering Algorithm Based on Battery Power for Mobile Ad Hoc Networks. *Tamkang Journal of Science and Engineering*, *9*(3), 233-242.

Shiflet, C. F., Belding-Royer, E. M., & Perkins, C. E. (2004). Address Aggregation in Mobile Ad-hoc Networks. *Proceedings of the IEEE International Conference on Communications*, (pp. 3734-3738). Paris, France.

Sucec, J., & Marsic, I. (2004). Hierarchical Routing Overhead in Mobile Ad Hoc Netorks. *IEEE Transactions on Mobile Computing*, 3(1), 46-56.

Wang, Y., & Kim, M. S. (2007). Bandwidth-adaptive Clustering for Mobile Ad Hoc Networks. *International Conference on Computer Communications and Networks,* (pp. 103-108). Honolulu, Hawaii, USA.

Xue, M., ER, I., & Seah, W. K. G. (2006). Analysis of Clustering and Routing Overhead for Clustered Mobile Ad Hoc Networks. *Proceedings of the 26th IEEE international Conference on Distributed Computing Systems*, (pp. 46-53). Lisboa, Portugal.

Yu, J. Y., & Chong, P. H. J. (2005). A Survey of Clustering Schemes for Mobile Ad Hoc Networks. *IEEE Communications Survey &Tutorials*, *7*(1), 32-48.

Zidek, K., Saloky, T., & Polanecka, I. (2006). Usability of GPS Systems for Mobile Robots Navigation. *4th Sloyakian-Hungarian Joint Symposium on Applied Machine Intelligence*. Herlany, Slovakia.

KEY TERMS AND THEIR DEFINITIONS:

Clustering in a MANET: Nodes are divided into non-overlapping clusters according to certain rules, and are assigned different roles: cluster heads or cluster members. A cluster head manages its clusters, coordinates intra/inter-cluster communication and so on, while a cluster member is a node that belongs to a cluster and is not a cluster head.

Mobile Ad Hoc Network (MANET): A collection of battery-powered nodes connected by relatively lower bandwidth wireless links. Nodes move randomly and organize themselves arbitrarily.

Mobility: Every node has ability to roam in the network.

.
Mobility Prediction: A quantity is applied to predict whether a node moves along with all its 1-hop neighbors

Multi-hop Communication: Communication between two nodes is carried out through a number of intermediate nodes that relay packets from one point to another.

Network Lifetime: The duration elapsed from the beginning until the time when a node runs out of its battery power.

Power Decrease Rate: The rate of a node's remaining power decreasing within a time interval.

Relative Mobility: A quantity is used to determine whether a node moves close to or away from another node.

Weight-based Clustering Algorithm: A clustering algorithm utilizes a node's weight to determine the eligibility of a node's being a cluster head, where the weight can be the value of a local system parameter or a combination of local system parameters.

BIOGRAPHIES

Zhaowen Xing received his B.S. degree in Mathematics from Suzhou Railway Teachers College, China, in 1995, M.S. degree in Mathematics and M.S. degree in Computer Science from the University of Oklahoma, in 2001 and 2002, respectively. From July 1995 to May 1999, he was a MATH teacher at Beijing Nankou Railway Middle School, China. From January 2003 to May 2006, he worked for Oklahoma State Department of Health, USA, as a web developer and an application specialist. As a Ph.D. candidate, His research interests include mobile ad hoc networks, mobile peer-to-peer networks, distributed databases and mobile databases.

Le Gruenwald is the Director and Presidential and Dr. David W. Franke Professor in the School of Computer Science at The University of Oklahoma (OU). She received her Ph.D. in Computer Science from Southern Methodist University in 1990. She was a Program Director at National Science Foundation, a Member of Technical Staff in the Database Management Group at the Advanced Switching Laboratory of NEC, America, a Software Engineer at White River Technologies, and a Lecturer in the Computer Science and Engineering Department at Southern Methodist University. Her major research interests include Mobile Databases, Sensor Databases, Information Privacy and Security, Web-enabled Databases, Real-Time Main Memory Databases, Bioinformatics, Data Warehouse and Data Mining. She has published more than 150 technical articles in journals, books and conference proceedings.

Keat Keong Phang is an Associate Professor at the Faculty of Computer Science & Information Technology, University of Malaya, Malaysia. He obtained his PhD from the University of Malaya in 2004. He teaches object-oriented programming, computer networks and distributed systems. His current research interests include high speed computer networks, network quality of service, grid computing, mobile ad hoc networks, distributed systems, distributed databases and fuzzy logic.