Efficient Broadcast With Forward Node Set in Clustered Mobile Ad Hoc Networks

Wei Lou and Jie Wu
Department of Computer Science and Engineering
Florida Atlantic University
Boca Raton, FL 33431

Abstract— **The clustered architecture of a mobile ad hoc network (MANET) has the virtue of keeping the node information locally which is suitable for scalability. Reducing broadcast redundancy in a clustered network can avoid the broadcast storm problem and save scarce resources such as bandwidth and energy. In this paper, we propose an approach that chooses a subset of nodes, called forward node set, to relay the broadcast packet. Each clusterhead computes its forward node set that connects its adjacent clusterheads. A non-clusterhead node just relays the broadcast packet if it is selected as a forward node or else it does nothing. Therefore, the broadcast operation can be restricted only to clusterheads and nodes in locally selected forward node sets. We also utilize the information of clusterheads that are piggybacked with the broadcast packet to further reduce each forward node set. Simulation shows its performance improvement against other broadcast algorithms.**

Keywords— **Broadcast, cluster, forward node set, MANET.**

I. Introduction

A mobile ad hoc network (MANET) is a totally on-the-fly network used to support the notion of "any time, any place". It is an infrastructureless network which consists of a collection of wireless mobile hosts to form a temporary network without the aid of any wired base stations. Each mobile host in such a network commits to operate not only as a host but also as a router. The inherent limitations of the MANET, such as scarce resources and dynamic topologies, require any routing protocol designed for such an environment be simple, efficient and robust. In general, a MANET can be represented as a unit disk graph $G = (V, E)$, where V represents a set of wireless mobile hosts (nodes) and E represents a set of links between the neighbors, assuming all hosts have the same transmission range. Two hosts are neighbors if and only if they are within each other's transmission range. Therefore, the connections of hosts are based on geographic distances of hosts. The way that packets are transmitted in the MANET is quite different from that in the wired network, because when a node sends a packet, all its neighbors will receive that packet under the promiscuous receive mode.

Broadcasting a packet to the entire network is a basic operation and has extensive applications in the MANET. The straightforward approach for broadcast is blind flooding, in which each node is obligated to re-broadcast the packet whenever it receives a packet for the first time. Blind flooding will generate many redundant transmissions. These redundant transmissions may cause a serious problem, referred as the *broadcast storm problem* [8], in which redundant packets cause communication congestion and contention. In a broadcast process, the source and a subset of nodes form a flood tree such that any other node in the network is adjacent to a node in the tree. Nodes on the tree are called *forward nodes* and form a *connected dominant set* (CDS). A *dominating set* (DS) is a subset of vertices such that every vertex in the graph is either in the set or has a link to a vertex in the set. If the vertices in a DS are also connected, it is called a CDS. Finding a *minimum connected dominating set* (MCDS) in a given graph is NP-complete and it has also been proved to be NP-complete in a unit disk graph [7]. Even when an MCDS is identified, maintaining such a structure in a mobile environment is a costly operation for practical use.

This work was supported in part by NSF grants CCR 9900646 and ANI 0073736. Email:{wlou, jie}@cse.fau.edu.

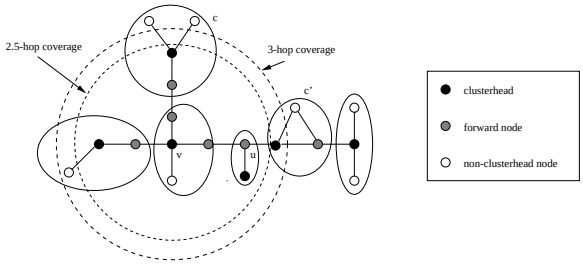

Fig. 1. A network showing 3-hop and 2.5-hop coverage areas of node v.

In this paper, we propose an efficient broadcast protocol based on the *clustered network* which is a 2-level hierarchical network. Basically, the clustered network converts a "dense" network to a "sparse" one that consists of clusterheads and some gateways. The proposed broadcast protocol uses a subset of nodes, called forward node set, to relay a broadcast packet in a clustered network. Only a clusterhead computes its forward node set to cover other clusterheads within its vicinity, say within the 3-hop coverage area as shown in Figure 1. A non-clusterhead node just relays the broadcast packet if it is selected as a forward node or else it does nothing. The forward node set is computed by clusterheads such that all the clusterheads in the network can be connected. Therefore, a broadcast packet can be delivered to the entire network eventually. The broadcast process is limited only to the clusterheads and the nodes in the forward node sets so that the broadcast redundancy can be reduced. The information about the clusterheads that will receive the broadcast packet from the sending clusterhead is also piggybacked with the broadcast packet and the receiving clusterhead can further reduce its forward node set with this information. A novel notation of 2.5-hop coverage is introduced where each clusterhead just covers the clusterheads that have members within 2 hops. That is, only partial 3-hop clusterheads need to be covered. In Figure 1, clusterhead of c is covered by v, but not clusterhead of c'. Simulation shows that the 2.5-hop coverage method has its comparable performance to the 3-hop coverage method when the pruning technique is used. It also shows that our protocol outperforms the protocols proposed in [10], [11], [12].

The paper is organized as follows: Related work is given in the next section. Section 3 gives the description of our protocol for the clustered network. In Section 4, we compare the performance of different protocols through simulations. Finally, Section 5 concludes the paper.

II. Related Work

Recently, many broadcast algorithms besides blind flooding have been proposed in [5], [8], [9], [10], [11], [12]. These algorithms utilize neighborhood and/or previous routing information to reduce redundant packets. Ni, Tseng, Chen and Sheu [8] firstly analyze broadcast redundancy, contention and collision in blind flooding and point out that blind flooding is very costly and will result in the broadcast storm problem. Algorithms for reducing broadcast redundancy, such as probabilistic, counter-based, distance-based, location-based and cluster-based schemes, are also proposed. All these algorithms require each forward node estimate network redundancy and accumulate information about the network to assist its decision.

In [5], Lim and Kim provide two heuristic algorithms for selecting the forward node set: self pruning (SP) and dominant pruning (DP). The SP algorithm only exploits the knowledge of directly connected neighbor information. A node does not re-broadcast a packet if all its neighbors have been covered by the previous transmission. The DP algorithm uses 2-hop neighbor information to compute each node's forward node set. The forward node set is selected in such a way that they cover all the nodes within 2 hops. A similar forward nodes selection algorithm is also proposed in [9]. Lou and Wu extend the dominant pruning approach in [6]. Two algorithms, *total dominant pruning* (TDP) and *partial dominant pruning* (PDP), are proposed. These approaches utilize the previous forward node's 2-hop neighbor set information to further reduce broadcast redundancies. The TDP requires the sender piggyback its 2-hop neighbor set information along the broadcast packet. With this information, the receiver can prune all the nodes in the sender's 2-hop neighbor set from the receiver's 2-hop neighbor set that needs to be covered. Apparently, the TDP will generate a smaller forward node set than the DP, but it also introduces some overhead when the broadcast packet piggybacks the 2-hop neighbor set information. The PDP, without using the piggybacking technique, directly extracts the 1-hop neighbors of the common neighbors of both sender and receiver from the receiver's 2-hop neighbor set. Simulation results show that the PDP algorithm avoids the extra cost as in the TDP introduced by piggybacking 2-hop neighbor set information with the broadcast packet, but achieves almost the same performance improvement.

Since finding an MCDS in a given graph is NP-complete in a unit disk graph, we use the following heuristic approach proposed in [2] to get an approximation of the MCDS: All nodes are initially colored white. The node with the maximum node degree is selected and colored black, all its neighbors are colored grey. A recursive selection process runs until no white node exists: Select a grey node that has the maximum number of white neighbors, color the selected node black and its white neighbors grey. The resultant set of black nodes is an approximation of the MCDS. Wu and Li [12] propose a *making process* followed by Rules 1 and 2 to form a CDS in a localized way (i.e., no sequential propagation of the information to the entire network). In the marking process, nodes only interact with their neighbors. All nodes are initially white. A node marks itself black only when it has two unconnected neighbors. After the marking process, the black nodes form a CDS. Rules 1 and 2 aim to remove redundant nodes from the CDS. Rule 1 allows a black node u to change its color to white if it can find another black node v, with $id(u) < id(v)$, to cover all u's neighbors. For Rule 2, a black node u changes itself to white if there exist two connected nodes v and w, with $id(u) = min\{id(u), id(v), id(w)\}$, that can collectively cover all u's neighbors. Wu and Li's approach has also been applied to the broadcast algorithm in [11].

Clusters are formed by first electing a clusterhead and then its neighbors joining in the cluster as non-clusterhead members. There are many clustering approaches [1], [3], [4]. A simple one, the lowest-ID cluster algorithm, initializes all nodes to be white. When a white node finds itself have the lowest ID among all its white neighbors, it becomes a clusterhead and colors itself black. All its white neighbors join in the cluster and change their colors to grey. The process continues until there is no white node. The black nodes form the set of clusterheads. Each grey node belongs to one and only one clusterhead. That clusterhead is also called the *dominator* of the grey node. The clusterhead and its dominated grey neighbors form a cluster. The lowest-ID cluster algorithm may exhibit sequential propagation when the network is a linear chain with decreasing IDs from one end to the other end. Other clustering formations are possible: Clusters can be formed to be overlapped or disjointed; clusterheads can be elected by choosing the node with the lowest node ID (LID) or the one with the highest node degree (HD) [4]. Because the set of clusterheads is an *independent set* of the network (i.e., no two clusterheads are connected directly), gateways are needed to connect clusterheads together. A gateway is a non-clusterhead node in a cluster that has a neighbor in another cluster. The clusterheads and gateways together form a CDS of the network. The cluster-based broadcast algorithm only requires nodes in this CDS forward the broadcast packet. With this strategy, except for the CDS, other nodes do not participate in the packet forwarding process. Since two neighboring clusters usually have more than one gateway to forward the broadcast packet, the cluster-based broadcast still has more broadcast redundancy compared with flood-tree-based broadcast.

In [10], Sinha, Sivakumar and Bharghavan propose an approach called *core broadcast* (CB): Initially all nodes are white. A white node will determine its dominator by selecting its black neighbor which has the maximum number of nodes that regard this black node as their dominators. In case there is no black neighbor, the white node selects the node (white or grey) with the maximum node degree within its 1-hop neighborhood as its dominator. After the white node has chosen its dominator, it colors itself grey if it is not selected as a dominator by itself or by its neighbors; otherwise, it marks itself black if it has been selected as a dominator (probably selected by itself). The coloring process continues until no white node exists. Eventually, all the black nodes become cores. In the core broadcast, each node computes its forward node set. A node's forward node set includes all its black neighbors. It also includes those grey neighbors that either have a black neighbor which is not *covered* by the forward node set or have a grey neighbor whose dominator is not covered by the forward node set. A node is said to be covered by a forward node set if it is a member of the forward node set or a neighbor of any member of the forward node set. The core broadcast requires only the nodes in the forward node set relay the broadcast so it reduces the broadcast redundancy. The set of cores, like the set of clusterheads, is a DS of the network. While the set of clusterheads is also an independent set, the set of cores does not have this property since two cores may be neighbors. Cluster-based broadcast and core broadcast are similar in using forward node set to disseminate a broadcast packet in the network.

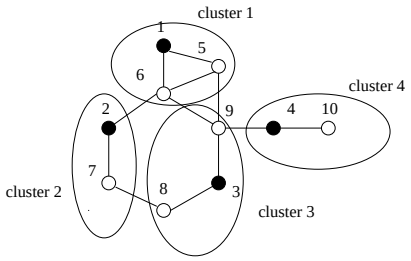

Fig. 2. A sample clustered network.

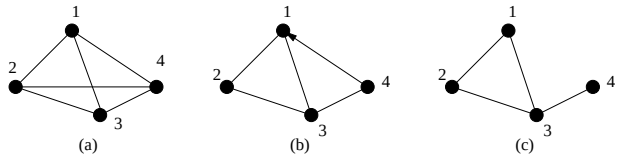

Fig. 3. The cluster graph and neighbor cluster graph of the sample network in Figure 2: (a) the cluster graph based on the 3-hop coverage method, (b) the cluster graph based on the 2.5-hop coverage method, and (c) the neighbor cluster graph.

III. Efficient Broadcast with Forward Node Set

A. Cluster Formation and Maintenance

For a clustered network, the network is partitioned into clusters with one clusterhead per cluster. Clusterheads are elected through an election process. A clusterhead directly connects to all the nodes in the cluster. Other members in the cluster are non-clusterhead nodes. In our protocol, the clustered network is formed by the lowest-ID cluster algorithm. Figure 2, shows the result of applying the algorithm in a network with 10 nodes.

The pure clustered network does not support locality of maintenance; however, it can be made localized if a slightly different cluster construction strategy which is proposed in [1] is applied: Once a cluster is formed, a non-clusterhead node never challenges the current clusterhead. If a clusterhead moves into an existing cluster, the clusterhead that has the higher ID gives up its role of clusterhead. If a clusterhead moves out of a cluster, the remaining non-clusterhead nodes in this cluster determine their new clusters. A node that has clusterhead neighbors will take the adjacent clusterhead that has the lowest ID as its new clusterhead and joins in that cluster. For those nodes that have no clusterhead neighbors, the cluster formation process is applied among those nodes to form new clusters. Thus, the clusters can be mobility adaptive and the changes in a cluster can be limited in a restricted area.

B. Adjacent Clusterhead Information Gathering

There are two ways to define a clusterhead v's *adjacent clusterhead set* $C(v)$:

1. *3-hop coverage*: $C(v)$ includes all clusterheads in $N^3(v)$.
2. *2.5-hop coverage*: $C(v)$ includes all clusterheads in $N^2(v)$ and the clusterheads that have non-clusterhead members in $N^2(v)$.

Here, we use $N^i(v)$ to represent the i^{th}*-hop neighbor set* of nodes whose distances from v are within i hops. A node is called a *1-hop (2-hop) gateway* if it is used to connect an adjacent clusterhead that is 2 hops (3 hops) away. Sometimes a node may be a 1-hop gateway and a 2-hop gateway at the same time. From the neighbor set information periodically sent by 1-hop and 2-hop gateways, a clusterhead v can compute its adjacent clusterhead set $C(v)$. The clusterheads in $C(v)$ are grouped into two classes: (a) the clusterheads that are 2 hops away from v, represented by $C^2(v)$ and (b) the clusterheads that are 3 hops away from v, represented by $C^3(v)$. Therefore, $C(v) = C^2(v) \cup C^3(v)$. Notice that $C^2(v)$ is the same for both methods, but not for $C^3(v)$. For the sample network in the Figure 2, based on the 3-hop coverage method, $C(1) = \{2, 3, 4\}$, $C^2(1) = \{2\}$, and $C^3(1) = \{3, 4\}$; while based on the 2.5-hop coverage method, $C(1) = \{2, 3\}$, $C^2(1) = \{2\}$, and $C^3(1) = \{3\}$. For clusterhead 3, $C(3) = \{1, 2, 4\}$ for both methods where $C^2(3) = \{4\}$ and $C^3(3) = \{1, 2\}$.

To gather neighbor cluster information, each node exchanges information with its neighbors. Each node needs only two sends after the formation of the cluster. Under the 2.5-hop coverage method, since each 2-hop neighbor belongs to only one cluster and it knows its clusterhead after the cluster formation process, these two sends are equivalent to the 2-hop neighbor set information gathering process in terms of the size of each packet (one with $O(\Delta)$ and the other with $O(\Delta^2)$, where Δ is the maximum node degree of the network). Under the 3-hop coverage method, each 2-hop neighbor may have multiple adjacent clusterheads that belong to different clusters (as 2-hop neighbor u of v in Figure 1). In addition, the maintenance of 3-hop neighbor set information costs more than that of 2.5-hop neighbor set information.

A *cluster graph* G' is constructed from clusterheads in the given clustered network G: Each vertex of G' comes from a clusterhead in G, and each directed link (v, u) of G' is from clusterhead v to a clusterhead u ($u \in C(v)$). For the 3-hop coverage method, all the clusterheads within $N^3(v)$ will be included in $C(v)$. Therefore, the corresponding cluster graph G' will be an undirected graph. That is, for any two clusterheads v and u, if $u \in C(v)$, then $v \in C(u)$. Both directed links (v, u) and (u, v) exist in graph G'. But with the 2.5-hop coverage method, the clusterhead v builds its adjacent clusterheads set $C(v)$ that consists of clusterheads in $N^2(v)$ and clusterheads of those non-clusterhead members that are in $N^2(v)$. In this case, there may exist two clusterheads v and u, where $u \in C(v)$, but $v \notin C(u)$. For the sample network in Figure 2, the cluster graphs under both methods are shown in Figures 3 (a) and (b). For both cases, the following theorem is true for the cluster graph G'.

Theorem 1: The cluster graph G' built from a connected undirected graph G is a strongly connected graph.

Proof: A graph is a strongly connected graph if and only if there exists a path for any two vertices in the graph. Construct a *neighbor cluster graph* G'' from the given clustered network G: Each cluster in G is considered as a vertex in G'', if two clusters $c(u)$ and $c(v)$ in G are neighbors (i.e., a directed link exists in G that connects a node in cluster $c(u)$ and a node in cluster $c(v)$), a link exists between the corresponding vertices in G''. Because G is a connected undirected graph, for any two vertices $c(u)$ and $c(v)$ in G'', there exists a path $P = (u, u_1, u_2, ..., v)$ in G. The path traverses the clusters $(c(u), c(u_1), c(u_2), ..., c(v))$ in G. It means that there exists a path in G'' that connects $c(u)$ and $c(v)$. Therefore, G'' is a strongly connected graph.

It can be easily seen that neighbor cluster graph G'' is a subgraph of cluster graph G' based on either the 3-hop coverage method or the 2.5-hop coverage method. That is, the cluster graph G' contains the neighbor cluster graph G''. Therefore, adding some links to the neighbor cluster graph G'' can build the cluster graph G' without changing its strongly connected property. Since G'' is strongly connected, G' is also strongly connected. ■

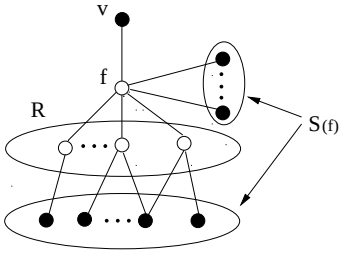

Fig. 4. A forward node $< f, R >$ of clusterhead v.

C. Forward Node Set Selection and Reduction

Clusterhead v's forward node set is a subset of gateways by which v can connect to the clusterheads in $C(v)$. v connects to a clusterhead in $C^2(v)$ via a 1-hop gateway and it connects to a clusterhead in $C^3(v)$ with two 2-hop gateways. v's forward node set is computed on-demand to connect to all the clusterheads in $C(v)$. Notice that since the 3-hop coverage and the 2.5-hop coverage methods generate different $C(v)$s, the corresponding forward node sets are also different.

We use a greedy algorithm to determine the forward node set at each clusterhead. When a clusterhead v receives a packet from another clusterhead u via u's forward node set $F(u)$, v selects a minimum number of gateways to form its forward node set $F(v)$, through which v can connect to all the clusterheads in $C(v)$. The forward node set is organized as $\{< f_i, R_i >\}$, where f_i is a 1-hop gateway used to connect to the clusterheads in $C^2(v)$ and R_i is a set of 2-hop gateways that are neighbors of f_i and are used to connect to the clusterheads in $C^3(v)$ (see Figure 4). R_i may be empty if none in $C^3(v)$ is considered.

Selection process:
Let $F(v) = \phi$, $U(v) = C(v)$, $K = N^1(v)$, and $S(f_i) = (N^1(f_i) \cap C^2(v)) \cup (N^2(f_i) \cap C^3(v))$ where $f_i \in N^1(v)$.
While $(U(v) \neq \phi)$
(a) Find f_i such that $S(f_i)$ is the maximum in $N^1(v)$. A tie is broken by selecting the node with the lower ID.
(b) Let $R_i = \phi$, $V(f_i) = N^2(f_i) \cap C^3(v)$.
While $(V(f_i) \neq \phi)$
i. Find r_j that covers the maximum number of $V(f_i)$ where $r_j \in N^1(f_i)$. A tie is broken by selecting the node with the lower ID.
ii. $R_i = R_i \cup \{r_j\}$, $V(f_i) = V(f_i) - N^1(r_j)$.
(c) $F(v) = F(v) \cup \{< f_i, R_i >\}$, $U(v) = U(v) - S(f_i)$, $K = K - \{f_i\}$, and $S(f_j) = S(f_j) - S(f_i)$ for all $f_j \in K$.

At the beginning, all the clusterheads in $C(v)$ are *uncovered*. When a forward node $< f, R >$ is selected, some clusterheads in $C^2(v)$ and/or $C^3(v)$ are covered. f is selected if it can cover the maximum number of uncovered clusterheads in $C(v)$ directly or indirectly via its neighbors. Notice that some clusterheads in $C^2(v)$ are directly covered by f and some clusterheads in $C^3(v)$ are indirectly covered by f via nodes in $N^2(v)$, they all count to the size of clusterheads in $C(v)$ covered by f. After f is determined, R can also be determined in a similar way: The neighbor of f which covers the maximum number of the uncovered clusterheads in $C^3(v)$ will be first selected into R until all the clusterheads in $C^3(v)$ that are indirectly covered by f are selected. When $< f, R >$ is included in $F(v)$, all the clusterheads that are covered by $< f, R >$ in $C(v)$ change their states to *covered*. The selection process repeats until all the clusterheads in $C(v)$ are covered.

Specifically, assume $F(v) = \{< f_1, R_1 >, < f_2, R_2 >, ..., < f_m, R_m >\}$ is v's forward node set, $U(v)$ is the subset of $C(v)$ that is uncovered so far. At the i^{th} iteration, the forward node f_i is selected from $N^1(v)$. $S(f_i)$ is the subset of the clusterheads in $U(v)$ that is covered by f_i and it consists of two parts: clusterheads in $C^2(v)$ that f_i covers directly and clusterheads in $C^3(v)$ that f_i covers indirectly via nodes in R_i (see Figure 4). The f_i with the maximum size of $S(f_i)$ will be first selected. Set $R_i = \{r_j | r_j \in N^1(f_i) \cap N^2(v)\}$ where r_j is a 2-hop gateway that can cover the clusterheads in $C^3(v)$. The selection of r_j follows the same rule as the one applied for f_i.

When a clusterhead u uses its forward node set to deliver a packet to all the clusterheads in $C(u)$, it piggybacks $C(u)$ and u with the broadcast packet so that these information can be used by the receiver to prune more clusterheads from the coverage. Consider two adjacent clusterheads u and v. Suppose a broadcast packet sent by u, piggybacked with $C(u)$ and u, reaches v with the help of forward nodes relaying. Notice that all the neighbors of u's forward node set will receive the packet when u's forward nodes relay the packet. Since all the clusterheads in $C(u) \cup \{u\}$ are covered by u's forward node set, they do not need to be covered again when v computes its forward node set. There are some differences in computing $U(v)$ between the 3-hop coverage method and the 2.5-hop coverage method. For the 3-hop coverage method, all clusterheads that are covered by u's forward node set are also included in $C(u) \cup \{u\}$; therefore, v will determine its forward node set $F(v)$ from $N^1(v)$ to cover any nodes in $U(v) = C(v) - C(u) - \{u\}$. But for the 2.5-hop coverage method, in some case u's forward node set may cover some extra clusterheads besides $C(u) \cup \{u\}$. More specifically, if v is 2 hop away from u and u uses a path (u, f, r, v) to deliver the packet to v, clusterheads in $N^1(r)$ that are not in $C(u)$ also receive the broadcast packet. (See Figure 2, node 4 is covered by node 9 but not in $C(1)$ based on the 2.5-hop coverage method.) These clusterheads can also be excluded from $U(v)$. Therefore, $U(v) = C(v) - C(u) - \{u\} - N^1(r)$.

D. Broadcast Process and Its Termination

When a source node initiates a broadcast process, it follows the steps below:
1. If the source is not a clusterhead, it just sends the broadcast packet to its clusterhead.
2. When a clusterhead receives the broadcast packet for the first time, it chooses its forward node set to forward the packet to all its adjacent clusterheads. These adjacent clusterheads should exclude the forwarding clusterhead itself and those adjacent clusterheads of the forwarding clusterhead. The adjacent clusterheads of this forwarding clusterhead are piggybacked with the broadcast packet as well as the forward node set for the forwarding purpose. If the received packet is a duplicated one, the clusterhead does nothing.
3. When a non-clusterhead node receives the broadcast packet for the first time, and if it is in the forward node set, it relays the packet; otherwise, it does nothing.

Using such an approach, all the clusterheads in the network will eventually receive the broadcast packet provided that the network is connected. After all the clusterheads re-broadcast the packet in their clusters, all the nodes in the entire network will receive the packet.

Theorem 2: The proposed protocol successfully delivers a packet to all of the nodes in a given connected network and the broadcast process terminates in finite time.

Proof: The set of clusterheads forms a DS of the network. That is, a node in the network is either a clusterhead or a neigh-

bor of a clusterhead. Based on the protocol, each clusterhead selects a set of forward nodes to connect all its neighbor clusterheads (in a 3-hop coverage area or a 2.5-hop coverage area). In either case, all its neighbor clusters are connected. Therefore, the broadcast packet reaches all the clusterheads in a finite number of steps. Since clusterheads form a DS, one extra step of forwarding covers all the nodes in the networks. ■

E. Illustration

We use the sample network in Figure 2 to show how the proposed protocol works and the difference between our protocol and others. Suppose node 1 is the broadcast source, running the selection process with the 3-hop coverage method, $U(1) = C(1) = C^2(1) \cup C^3(1) = \{2\} \cup \{3,4\} = \{2,3,4\}$, $S(5) = (N^1(5) \cap C^2(1)) \cup (N^2(5) \cap C^3(1)) = \{3,4\}$, $S(6) = (N^1(6) \cap C^2(1)) \cup (N^2(6) \cap C^3(1)) = \{2,3,4\}$, so node 6 is selected. $V(6) = \{3,4\}$ and node 9 is selected. Therefore, the forward node set of node 1 is $F(1) = \{< 6, \{9\} >\}$. $C(1) = \{2,3,4\}$ and node 1 are piggybacked with the packet. Once nodes 2, 3 and 4 receive the packet, they will compute their corresponding forward node sets. $U(2) = C(2) - C(1) - \{1\} = \phi$, $F(2)$ is empty. Similarly, $U(3) = U(4) = \phi$ and $F(3)$ and $F(4)$ are both empty. So, nodes 2, 3 and 4 just broadcast the packet among its neighbors. Therefore, nodes $1, 2, 3, 4, 6$ and 9 forward the packet.

With the 2.5-hop coverage method, $U(1) = C(1) = C^2(1) \cup C^3(1) = \{2\} \cup \{3\} = \{2,3\}$, $S(5) = \{3\}$, $S(6) = \{2,3\}$, so node 6 is selected. $V(6) = \{3\}$ and node 9 is selected. Therefore, $F(1) = \{< 6, \{9\} >\}$; $C(1) = \{2,3\}$ and node 1 are piggybacked with the packet. For nodes 2 and 3, $U(2) = C(2) - C(1) - \{1\} = \phi$, $U(3) = C(3) - C(1) - \{1\} - N^1(9) = \{1,2,4\} - \{2,3\} - \{1\} - \{3,4\} = \phi$, $F(2)$ and $F(3)$ are both empty. $U(4) = \phi$ and $F(4)$ is empty. Therefore, nodes $1, 2, 3, 4, 6$ and 9 forward the packet.

For the core broadcast protocol [10], the cores are determined in a distributed way. One possible set of cores is $\{3, 4, 6, 7, 9\}$. Each core computes its forward node set independently. Nodes 2 and 8 will be included in the forward node sets of nodes 3 and 7, respectively. While node 1 is the source, nodes $1, 2, 3, 4, 6, 7, 8$ and 9 need to forward the packet.

For the partial dominant pruning algorithm (PDP) in [6], when node 1 is the source, its forward node set is $F(1) = \{6\}$, then, $F(6) = \{2, 9\}$, $F(2) = \{7\}$, $F(9) = \{3, 4\}$, $F(7) = \{8\}$, $F(3) = \{8\}$, $F(4) = \phi$. Therefore, nodes $1, 2, 3, 4, 6, 7, 8$ and 9 forward the packet.

Based on the marking process proposed in [12], the CDS is $\{2, 3, 4, 5, 6, 7, 8, 9\}$. With Rule 1, node 5 is extracted from the CDS. The final CDS with Rule 1 and 2 is $\{2, 3, 4, 6, 7, 8, 9\}$. The source and nodes in the CDS, that is, nodes $1, 2, 3, 4, 6, 7, 8$ and 9, will forward the packet.

For the approximation algorithm of the MCDS in [2], the MCDS of the network is {2,3,4,6,9}. Therefore, when node 1 is the source, nodes $1, 2, 3, 4, 6$ and 9 forward the packet.

We can see that the broadcast redundancy is reduced more when using our protocol than others.

IV. Performance Simulation

In this section, we measure the average number of forward nodes for a packet to reach all the nodes in a randomly generated network. The simulation runs under the following simulation environment: The area of working space is 100×100. Nodes are randomly placed in this confined area. With the predefined fixed transmission range, two nodes have a connection link if their distance is less than the node's transmission range. The links are bi-directional links. There are no other traffic except the one generated from the broadcast packets and the one used to maintain cluster structures. No transmission errors (such as contention and collision) are considered here. It is assumed that all these issues are taken care of at the MAC layer. The movement range of nodes should be relatively small during the broadcast period compared with the node's transmission range. Otherwise, it is not cost-effective to maintain cluster structures since the clusterheads cannot set up the fresh neighbor set in a timely manner. The node's transmission range is 25 and the number of nodes in the graph ranges from 20 to 100. We generate 1000 random graphs to get the average number of the forward node set. If the graph is not connected, it is just discarded. The source node for each network is also randomly selected.

The performance of our proposed protocol (EBFNS) is compared with other approaches mentioned in this paper: the core broadcast protocol (CB) in [10], the dominating set based broadcast protocol [11] by using the marking process with Rules 1 and 2 (MPR1&2) in [12], and the partial dominant pruning algorithm (PDP) proposed in [6]. The clusters are constructed with the lowest-ID (LID) and highest node degree (HD) cluster algorithms. Both 3-hop coverage (referred as I) and 2.5-hop coverage (referred as II) methods are applied for a clusterhead to gather its adjacent clusterhead information. We use the result of the MCDS in [2] as the approximation of the lower bound for the broadcast problem.

Figure 5 shows the average number of the clusters in a network when node's transmission range is 25. The number of clusters does not change significantly when the size of network increases. It suggests that the cluster structure is more suitable for a dense network. The number of clusters built by the LID algorithm is slightly larger than that built by the HD algorithm.

Figure 6 shows the result when the node's transmission range is 25. When the number of nodes is small (between 20 and 30), the EBFNS has 10-20% more forward nodes than any other algorithms (PDP, CB, and MPR1&2). While the number of nodes increases, the curves of the PDP and the CB rise significantly, but the slopes of the EBFNS and the MPR1&2 stay relatively flat. When the number of nodes is 100, the number of forward nodes of the EBFNS is only 60% of that of the PDP and the CB. This is because the number of clusters in the confined area is rather insensitive to the number of nodes in this area. So the size of forward nodes does not increase much when the size of network increases. The CB protocol does not work well when the network is large because it requires each node determine its forward node set independently. More redundant nodes will be included in the forward node set. The PDP algorithm also has a large number of forward node set when the network is large, even though it piggybacks the forward node set information in the broadcast packet. The EBFNS (LID-I) is slightly better than the EBFNS (LID-II) because more clusters can be included in $C(v)$ by using the 3-hop coverage method so that more nodes can be pruned.

In Figure 7, the difference between clusters that constructed by the LID and the HD algorithms are compared. When the transmission range is 25, the HD has a better performance than the LID since each cluster can include more nodes if the node with the highest node degree is selected as a clusterhead, and hence, the total number of clusters is smaller on average.

Figure 8 shows the effect of using the pruning technique to extract more forward nodes in the EBFNS protocol when the node's transmission range is 25. We can see that the number of forward nodes, when using the pruning technique, is less than the one without using it, especially for the one based on the

2.5-hop coverage method. One interest thing is that without pruning, the performance of the EBFNS based on the 3-hop coverage method is much better than that based on the 2.5-hop coverage method. But when using the pruning technique, they have almost the same number of forward nodes. Since the 3-hop coverage method is more costly for gathering the neighboring information, the 2.5-hop coverage method is a better choice.

From the simulations, we have the following observations:

- The number of clusters is relatively stable while the size of the network increases.
- The way that each cluster is constructed affects the size of the forward node set. A cluster formed by the HD algorithm has a smaller forward node set than the one formed by the LID algorithm, but the difference is insignificant.
- The EBFNS protocol has good performance when the node's transmission range is small and the number of the nodes is large (i.e., the diameter of the network is relatively large).
- The EBFNS protocol is relatively insensitive to the size of the network.
- The pruning technique can greatly reduce the total number of forward nodes compared with the one without using it in the clustered network.
- The 2.5-hop coverage method is almost as efficient as the 3-hop coverage method when the pruning technique is used.

V. Conclusions

We have described a protocol, called EBFNS, for efficiently broadcasting a packet in a clustered MANET by using the forward node set to relay the broadcast packet. The broadcast operation is limited in clusterheads and the nodes in the forward node set. We have also proposed a 2.5-hop coverage method for the broadcasting. The clusterheads utilize the attached adjacent clusterhead information to further reduce the forward node set. This approach makes the broadcast more efficient. The empirical results of the EBFNS show that the total number of the forwarding nodes is relatively stable for different sizes of the network and it outperforms the core broadcast protocol and the partial dominant pruning algorithm when the diameter of the network is relatively large.

References

[1] C. C. Chiang and M. Gerla. Routing and multicast in multihop, mobile wireless networks. *Proc. of IEEE Intl. Conf. on Universal Personal Communications*, 2:546–551, 1997.

[2] B. Das, R. Sivakumar, and V. Bharghavan. Routing in ad-hoc networks using a virtual backbone. *Proc. of ICCCN'97*, pages 1–20, Sept. 1997.

[3] A. Ephremides, J. E. Wieselthier, and D. J. Baker. A design concept for reliable mobile radio networks with frequency hopping signaling. *Proc. of the IEEE*, 75(1):56–73, 1987.

[4] M. Gerla and J. Tsai. Multicluster, mobile, multimedia radio network. *Wireless Networks*, 1:255–265, 1995.

[5] H. Lim and C. Kim. Flooding in wireless ad hoc networks. *Computer Communications Journal*, 24(3-4):353–363, 2001.

[6] W. Lou and J. Wu. On reducing broadcast redundancy in ad hoc wireless networks. *accepted to appear in IEEE Trans. on Mobile Computing*, 2002.

[7] M. V. Marathe, H. Breu, H. B. Hunt III, S. S. Ravi, and D. J. Rosenkrantz. Simple heuristics for unit disk graphs. *Networks*, 25:59–68, 1995.

[8] S. Ni, Y. Tseng, Y. Chen, and J. Sheu. The broadcast storm problem in a mobile ad hoc network. *Proc. of MOBICOM'99*, pages 151–162, Aug. 1999.

[9] A. Qayyum, L. Viennot, and A. Laouiti. Multipoint relaying for flooding broadcast message in mobile wireless networks. *Proc. of HICSS-35*, Jan. 2002.

[10] P. Sinha, R. Sivakumar, and V. Bharghavan. Enhancing ad hoc routing with dynamic virtual infrastructures. *Proc. of INFOCOM'2001*, 3:1763–1772, 2001.

[11] I. Stojmenovic, S. Seddigh, and J. Zunic. Dominating sets and neighbor elimination based broadcasting algorithms in wireless networks. *IEEE Trans. on Parallel and Distributed Systems*, 13(1):14–25, Jan. 2002.

[12] J. Wu and H. Li. On calculating connected dominating sets for efficient routing in ad hoc wireless networks. *Proc. of ACM DIALM'99*, pages 7–14, Aug. 1999.

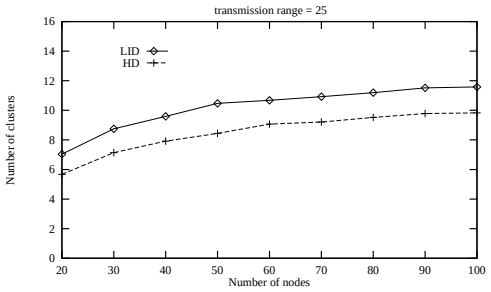

Fig. 5. Average number of clusters in a network.

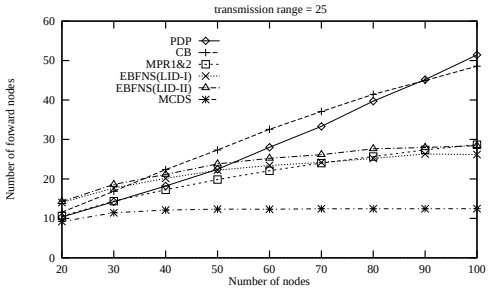

Fig. 6. Average number of the forward nodes when the range is 25.

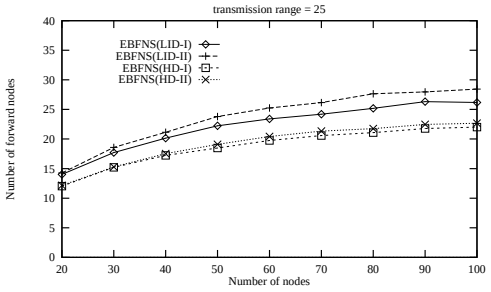

Fig. 7. The difference between clusters constructed by the LID algorithm and by the HD algorithm when the range is 25.

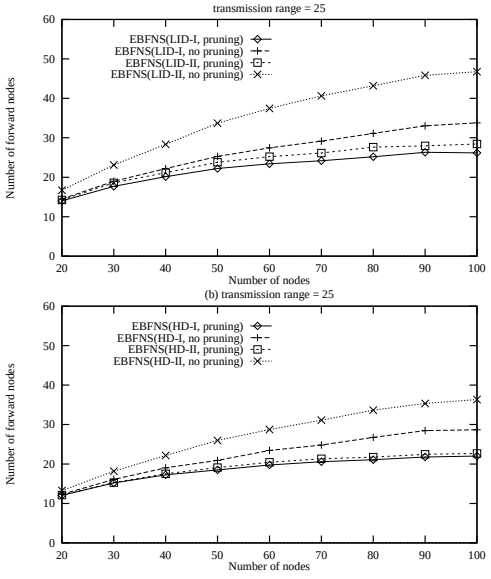

Fig. 8. The number of forward nodes with using pruning technique vs. without using pruning technique when the range is 25: (a) the clusters are formed by the LID algorithm and (b) the clusters are formed by the HD algorithm.