Mobile Netw Appl (2007) 12:381–391
DOI 10.1007/s11036-008-0039-3

Multi-hop Clustering Based on Neighborhood Benchmark in Mobile Ad-hoc Networks

Stephen S. Yau · Wei Gao

Published online: 4 April 2008

Abstract Large-scale mobile ad-hoc networks require flexible and stable clustered network structure for efficient data collection and dissemination. In this paper, a technique is presented to construct multi-hop clusters with balanced sizes, based on the neighborhood benchmark (NB) to quantify the connectivity and link stability of mobile nodes. By exploiting autonomous clusterhead selection and a specialized handshake process with the clusterheads, the nodes with highest NB scores are selected as clusterheads and all the clusters constructed are connected. The deviation of cluster sizes is kept small using a partial probability-based approach. Our technique generates highly stable multi-hop clusters with low overhead, and provides the flexibility of controlling the cluster radius adaptively for various network applications.

Keywords mobile ad-hoc networks · multi-hop clusters · neighborhood benchmark · balanced sizes

1 Introduction

The highly dynamic nature and severe resource constraints of mobile ad hoc networks (MANETs) make the flat network architecture difficult to achieve scalability and cost effectiveness in data collection and dissemination [3]. Clustered network structure can be used to overcome these difficulties because the clusterheads form a virtual backbone for restricting inter-cluster data transmission from flooding [17]. Because the virtual backbone reduces the path lengths between node pairs, the effect of node mobility to the network stability is limited to a local range, and a mobile node in the network is able to access remote counterparts in much shorter time. Currently, most of the clustering techniques in MANETs fix the cluster radius to be one hop, and select clusterheads casually without considering network conditions. Hence, the clusters constructed by these existing clustering techniques [17] have low stability and flexibility.

In this paper, we will present a technique to construct multi-hop clusters with balanced sizes based on the neighborhood benchmark (NB) scores of mobile nodes in MANETs. Our approach conducts network initialization and cluster formation at run-time without the "frozen period" assumption [17], and hence is more realistic in practice. The NB quantifies the connectivity and link stability of mobile nodes using the nodes' neighbor degrees and the link failures encountered in unit time. Therefore, comparing to other mobile nodes in the network, the clusterheads selected based on their NB scores have better efficiency as aggregation points of data flows, and better stability against link failures. We will show that the clusters so constructed are connected through a handshake process, and that the deviation of cluster sizes due to autonomous clusterhead selection is controlled without degradation of the quality of clusters.

S. S. Yau · W. Gao (✉)
Department of Computer Science and Engineering,
Arizona State University, Tempe, AZ 85287-8809, USA
e-mail: wgao6@asu.edu

S. S. Yau
e-mail: yau@asu.edu

2 Current state of the art

Network clustering has been well studied in various types of decentralized p2p and wireless networks. Some combinatorial clustering algorithms based on abstracted network topology were developed, regardless of the network constraints [17]. In these algorithms, clusterheads are selected randomly, and a "frozen period" is assumed, such that all the mobile nodes can be static in the cluster formation. With this assumption, mobile nodes are able to exchange accurate information with their neighbors, and thus the clusters can be formed with some specific characteristics. However, this assumption is unrealistic because mobile nodes are not static and usually moving all the time.

The most common technique for selecting random clusterheads is the lowest-ID technique [7], in which each node is assigned with a random ID, and the node with the lowest ID in a neighborhood is assigned as the clusterhead. Various clustering algorithms have been presented based on the lowest-ID technique [5, 6, 15]. In [15], clusterheads were selected to construct a connected dominating set (CDS). Since the problem of finding a CDS with the minimum size in a connected graph is NP-complete, approximation algorithms were developed to minimize the size of the dominating set and the computational complexity. In [5], a weakly CDS with a size smaller than CDS in [15] was constructed by relaxing the requirement of direct connection between neighboring dominating nodes. Least Cluster Change (LCC) [6] increased the cluster stability by relinquishing the requirement that a clusterhead should always bear some specific attributes in its local area. In [1], the scope of CDS was expanded to d-hop, and a max-min heuristic was developed for clusterhead selection based on the NP-completeness proof of the problem of finding a d-hop CDS.

All the above clustering algorithms have very limited usage in practice because they generally ignore the actual network conditions and constraints, such as the mobility pattern and communication capability of nodes. Alternative approaches have been developed to overcome these difficulties [2, 4, 9, 11, 14]. Highest Connectivity Clustering (HCC) [14] takes the node connectivity into account, but clusters so constructed are unstable because a node is forced to change its clusterhead once it finds another clusterhead with a higher connectivity. Some methods [4, 11] were developed to improve connectivity-based clustering. However, simply using the connectivity of mobile nodes as the selection criterion of clusterheads is not sufficient to incorporate the actual network conditions in the clustered network structure, such as the dynamic network topology and volatile wireless channel characteristics. Mobility support is another issue which has been ignored in these approaches. Mobility-aware clustering has been considered [2, 9], but most of them are based on certain assumptions of the network mobility patterns, which only fit the specialized network scenarios.

Comparing to the 1-hop clustering techniques based on CDSs [5, 15], which are currently widely used in scalable routing, multi-hop clusters with controllable cluster radius provide more flexibility for sub-structures within clusters, and are more stable in dynamic network topology because a node has a higher chance to have multiple connections to its selected clusterhead, and a lower chance to lose its connection to its clusterhead. Kim et al. [8] first defined a k-hop cluster for multi-hop clustering in MANETs. Some extensions have been made to ensure the connectivity among clusters, and to reduce the number of gateway nodes connecting the clusterheads and the overhead for the k-hop cluster construction [10, 16]. In [9], multi-hop clusters are constructed by grouping the nodes with a similar mobility pattern together. However, possible inconsistency and conflict during the process of clusterhead selection is generally ignored. None of these techniques guarantee their constructed multi-hop clusters are connected with balanced sizes.

3 Our approach

A mobile ad-hoc network normally consists of nodes with heterogeneous mobility and link characteristics. We assume that all the mobile nodes in the network have omni-directional antenna and all the network links are bi-directional. The NB score of a mobile node N_i used to indicate the qualification of this node to be a clusterhead is defined as

$$NBS_i = d_i / LF_i \tag{1}$$

where d_i is the neighbor degree of N_i indicating the connectivity of N_i's neighborhood, and LF_i is the number of link failures encountered by N_i in unit time indicating the link stability of N_i's neighborhood.

Our approach to constructing R-hop clusters consists of the following four steps:

1) *Network initialization*. Every mobile node periodically sends hello beacons to its neighbors for them to calculate and exchange their initial NBSs. The interval of hello beaconing is set according to the mobility and communication ranges of mobile nodes.

2) *Autonomous clusterhead selection*. After the network initialization, every node selects its clusterhead autonomously based on the NBSs of its neighbors, in R consecutive selection iterations.
3) *Handshake with clusterheads*. After a node finishes its clusterhead selection, it starts a handshake process with its selected clusterhead, to build up its cluster membership and cluster structure.
4) *Cluster maintenance.* After a node completes the handshake with its clusterhead, it starts to conduct bilateral beaconing with its clusterhead to maintain the cluster structure. Such cluster maintenance applies to the entire network lifetime.

These steps will be elaborated in the following sections.

4 Network initialization

The network initialization is done via hello beaconing, i.e., all the mobile nodes periodically send hello beacons to their neighbors. Such beaconing process is conducted independently on individual nodes, which does not rely on global time synchronization. The interval of such hello beaconing is $T_H = r_c/v_c$, where r_c is the average transmission range of the nodes, and v_c is the average moving speed of mobile nodes in the network. Hence, such an interval is positively proportional to the global network connectivity, and inversely proportional to the network mobility level.

Since in MANETs the link failures may be due to node mobility, channel interference and/or node power depletion, the network initialization must be conducted at run-time, instead of a "frozen period".

The hello beacons sent by node N_i are encapsulated as $\{IP_i, NBS_i, Head\_IP_i, Head\_NBS_i, hop_i, size_i\}$, where IP_i and NBS_i are for N_i, and $Head\_IP_i$ and $Head\_NBS_i$ are for the current clusterhead of N_i, hop_i indicates the hop count from N_i to its current clusterhead with the range [0, R], and $size_i$ is the size of the cluster to which N_i belongs. Specifically, if N_i is in the process of clusterhead selection, $Head\_IP_i$ and $Head\_NBS_i$ are for the current temporary clusterhead selected by N_i. Initially, NBS_i, $Head\_IP_i$, $Head\_NBS_i$, hop_i and $size_i$ are set to be -1.

Every node maintains a neighborhood information table (NIT), which records and updates all the information included in the hello beacons from the node's neighbors. If the record of a neighbor in the NIT of any node has not been updated longer than $2T_H$, the neighbor is considered unreachable by the node, and one link failure of the node is counted. Nodes count the numbers of their neighbors and link failures before they send out the hello beacons, and update their NBSs using Eq. 1 and LF_i given in Eq. 2. LF_i is calculated iteratively as follows:

$$LF_i^{(k)} = (LF_i^{(k-1)} \cdot (k-1) \cdot T_H + LF_{new})/(k \cdot T_H) \quad (2)$$

where k is the current hello beaconing period, and LF_{new} is the number of link failures in this period.

It takes the time $\lceil SIZE/r_c \rceil \cdot T_H$ to initialize the NBSs of mobile nodes in the entire network, where $SIZE$ is the size of network application area,. Hello beaconing will be conducted continuously in the network lifetime to keep updating the NBSs of mobile nodes.

5 Construction of multi-hop clusters

5.1 Autonomous clusterhead selection

The clusterheads are selected in an autonomous manner based on the NBSs of mobile nodes. As defined in Section 3, the NBS of a mobile node quantifies the connectivity and link stability of the node. Our approach ensures that every clusterhead has its NBS higher than the NBS of any of its cluster members, and hence the selected clusterheads are as efficient as the aggregation points for forwarding data, and are stable to avoid frequent clusterhead changes.

Autonomous clusterhead selection is conducted on all the mobile nodes in parallel after the network initialization. The clusterhead selection process on each mobile node consists of R iterations, where R is the cluster radius in terms of the maximum number of hops from a node in the cluster to the clusterhead. In each iteration, a node N_i puts all the clusterheads of its 1-hop neighbors, and its own clusterhead into a selection pool. If N_i does not have its clusterhead yet, it uses itself as its clusterhead. Then, N_i selects the node with the highest NBS in the selection pool to be its clusterhead. N_i's clusterhead is updated in each iteration, and is finalized in the last iteration of the selection process.

The cluster radius R is pre-selected by the node owners, and different clusters can have different cluster radii. We will show in Section 7 that by choosing different cluster radius according to the specific conditions of network environments, the constructed clusters can achieve different tradeoffs among the cluster stability, construction overhead, and cluster coverage.

The interval between two iterations is set to be the same as the interval of hello beaconing described in Section 4. Hence, on a mobile node, each iteration in

the clusterhead selection process is corresponding to a hello beaconing period. Any iteration will not start until the node receives all the corresponding hello beacons from its 1-hop neighbors. The clusterhead selection process can be shown correct as follows:

Theorem 1 *The kth iteration on a node N_i selects the node with the highest NBS within the k-hop neighborhood of N_i as N_i's clusterhead.*

Proof This theorem is proved by induction. For the first iteration, the theorem is self-evident. Assume that the theorem holds after the mth iteration. Because each iteration corresponds to a hello beaconing period, N_i will know the clusterheads of its 1-hop neighbors before the $(m+1)$th iteration, via the hello beaconing. These clusterheads are the nodes with the highest NBSs in the m-hop neighborhoods of N_i's 1-hop neighbors. Hence, in the $(m+1)$th iteration, all the possible new elements in N_i's selection pool are N_i's $(m+1)$-hop neighbors. □

Based on Theorem 1, we can easily derive the following corollary, which ensures that a selected clusterhead has the NBS larger than that of any of its cluster members:

Corollary 1 *Given a cluster $C=\{N_i|\ i=0, 1, \ldots..n\}$, if a node $N_k \in C$ is the clusterhead of C, then for $\forall i \in [0, n]$, $i \neq k$, $NBS_k \geq NBS_i$.*

5.2 Handshake with clusterheads

In a multi-hop cluster, the clusterhead has the complete member list of the cluster, and the cluster members only record their clusterhead. After clusterhead selection, a mobile node handshakes with its selected clusterhead to construct the multi-hop cluster. Such handshake process can detect possible inconsistency during the handshake, and ensures that all the constructed clusters are connected. Such process is described in Fig. 1 using a case including a node N_0, its selected clusterhead N_h, and an intermediate node N_i. It is noted that because of the autonomous and simultaneous clusterhead selection, all the mobile nodes also handshake with their selected clusterheads simultaneously. In such cases, if a node in its handshake process is requested to be a clusterhead by some other nodes, and the request is accepted, it should send another message to its selected clusterhead to cancel the ongoing handshake process, and start to handshake with the requesters as a new clusterhead.

An example of 2-hop clusters is shown in Fig. 2. Because every mobile node only has the knowledge of its neighborhood, there will be possible conflict and inconsistency during the autonomous cluster selection and handshake process. For example, in Fig. 2a, in the first cluster selection round, N11 selects N3 as its clusterhead, and notifies N9 of this via hello beaconing. In the second round, because the NBS of N3 is higher than that of N1, N9 also selects N3 as its clusterhead. At the same time, N11 hears from N23 that N6 has a higher NBS. Since N11 has no idea about N9's choice by that time, N11 will change its clusterhead to N6. However, this change will break the link from N9 to N3, and end up with a disconnected cluster.

This problem is avoided by the handshake process such that a node will only forward the clusterhead requests if it is in the same cluster. Hence, in the above example, N9's clusterhead request cannot reach N3 because N11 will not forward it, and N9's handshake process will timeout and another clusterhead selection process starts. The clusters constructed in this example are shown in Fig. 2b. In this way, we can guarantee that all the constructed clusters are connected.

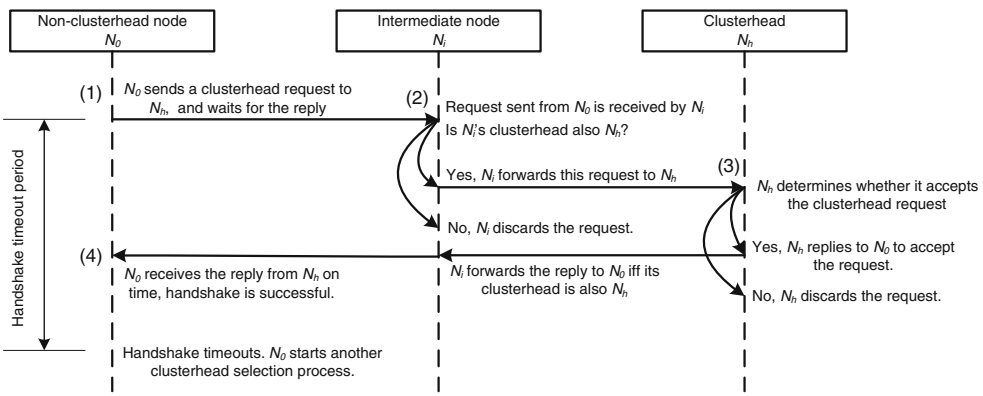

Figure 1 The handshake process with clusterheads in our approach

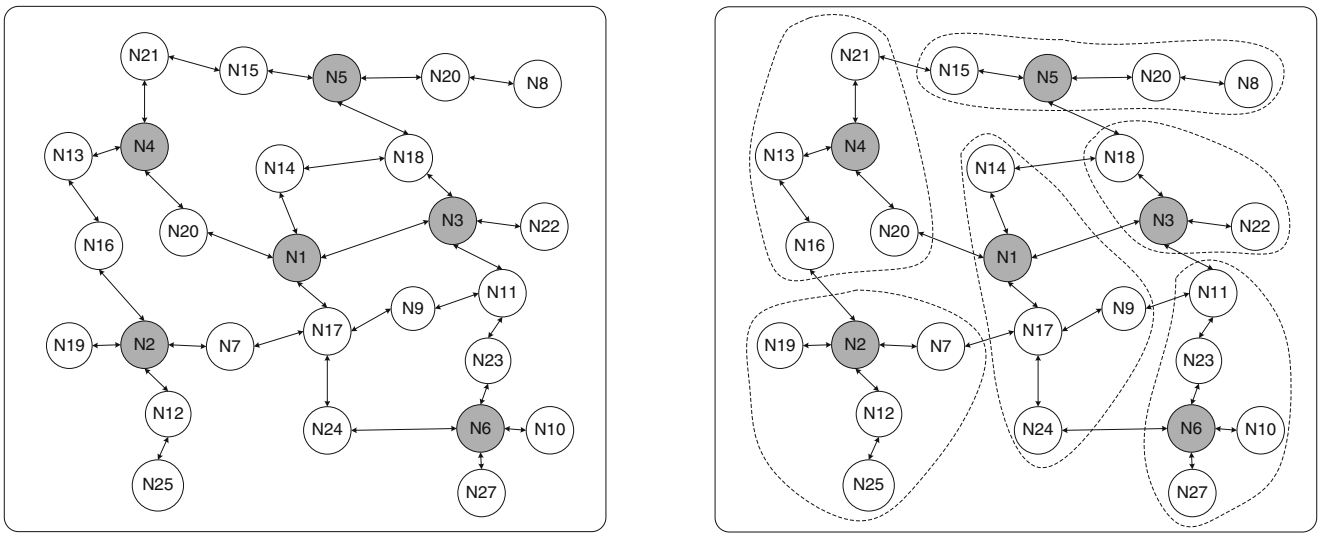

Figure 2 An example of 2-hop clusters: **a** before handshake and **b** after handshake

Theorem 2 *For any node N_i in an R-hop cluster C with the clusterhead N_h, there exists a path from N_i to N_h, such that the path only contains the nodes in C.*

Proof If N_c is an 1-hop neighbor of N_h, the theorem is self-evident. Only through the nodes which also select N_h as their clusterhead, N_c can complete the handshake process with N_h. Therefore, N_c must have at least one of its 1-hop neighbors which is also in C. The theorem can be shown by induction. □

5.3 Cluster maintenance

Cluster maintenance is conducted via bilateral beaconing. A clusterhead multicasts beaconing messages periodically at the interval T_{cb} to all the cluster members, where T_{cb} is set to be the same as the hello beaconing interval T_H. Every cluster member returns an acknowledgment to its clusterhead upon receiving the beacon message. Only the nodes within the same cluster will forward the beaconing messages and acknowledgments.

In order to keep stable cluster structures, a non-clusterhead node searches only for its new clusterhead if its current clusterhead is unavailable. If a non-clusterhead node has not received the beacon message from its current clusterhead longer than $2T_{cb}$, it considers its current clusterhead unreachable, and reselects its clusterhead. If the clusterhead has not received the acknowledgment from a cluster member longer than $2T_{cb}$, the clusterhead considers the member unreachable and deletes the member from its member list.

6 Balancing cluster sizes

In a clustered network structure, clusterheads are the aggregation points of data flows and hence consume their resources faster. On the other hand, it is not desirable to change clusterheads too frequently because a large number of nodes will need to re-select their clusterheads, and this leads to large communication overhead. Clusters with balanced sizes are preferred to increase the network stability and lifetime.

In Section 5.1, every node selects its clusterhead deterministically based on the NBSs without considering size balancing. Hence, most of the selected clusterheads are nodes with higher NBSs, and clusters with large size deviation are formed. However, to construct clusters with balanced sizes will impair the quality of clusters represented by the NBSs of clusterheads. We exploit a predefined preference parameter P_t valued between 0 and 1 to tradeoff the cluster quality with size balancing according to network situations.

Based on the notation of P_t, a partially probability-based approach is used for clusterhead selection. In a clusterhead selection round at node N_0, assume that the set $\{H = N_{h1}, N_{h2}\ldots, N_{hk}\}$ is the set of possible nodes to be selected as clusterheads, and $size_i$ indicates the current cluster size of the clusterhead N_{hi}. Each N_{hi} in H has a probability p_i to be selected as the clusterhead of N_0. Such a probability is made up of a singular part and a common part.

- The singular part $p_{si} = 1 - P_t$ when N_{hi} has the highest NBS in H; otherwise, p_{si}=0.
- The common part p_{ci} is defined as

$$p_{ci} = ((1 - P_t) \cdot NBS_i + P_t \cdot E_i) \cdot P_t, \quad (3)$$

where E_i is the normalized entropy of N_i given by

$$E_i = \frac{\ln(S/size_i)}{\sum_{j=1}^{k} \ln(S/size_j)} \quad (4)$$

$$S = \sum_i size_i \quad (5)$$

In H, the entropy of N_i is $\ln(S/size_i)$, which is inversely proportional to $size_i$, and the normalized entropy ensures that

$$\sum_i p_i = \sum_i (p_{si} + p_{ci}) = 1 \quad (6)$$

Due to the common part of p_i, N_{hi} with the higher NBS or in a smaller cluster has higher probability to be selected as a clusterhead. When N_{hi} has the highest NBS in H, it is given extra preference by the singular part of p_i, $1 - P_t$. When $P_t = 0$, p_{ci}=0, clusterheads are selected deterministically without considering size balancing. When $P_t = 1$, p_{si}=0, and $p_{ci} = E_i$. Then, clusterheads are selected probabilistically according to E_i defined in Eq. 4 to reduce the deviation of cluster sizes, without considering nodes' NBSs. Any intermediate value of P_t between 0 and 1 produces a probabilistic distribution to tradeoff cluster quality with balanced cluster sizes.

7 Performance evaluation

We have simulated our technique using ns-2 with the CMU wireless extensions. We have evaluated the performance of our clustering approach with respect to the stability of the constructed clusters, the communication overhead of clustering, the time length of the cluster construction, and the coverage ratio of clusters to the entire network. We have also evaluated the effects of our size balancing mechanism on the deviation of cluster sizes, cluster stability and the cluster coverage.

In the simulations, we uniformly deployed 50 mobile nodes in a $1000 \times 1000\ m^2$ square area. 802.11 WLAN was used as the underlying MAC protocol. We assume that all the nodes have omni-directional antennas and uniform communication range of 250 m, and the two-way ground propagation model was used. The node mobility follows the random-walk mobility model [9] with the node moving speeds normally distributed in a range $[0, v_{max}]$. Each simulation lasts 5000 s, and each point in the simulation figures is averaged over 10 random simulation scenarios.

7.1 Performance of multi-hop clustering based on NB

The performance of multi-hop clustering based on neighborhood benchmark (MCNB) with different cluster radius was evaluated and compared to two traditional clustering methods LCC [6] and HCC [14]. Figure 3 shows the evaluation result of cluster stability represented by the average number of clusterhead changes per node during the simulation. Compared to HCC, because MCNB does not force a node to change its clusterhead when it discovers another clusterhead with higher NBS, the cluster stability is greatly improved. The cluster stability inevitably degrades when the node mobility increases. It is shown that larger cluster radius can mitigate the effects of node mobility because a node has a smaller chance to lose its clusterhead connection in a larger and heavier connected cluster, which enables every node to reach its clusterhead via multiple paths. Even when R=1, the performance of MCNB is similar to LCC which is aiming at minimizing the cluster changes. As shown in Fig. 3, when a larger cluster radius R is used, the average number of clusterhead changes of MCNB is nearly 50% lower than LCC when R=3.

Figure 4 shows the communication overhead for clustering evaluated by the average transceiving and receiving messages per second per node through the entire simulation process. Such overhead includes both cluster construction and maintenance cost. Since MCNB restricts all the beaconing mechanisms to a localized scope, and does not enforce proactive clusterhead reselection, its clustering overhead is reduced, compared to HCC and LCC. It can be seen that the overhead of MCNB is 20% lower than that of LCC, and such reduction is up to 50% when compared to HCC.

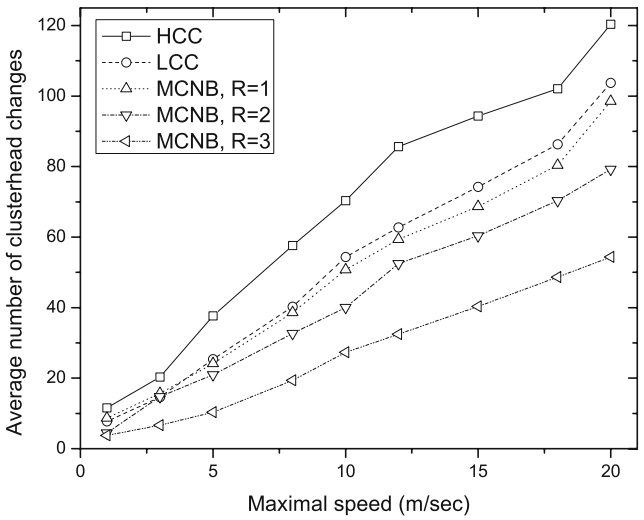

Figure 3 Cluster stability of our simulation settings

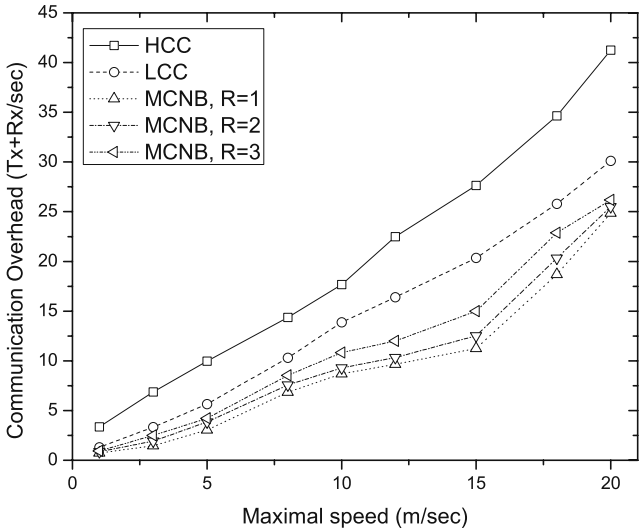

Figure 4 Communication overhead of our simulation settings

The difference of MCNB clustering overhead with different cluster radii is very small because the major overhead of clustering is restricted to the small neighborhood of individual mobile nodes, and the overhead of clusterhead selection, which is the only overhead affected by cluster radius, is not the major overhead of clustering. On the other hand, the cluster overhead increases from 5 messages per second to 25 messages per second when the average node mobility increases from 1 m/s to 20 m/s. Such increase is inevitable and mainly because it needs more frequent hello beaconing in scenarios with higher mobility to calculate accurate NBSs of the neighbors. It can be seen that such increase is even higher in HCC and LCC.

Figure 5 shows the cluster coverage, i.e., the percentage of nodes being clustered with various cluster radii. A number of nodes may have their clusterhead lost due to network topology changes. However, our approach can guarantee that a majority of nodes are clustered over a long time. It is shown that, the cluster coverage can achieve above 80% in mobility scenarios with various average mobility levels, and such coverage is only slightly affected by the increase of node mobility level. Figure 5 shows that the coverage suffers only a 5% deviation in different mobility levels.

On the other hand, it is shown that using a larger cluster radius decreases the cluster coverage. When the cluster radius increases from 1 to 3, the coverage percentage on the average drops from 90% to 80%. This is because the construction of larger clusters needs more complicated clusterhead selection and handshake process. When the cluster radius is larger and more clusterhead selection rounds are needed to decide the clusterhead, the possibility of inconsistent cluster membership will increase.

Figure 6 shows the convergence performance of the clustering process through the average time length of clusterhead selection. It shows that selecting larger cluster radius will greatly increase the time needed for the clusterhead selection. When the average node mobility level is 1 m/s, 1-hop clusters only need 20 s on the average to get the clusterhead selection process to convergence, while that time for 3-hop clusters is up to 260 s. This is obvious because even small increase of the cluster radius will lead to much larger increase of the range of each cluster, and thus it needs more time for a mobile node to find the best clusterhead candidate. On the other hand, the average time decreases greatly when the average node mobility increases. When the average node mobility is very high (20 m/s), the average time of clusterhead selection with different cluster radii

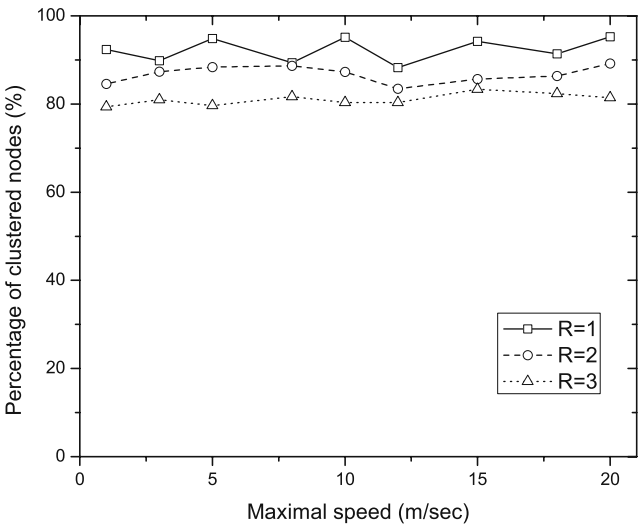

Figure 5 Cluster coverage of our simulation settings

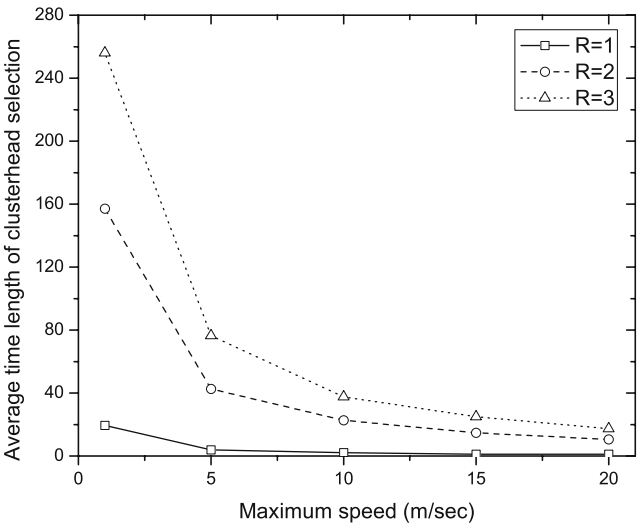

Figure 6 Time length of clusterhead selection of our simulation settings

will all be less than 20 s because the period of hello beaconing in high mobility scenarios is correspondingly much shorter. However, this also leads to much larger communication overhead, as shown in Fig. 4.

The advantages of multi-hop clusters constructed based on NB are their stability and flexibility. Different cluster radii can be used for our technique to adapt to various network applications. A larger cluster is more stable with network topology changes, and provide more flexibility to construct sub-structures within the cluster, but the construction of the cluster causes more communication overhead among network nodes and the cluster can only cover up to 80% of the network nodes as shown in Fig. 5. On the other hand, a network structure with smaller clusters is more volatile and sensitive to network topology changes, but smaller clusters are also easier to be constructed, and can provide better node coverage. Hence, larger clusters are suitable for data-intensive applications to achieve higher efficiency of data delivery, and small clusters are suitable for applications in severe physical environments to achieve larger node coverage and more flexible reconfigurability.

7.2 Performance of our technique with balancing cluster sizes

We have also evaluated the performance of our technique with balancing cluster sizes in various values of the preference parameter with a cluster radius R=2. In Fig. 7, the effect of balancing cluster sizes is shown by the standard deviation of cluster sizes. Figure 7 shows that the deviation of cluster sizes will be greatly reduced when larger P_t is used to put more emphasis on balanced cluster sizes. When size balancing was not considered, i.e., $P_t = 0$, the deviation can be up to 1.5; and when P_t increased from 0 to 1, such deviation decreased correspondingly down to 1 (50%). However, since the singular part of the clusterhead selection probability described in Section 6 gives extra preference to the node with the highest NBS, the effect of size balancing can be apparent only when $P_t \geq 0.7$. On the other hand, the deviation of cluster sizes is not affected much by the node mobility. Such deviation is smaller when the node mobility is high because an arbitrary node has more clusterhead choices in a high-mobility network scenario.

Figure 8 shows that constructing clusters with size balancing did not impair the cluster stability. When P_t

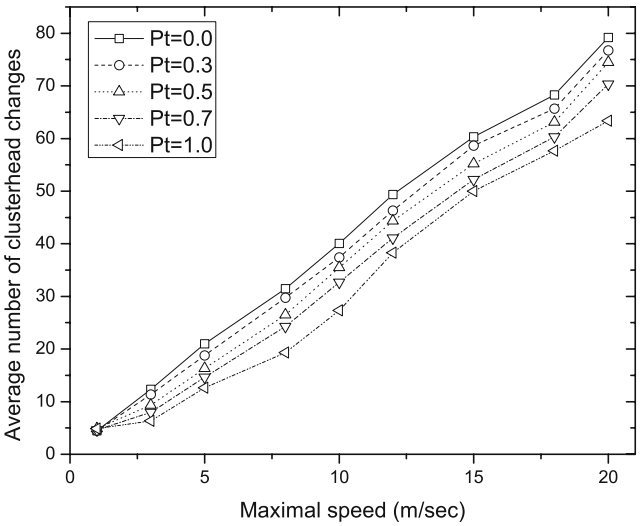

Figure 8 Effects of size balancing to cluster stability of our simulation settings

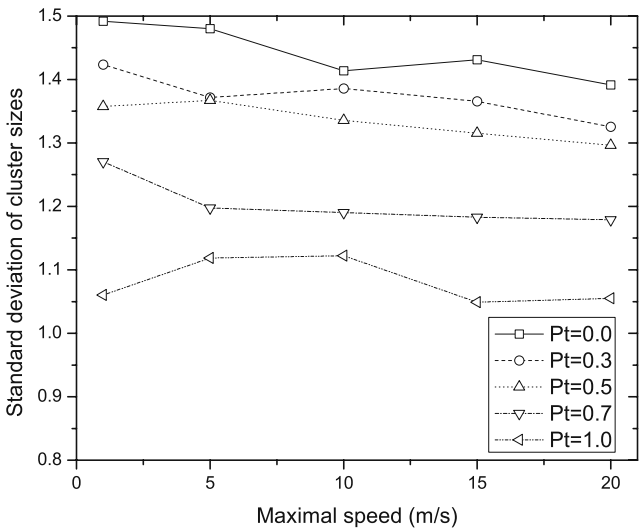

Figure 7 Effects of size balancing of our simulation settings

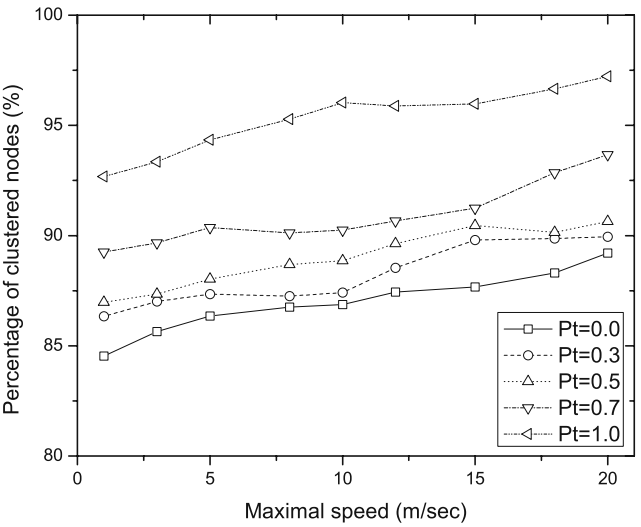

Figure 9 Effects of size balancing to cluster coverage of our simulation settings

varied from 0 to 1, the average number of clusterhead changes even slightly reduced by approximately 20%. Figure 9 also shows that our size balancing approach increased the cluster coverage. When P_t increased from 0 to 1, the cluster coverage in different mobility scenarios gained an average increase of 10%. When the size balancing technique was applied in the cluster construction process, clusters were expanded evenly in all directions, and hence had higher possibilities to cover more nodes in the network.

8 Potential improvement with mobility-awareness capability

Currently, as described in Section 3, we have used NBSs as the criterion of clusterhead selection. Such criterion take both the node connectivity and link stability into consideration and hence guarantees the efficiency and stability of the selected clusterheads in various mobile applications. In the NBS-based clusterhead selection process, we consider node mobility implicitly because nodes with low mobility are likely to have more stable neighborhood and thus have higher NBSs. In fact, node mobility can be considered explicitly to improve the adaptability of our approach in mobile applications. For example, node mobility can be explicitly integrated into the expression of NBS as follows:

$$NBS_{ij} = d_i / v_{ij} \tag{7}$$

where v_{ij} is the relative speed between node N_i and N_j. In this case, for the same mobile node N_i, the calculation of NBSs of other different mobile nodes will be different.

There are two major problems for explicit mobility consideration. The first is how the individual mobile nodes are able to characterize the mobility features of its neighborhood without global knowledge. The second is how to keep the continuity of such mobility characterization in case of node failures in MANETs.

For the first problem, the major challenge is how to find an appropriate formulation of the localized node mobility. On one hand, due to the self-organizing nature of MANETs, a mobile node without a GPS receiver or fixed anchor nodes can observe only its neighborhood mobility based on its local coordinate system, which invalidates the usage of the global mobility metrics and requires an efficient localized mobility metric. On the other hand, although any node mobility pattern in general can be considered as a continuous-time stochastic process, and a k-order Markov process if the pattern has finite memory level, it is hard to find an appropriate way to formulate such mobility patterns to associate the actual node mobility with abstract states in Markov processes defined in a finite field. Instead of formulating node mobility patterns using normal Markov processes, we may exploit Hidden Markov Models (HMMs) to associate the numeric node mobility levels with the abstract Markovian states through the observation probability density functions defined in HMMs. However, it is still necessary to determine how to use the Markovian states defined on a finite discrete field to express the node mobility on a continuous numeric range.

The second problem is due to the fact that node neighborhoods in MANETs are highly dynamic. The neighbors of a mobile node can become unavailable due to their mobility or energy depletion, and they may reappear again in the future.

A direct way to handle such neighbor unavailability for mobility characterization is to predict the mobility of the unavailable neighbor in the future. Such prediction must imply certain assumption on the mobility pattern of the unavailable neighbors, which affects the validity of the mobility prediction. The simplest assumption is that the node mobility in the future is linearly dependent of the past p mobility records, and thus the node mobility pattern is formulated by the following p-order auto-regressive (AR) process:

$$X_t = \phi_0 + \phi_1 X_{t-1} + \ldots + \phi_p X_{t-p} + \varepsilon_t \tag{8}$$

where X_t, X_{t-1}, ..., X_{t-p} are the mobility observations, ε_t is an error term with the distribution $N(0, \sigma^2)$, and $\hat{\phi}_0, \hat{\phi}_1, \ldots, \hat{\phi}_p$ are the estimated values by solving the following non-linear least-square regression:

$$\min_{\phi_0,\phi_1,\ldots,\phi_p} \sum_{t=p+1}^{L} \frac{(X_t - \phi_0 - \phi_1 X_{t-1} - \ldots - \phi_p X_{t-p})^2}{2\sigma^2} \tag{9}$$

where σ^2 is given by the Residual Sum of Squares of the regression

$$\hat{\sigma}^2 = \frac{1}{L-p} \sum_{t=p+1}^{L} (X_t - \hat{\phi}_0 - \hat{\phi}_1 X_{t-1} - \ldots - \hat{\phi}_p X_{t-p})^2 \tag{10}$$

We may consider more complicated dependency forms for the mobility predictions, such as a p-order polynomial, but this will greatly increase the complexity of calculating the likelihood function, and solving the non-linear regression for parameter estimation.

9 Conclusion and future work

In this paper, we have presented a technique to provide more flexible and stable clustered network structure

for efficient data collection and dissemination by constructing multi-hop clusters in MANETs based on the NBSs of mobile nodes. The multi-hop clusters are constructed by letting every node autonomously select its clusterhead based on the NBSs and handshake with the clusterheads, so that all the clusters constructed are connected. We have also presented a partial probability-based approach to control the possible deviation of cluster sizes. We have intensively simulated our approach, and the results indicate that our clustering technique can provide stable clustered network structures with balanced cluster sizes in various network scenarios, and provide users with the flexibility of controlling the cluster radius adaptively for different applications.

In the future, since the multi-hop clusters constructed by our approach have better flexibility and stability than traditional 1-hop clusters in various mobility scenarios, our approach can be used in various mobile applications for performance optimization, especially for improving the scalability and robustness of service discovery in MANETs.

Because service discovery is needed in various types of mobile environments for users to discover their needed services, and because of the severe energy constraints and heterogeneity of mobile computing devices which collaborate with each other in an ad-hoc manner in large-scale pervasive computing environments, service discovery needs not only to be *scalable* to discover the needed services quickly, accurately, and efficiently, but also to be *robust* against node failures and network topology changes. To satisfy such requirements, we need a clustered network architecture, and employing only a small part of the network nodes for the service discovery functionality. The multi-hop clusters constructed by our approach, due to their better flexibility and stability, are more suitable for scalable and robust service discovery.

Multi-hop clusters also provide the possibility to enhance the robustness of service discovery against unpredictable node and link failures, by constructing local Distributed Hash Tables on the multi-hop clusters, and storing multiple service replicates in each cluster according to the local hashing results. To do this, multiple existing p2p network architectures, such as CAN [12], Chord [13] and Tapestry [18] can be exploited.

Acknowledgements This work was supported by the National Science Foundation under grant number ITR-CYBERTRUST 0430565.

References

1. Amis AD, Prakash R, Huynh D, Vuong T (2000) Max-min d-cluster formation in wireless ad hoc networks. In: Proc. 19th annual joint conference of the IEEE computer and communications societies (INFOCOM). IEEE, Piscataway, pp 32–41
2. An B, Papavassiliou S (2001) A mobility-based clustering approach to support mobility management and multicast routing in mobile ad-hoc wireless networks. Int J Netw Manage 11(6):387–395
3. Belding-Royer EM (2002) Hierarchical routing in ad-hoc mobile networks. Wirel Commun Mob Comput 2(5):515–532
4. Chatterjee M, Das SK, Turgut D (2002) Wca: a weighted clustering algorithm for mobile ad hoc networks. J Clust Comput 5(2):193–204
5. Chen Y-P, Liestman AL (2002) Approximating minimum size weakly-connected dominating sets for clustering mobile ad hoc networks. In: Proc. 3rd ACM interational symposium on mobile ad hoc networking and computing (MobiHoc). ACM, New York, pp 165–172
6. Chiang C-C, Wu H-K, Liu W, Gerla M (1997) Routing in clustered multihop mobile wireless networks with fading channel. In: Proc. IEEE Singapore international conference on networks (SICON). IEEE, Piscataway, pp 197–211
7. Ephremides A, Wieselthier JE, Baker DJ (1987) A design concept for reliable mobile radio networks with frequency hopping signaling. Proc IEEE 75(1):56–73
8. Kim D, Ha S, Choi Y (1998) k-hop cluster-based dynamic source routing in wireless ad-hoc packet radio networks. In: Proc. IEEE vehicular technology conference (VTC). IEEE, Piscataway, pp 224–228
9. McDonald A, Znati TF (1999) A mobility-based framework for adaptive clustering in wireless ad hoc networks. IEEE J Sel Areas Commun 17(8):1466–1487
10. Nocetti FG, Gonzalez JS, Stojmenovic I (2003) Connectivity based k-hop clustering in wireless networks. Telecommun Syst 22(1–4):205–220
11. Ramaswamy L, Gedik B, Liu L (2005) A distributed approach to node clustering in decentralized peer-to-peer networks. IEEE Trans Parallel Distrib Syst 16(9):814–829
12. Ratnasamy S, Francis P, Handley M, Karp R, Schenker S (2001) A scalable content-addressable network. In: Proc. ACM SIGCOMM. ACM, New York, pp 161–172
13. Stoica I, Morris R, Karger D, Kaashoek F, Balakrishnan H (2001) Chord: a scalable peer-to-peer lookup service for internet applications. In: Proc. ACM SIGCOMM. ACM, New York, pp 149–160
14. Tsai JT, Gerla M (1995) Multicluster, mobile, multimedia radio network. J Wirel Netw 1(3):255–265
15. Wu J, Li H (1999) On calculating connected dominating set for efficient routing in ad hoc wireless networks. In: Proc. 3rd Int'l workshop on discrete algorithms and methods for mobile computing and communications (DIAL-M), Seattle, 20 August 1999, pp 7–14
16. Wu J, Lou W (2003) Forward node set based broadcast in clustered mobile ad hoc networks. Wirel Commun Mob Comput 3(2):155–173
17. Yu JY, Chong PHJ (2005) A survey of clustering schemes for mobile ad hoc networks. Commun Surv Tutor 7(1):32–48
18. Zhao BY, Huang L, Stribling J, Rhea SC, Joseph AD, Kubiatowicz JD (2004) Tapestry: a resilient global-scale overlay for service deployment. IEEE J Sel Areas Commun 22(1):41–53

Stephen S. Yau is currently the director of Information Assurance Center and a professor in the Department of Computer Science and Engineering at Arizona State University, Tempe, Arizona, USA. He served as the chair of the department from 1994 to 2001. He was previously with the University of Florida, Gainesville and Northwestern University, Evanston, Illinois. He served as the president of the Computer Society of the IEEE (Institute of Electrical and Electronics Engineers) and the editor-in-chief of IEEE Computer magazine. His current research is in distributed and service-oriented computing, adaptive middleware, software engineering and trustworthy computing. He received the Ph.D. degree in electrical engineering from the University of Illinois, Urbana. He is a life fellow of the IEEE and a fellow of American Association for the Advancement of Science.

Wei Gao received the B. S. degree from University of Science and Technology of China in 2005. He was a Ph. D. student in the Distributed and Parallel Software Engineering Laboratory at Arizona State University. He is currently working towards the Ph. D. degree at The Pennsylvania State University, University Park. He is a student member of IEEE and ACM.