On the Construction of Virtual Multicast Backbone for Wireless Ad Hoc Networks

WU Ya-feng, XU Yin-long, CHEN Guo-liang, WANG Kun
Department of Computer Science & Technology
University of Science & Technology of China
National High Performance Computing Center at Hefei
yfwu@ustc.edu, {ylxu, glchen}@ustc.edu.cn, kwang@mail.ustc.edu.cn

***Abstract*—With the proliferation of portable computing devices and ascending popularity of group-oriented computing, wireless ad hoc network multicasting remains a challenging research subject. While Virtual Multicast Backbone (VMB) structure are commonly used in current multicast protocols, this paper focuses on the construction of the optimal VMB with the fewest forwarding nodes to decrease overhead and cost, due to the scarce resource in ad hoc networks. Instead of conventional Steiner tree model, the optimal shared VMB in ad hoc networks is modeled as Minimum Steiner Dominating Set (MSCDS) in Unit-Disk Graphs (UDG), which is NP-hard. A performance evaluation of flooding for MSCDS is given and a One-hop Algorithm is proposed with an approximation ratio of at most 10. To adapt various network scenarios, this paper further presents a fully distributed d-hop algorithm also with a constant approximation ratio, which organizes multicast nodes to form a hierarchical VMB. Based on the hierarchical structure, this paper proposes some approaches to maintain and update VMB, and gives a security framework to exclude malicious nodes from multicast groups. Simulation results show that the proposed algorithms perform very well.**

***Keyword*—wireless ad hoc network, virtual multicast backbone, minimum Steiner connected dominating set, unit-disk graph.**

I. INTRODUCTION

Wireless ad hoc network is comprised of mobile nodes that are communicating via either direct wireless links or multi-hop wireless links through a sequence of intermediate nodes. They are autonomously formed without any pre-configured infrastructure or centralized control. A typical application of ad hoc networks is group-oriented communication, where multicast service is supported for collaboration among group members. Therefore, multicast support is one of most desirable characteristics of ad hoc networks.

Most existing multicast routing protocols [1, 2, 3, 4] for ad hoc network establish some form of connected distribution structures often called as *virtual multicast infrastructure* or *Virtual Multicast Backbone* (*VMB*). VMB spans all multicast members and contains all *forwarding* nodes that are responsible for forwarding multicast packets. Multicast communication and group membership management functions are performed only on VMB. Based on the structure of VMB that they use, Current multicast protocols can be mainly classified into two categories, tree based and mesh based. For example, ODMRP [4] builds a forwarding mesh as a VMB for each multicast group, which consists of forwarding nodes on shortest paths between any member pairs. AMRIS [1] constructs a shared multicast delivery tree to support multicast operation.

While ad hoc network is characterized by scarce resource (bandwidth, energy, etc), VMB should be constructed as small as possible to decrease network overhead, energy and bandwidth consumption. In wired networks, the optimal multicast backbone is modeled as a well-known *Minimum Steiner Tree* (*MStT*) in graphs. Some multicast protocols also establish a Steiner tree as an efficient VMB for ad hoc network. For instance, Chen and Nahrstedt [3] constructed a Location-Guided Steiner tree to minimize overall cost of the tree. In this paper, our objective is to construct the optimal shared VMB with the fewest forwarding nodes. We will use *Unit-Disk Graph* (*UDG*) [5] to model ad hoc network and show that *Steiner Connected Dominating Set* (*SCDS*) is more reasonable to model VMB in it. So to minimize the number of forwarding nodes for multicast group in ad hoc network is to compute the corresponding *Minimum Steiner Dominating Set* (*MSCDS*) in UDG.

The MSCDS problem is proposed by Guha and Khuller [6], as the generalization of the well-known *Minimum Connected Dominating Set* (*MCDS*) problem. To find an MSCDS is NP-hard in general graphs, even in UDG. A

0-7803-8815-1/04/$20.00 '2004 IEEE

centralized approximation algorithm is given for general graphs in [6], which has an approximation ratio of $O(H(\Delta))$, where H is the harmonic function and Δ is the maximum degree of the graph. However, few distributed MSCDS algorithms are designed for ad hoc networks. While distributed *flooding* is a common strategy to construct VMB, this paper prove that the approximation ratio of flooding for MSCDS is $m-1$, where m is the number of multicast nodes. Taking advantage of MIS and MStT, this paper will present a one-hop MSCDS approximation algorithm for UDG with a constant approximation ratio 10, which outperforms flooding. The One-hop Algorithm can be implemented in a fully distributed manner with the message and time complexity of $O(nlog(n))$ and $O(Dn)$ respectively, where D is the diameter of the graph and n is the number of nodes of networks.

To adapt various scenarios of ad hoc networks, the One-hop Algorithm is extended to *d*-hop based on *maximal d-hop independent set* (*d-MIS*), which is also with a constant approximation ratio. A fully distributed implementation of the *d*-hop Algorithm with analysis of its message and time complexity is presented in detail. The implementation relies on only one hop information.

The *d*-hop Algorithm organizes multicast nodes to form *d*-hop clusters and constructs a hierarchical VMB. We give some approaches to maintain and update the VMB to deal with mobility of hosts and join and leave of multicast hosts, and present a security framework to exclude malicious nodes from multicast groups. Taking the property of the hierarchical structure, all these function can be performed in a localized and lightweight manner.

Experimental results show that the proposed algorithms outperform flooding, especially in dense networks. The performance of *d*-hop Algorithms with different values of *d* is also studied.

The rest of the paper is organized as follows. Section 2 describes the network model and gives some basic concepts and notations. In section 3, we show that MSCDS is more suitable to model the VMB in ad hoc network and investigate the performance of flooding. The One-hop Algorithm for MSCDS problem with an approximation ratio of 10 is presented in section 4, followed by its extension to a d-hop algorithm and a detailed distributed implementation in section 5. And section 6 proposes some approaches to maintain and update the VMB, and also a security framework to exclude malicious nodes from multicast groups. Simulation results are shown in Section 7. Finally, Section 8 concludes this paper.

II. PRELIMINARIES

In this paper, we consider connected wireless ad hoc networks whose nodes are distributed in a two-dimensional area and with the same transmission radius of one unit. Each node is assigned a unique identifier (*ID*). This type of ad hoc networks can be modeled as a connected *unit-disk graph* (*UDG*) $G=(V, E)$, where V contains all hosts and E is the set of links. A link between hosts u and v exists if and only if their distance is at most one unit. A multicast group, denoted by V_m, is a subset of V. Each node in V_m is called a multicast node and is assigned a unique multicast identifier (*MID*).

The notations in this paper are defined as follows:

n: the number of nodes in the network, $n = |V|$.

m: the number of multicast nodes, $m = |V_m|$.

Δ: the maximum degree of graph.

V_m: the set of multicast nodes.

For a graph $G=(V, E)$ and a subset V' of V, $G(V')$ denotes the subgraph of G induced by V'.

Given a graph $G=(V, E)$, a subset V' of V is called a *connected set* if the subgraph $G(V')$ is connected. Given a subset V' of V, a *Steiner tree* of V' is a subtree of G which contains all nodes of V'. A *Steiner dominating set S* of V' is a subset of V such that each vertex of V' is in S, or has a neighbor in S, and furthermore is called a *Steiner Connected Dominating Set* (*SCDS*) of V' if it is connected. Among all SCDSs of V', the one with minimum cardinality is called *Minimum Steiner Connected Dominating Set* (*MSCDS*) of V'.

We also extend the well-known terms of *independent* and *dominating* as the following. For any $u, v \in V$, $hop\_count(u, v)$ is the number of edges (hops) in the shortest path from u to v. Two vertices u and v are *d-neighbors*, if $hop\_count(u, v) \leq d$. For vertex v, $N_d[v]=\{u | hop\_count(u,v) \leq d\}$ is the set of *d-neighborhoods* of v. Two vertices u and v are *d-independent* if they are not *d-neighbors*. A *d-independent set* S_d of G is a subset of V such that any pair of vertices in S_d is *d-hop* independent. S_d is *maximal* if any vertex u not in S_d has a d-neighbor in S_d. A *d-dominating set* D_d of G is a subset of V such that each node not in D_d has at least one d-neighbor in D_d. A maximal d-independent set is also a d-dominating set.

In this paper, *independent* and *neighbor* represent 1-*independent* and 1-*neighbor*, for short.

III. MSCDS AND FLOODING HEURISTIC

In this section, we will show that MSCDS is more accurate to model VMB in ad hoc networks first, and then investigate the performance of flooding to approximate MSCDS.

A. MSCDS Model

While MStT is often used to approximate the optimal multicast backbone in wired networks as well as in some ad hoc network protocols, it is not an applicable model for the problem stated in this paper, for two reasons. First, the objective of minimal Steiner tree is to minimize the total weight of all the links. But our goal is to minimize the number of the forwarding nodes. Second, the Steiner tree contains all multicast nodes and all forwarding nodes. But some multicast nodes, the leaves of the Steiner tree, are only receivers without forwarding packets during multicasting in ad hoc network and contribute none to the number of forwarding nodes. So to minimize the cardinality of the Steiner tree may not minimize the number of forwarding nodes. On the other hand, SCDS connects all multicast nodes and only consists of forwarding nodes. So we use the MSCDS to model the optimal VMB in ad hoc networks. As illustrated in Fig.1, the MSCDS, rather than MStT, induces a VMB consisting of the fewest forwarding nodes.

Based on the same consideration, some broadcast protocols [7, 8, 9, 10] use the MCDS, no the *minimum spanning tree*, to approximate the *virtual backbone* for broadcasting. Alzoubi and Wan also presented several localized methods to construct theoretically good backbone in [12, 16]

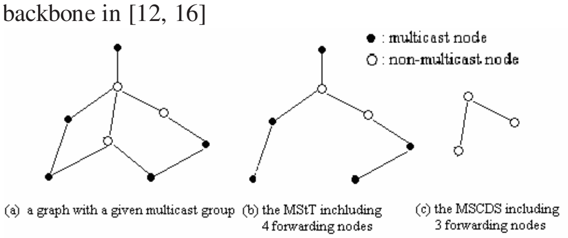

Fig. 1. An example to illustrate the difference between MSCDS and MStT.

B. Flooding Heuristics

The MSCDS problem is first proposed by Guha and Khuller, as a generalization of the well-known MCDS problem which is proved NP-hard, even in UDG. So the MSCDS problem in UDG is also NP-hard. While *flooding* is usually applied to construct VMB in multicast protocols for ad hoc networks [15], we analyze its performance for MSCDS in UDG.

To construct a VMB with flooding, the source node initiates the construction by broadcasting a RREQ packet. Upon receiving RREQ at the first time, each node records the next hop information and rebroadcasts the message. Upon receiving a RREQ, a multicast node unicasts a RREP to the source along the reverse path. All nodes on the reverse paths become the forwarding nodes of the VMB. Without considering any collision, the first RREQ that a multicast node receives comes along the shortest path from the source to the multicast node. Therefore, the VMB constructed by flooding is the combination of the shortest paths from the source to every multicast destination.

The following Lemma3.1 guarantees the upper bound m–1 on the approximation ratio of flooding. Let S_f denotes the SCDS computed by flooding.

Lemma 3.1 The size of S_f is at most $(m-1)\times opt+1$, where *opt* is the size of the MSCDS.

Proof. S_f is comprised of the multicast source and all intermediate nodes on m–1 shortest paths from the source to m–1 multicast destinations. For any multicast destination u, there are some nodes in the MSCDS to connect it and the source. Clearly, the number of these nodes does not exceed *opt*. Thus, the number of intermediate nodes on the shortest path from the source to u does not exceed *opt*. Therefore, The size of S_f is at most $(m-1)\times opt+1$.

In Fig.2, the multicast group consists of the *source* node and all m–1 *des* (destination) nodes. The shortest paths from *source* to every *des* node are disjoint and with the same hops. So the solution of flooding consists of *source* and all intermediate nodes on these m–1 shortest paths, while the optimal solution is only comprised of destination u and intermediate nodes on the shortest path from *source* to u. The instance shown in Fig.2 implies that m–1 is the lower bound of the approximation ratio of flooding for MSCDS.

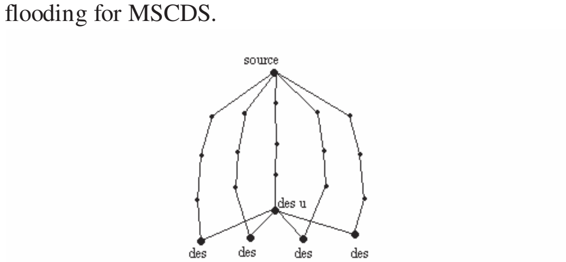

Fig. 2. An instance for which the size of the solution computed by flooding is larger than the optimal solution by a factor $m-1$

In the worst case, the multicast group involves all nodes of the network; all nodes are placed in a line and the maximum nodal degree is 2. The source node is at the end of the line. The total number of the RREQ message is exactly n, but the number of the RREP is $O(n^2)$. So the message complexity is $O(n^2)$ and the time complexity is $O(n)$.

From above, we have the following Theorem 1

Theorem 1: The approximation ratio of flooding for MSCDS is exactly m–1. Its message and time complexity are $O(n^2)$ and $O(n)$, respectively.

Overall, flooding is a fully distributed strategy and usually applied for routing discovery in many protocols for ad hoc network due to easy to be implemented. But flooding is not suitable to approximate MSCDS because (1) simply merging shortest paths, flooding involves overmany nodes in the VMB, (2) In the worst case, its total number of the control message $O(n^2)$ is a high overhead.

IV. ONE-HOP ALGORITHM

In this section, we will propose an MSCDS approximation algorithm for UDG, which is with a constant approximation ratio at most 10 and can be implemented in a fully distributed manner. The algorithm takes the advantages of the properties of MIS and MStT. It finds a small subset of multicast group to dominate all multicast nodes first, and then connects these nodes.

One-hop Algorithm

*Step*1. Find a *maximal independent set* I in $G(V_m)$.

*Step*2. In G, apply the Steiner tree algorithm in [11] to find a Steiner tree T for the subset I, with all edges having unit weight. The final solution is the set of the nodes of T.

Since I is an MIS and also a dominating set of $G(V_m)$, all multicast nodes of V_m are dominated by the set I. In step2, tree T spans all nodes of I and also dominates all nodes of V_m. Therefore, the nodes of T form a SCDS of V_m.

We study the approximation performance of the One-hop Algorithm. First, Alzoubi et. al. gave the following lemma in [12], which relates the size of any independent set of UDG to the size of any connected dominating set.

Lemma 4.1 The size of any independent set in a UDG is at most $4\times|D|+1$, where D is a connected dominating set of in the UDG.

Let S_{opt} be the MSCDS of V_m. The size of I can be bounded in terms of the size of S_{opt}, denoted by *opt*.

Lemma 4.2 The size of I is at most $4\times opt + 1$.

Proof. I is an independent set of $G(V_m \cup S_{opt})$. Since S_{opt} dominates all nodes of V_m, S_{opt} is a connected dominating set of $G(V_m \cup S_{opt})$. With Lemma 4.1, the size of I is at most $4\times opt + 1$.

With Lemma 4.2, the algorithm can be guaranteed with a constant approximation ratio of at most 10.

Theorem 2: The one-hop Algorithm computes a SCDS of multicast group V_m, whose size is at most $10\times opt + 1$.

Proof. Each node of I is a multicast node dominated by S_{opt} and S_{opt} is a connected set. So the subgraph $G(I \cup S_{opt})$ is a connected graph with at most $|I| + opt$ nodes. Let T' be a spanning tree of $G(I \cup S_{opt})$. Obviously, T' is also a Steiner tree of I in G with at most $|I| + opt-1$ edges. So the MStT of I will have also at most $|I| + opt-1$ edges.

In step2, we apply the (edge weighted) Steiner tree approximation algorithm in [11], with all edges having unit weight, and find a Steiner tree T with at most $c\times(|I| + opt-1)$ edges, where c is the Steiner approximation ratio and equal to 2 with the algorithm in [11]. So the number of nodes of T is at most

$$2\times(|I| + opt-1)+ 1 \leq 10\times opt + 1$$

The One-hop Algorithm can be implemented in a fully distributed manner. Some distributed approaches are designed to construct MIS in ad hoc networks [8, 13, 14]. In step1, the algorithm in [14] can be modified to compute an MIS among multicast group V_m. In step2, a distributed version of K-SPH algorithm [11] can be used. The message and time complexity of constructing a Steiner tree are $O(nlog(n))$ and $O(Dn)$ respectively, while both are $O(n)$ of constructing an MIS of multicast group, where D is the diameter of the graph. So the message and time complexity of One-hop Algorithm are $O(nlog(n))$ and $O(Dn)$ respectively.

In the SCDS constructed by the One-hop Algorithm, all multicast nodes are dominated by I. Each node of I and its multicast neighbors form a one-hop multicast cluster. Nodes of I are cluster-heads. So the algorithm can also be described as: some independent multicast nodes are chosen to be cluster-head; each cluster-head forms a one-hop multicast cluster with its multicast neighbors; and connect these cluster-heads with a Steiner tree. The One-hop Algorithm organizes multicast nodes to a hierarchical VMB.

V. D-HOP ALGORITHM

The One-hop Algorithm has an approximation ratio of 10 and constructs a hierarchical VMB. However, when deployed in sparse UDG, where most multicast nodes are two or more hops apart from each other, it mostly results in trivial single-node multicast clusters and consequently flat VMB. This implies that One-hop Algorithm is not fit for VMB construction in sparse ad hoc networks. To address this issue, an extended d-hop Algorithm is proposed with detailed description of distributed implementation, whose approximation ratio proves also constant.

A. Algorithm Description

The d-hop Algorithm finds a d-MIS among multicast nodes, which is also a d-hop dominating set of the multicast group, and then each node in the d-MIS becomes a cluster-head and forms a d-hop cluster with all its d-neighbors. Intra-cluster, some multicast nodes are further chosen to dominate multicast nodes of the cluster. These nodes are connected to the cluster-head with the shortest paths. Inter-cluster, a Steiner tree is used to connect all cluster-heads.

For two vertex $u, v \in V(G)$, let $P_{u,v}$ be the set of nodes on the shortest path between u and v, including u and v.

d-hop Algorithm

*Step*1. Construct a graph $G_d=(V_m, E_d)$, where V_m is the set of all multicast nodes and $E_d = \{(u, v) |$ if u is a d-neighbor of $v\}$. Find an MIS I_d in G_d.

*Step*2. For each node $u \in I_d$, let M_u be the set of all multicast nodes in d-neighborhood of u. Find an MIS S_u in $G(M_u)$. For each node $v \in S_u$, connect u and v with the nodes of $P_{u,v}$. We define $P_u = \bigcup_{v \in S_u} P_{u,v}$.

*Step*3. In G, apply the Steiner tree algorithm in [11] to find a Steiner tree T for the subset I_d, with all edges having unit weight. The final solution is the nodes of P and T, where $P = \bigcup_{u \in I_d} P_u$.

Because the MIS I_d is also a dominating set of G_d, any multicast node v not in I_d must be in d-neighborhood of some node in I_d. Suppose $v \in M_u$. Since S_u is a MIS, also a dominating set of $G(M_u)$, v is dominated by S_u and also by P. So the final solution dominates all multicast nodes and is obviously connected, which induces an SCDS of the multicast group.

To bound the size of I_d, we first give the following Lemma 5.1, which relates the size of any d-independent set S_d of UDG to the size of its connected dominating set D.

Lemma 5.1 The size of any d-independent set S_d of UDG is at most $2 \times |D| / d$ if d is even, and $(4 \times |D|+1) / (2d-1)$ if d is odd, where D is a connected dominating set.

Proof. Lemma 5.1 can be proved by induction on d. For any $u \in S_d$, there is a node v in D adjacent to u. v is called as *u's dominator* in the following.
Basic step: For $d = 1$, $|S_1| \leq 4 \times opt + 1$ with Lemma 4.1. For $d = 2$, any pair of nodes in S_2 are separated by at least three hops. Thus any node of D dominates at most one node of S_2. So $|S_2| \leq opt$. Lemma 5.1 holds.
Inductive step: Assume Lemma 5.1 is true for some $d > 0$ and consider d+2. Let d be even. For any node $u \in S_{d+2}$, we contract u and one of *u's* dominators to a new node u' (see in Fig. 3). Let the set of these new nodes be S' and clearly $|S'| = |S_{d+2}|$. Let the set of nodes left in D be D'. Since any node of D dominates at most one node of S_{d+2}, $|D'| \leq opt - |S_{d+2}|$. Clearly, S' is a d-independent set and dominated by D'. After contraction, D' may not be a connected set. Suppose $D' = \cup D'_i$, where D'_i is a connected set. Let $S' = \cup S'_i$, where S'_i is the subset of S' dominated by D'_i. In $G(S'_i \cup D'_i)$, S'_i is a d-independent set and D'_i is a connected dominating set. By the inductive hypothesis $|S'_i| \leq 2 \times |D'_i| / d$. Totally, we have $|S'| \leq 2 \times |D'| / d$. So $|S_{d+2}| \leq 2 \times (opt - |S_{d+2}|) / d$. Finally $|S_{d+2}| \leq 2 \times opt / (d+2)$.

If d is odd, $|S_{d+2}| \leq (4 \times opt+1) / (2d+3)$ can be proved in the same way,. Therefore Lemma 5.1 holds.

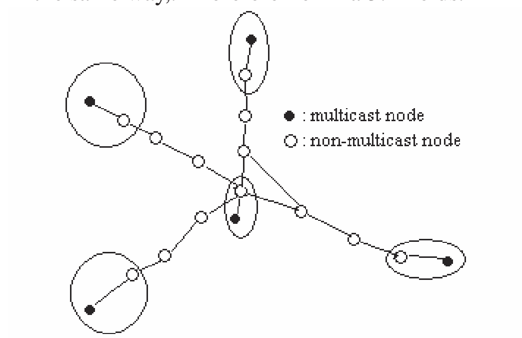

Fig. 3. An example of contraction in the inductive step

Based on Lemma 5.1, the size of the set I_d can be bounded in terms of *opt.*

Lemma 5.2 In d-hop Algorithm, the size of I_d is at most $2 \times opt / d$ if d is even, and $(4 \times opt+1) / (2d-1)$ if d is odd.

Proof. In $G(I_d \cup S_{opt})$, I_d is a d-independent set and S_{opt} is a connected dominating set. Lemma 5.2 holds from Lemma 5.1.

Next, we also bound the size of S_u for each $u \in I_d$. The following Lemma 5.3 gives a constant bound on the number nodes of each S_u.

Lemma 5.3 In step2, the size of S_u is at most $4d^2 + 4d$ for any node $u \in I_d$.

Proof. In UDG, the disk of a vertex denotes a disk of radius 0.5 centered at this vertex. And two vertices are independent if their disks are disjoint. For any node v in UDG, the disks of its d-neighbors all lie within the circle centered at v of radius d+0.5. In d-hop Algorithm, all nodes in S_u are d-neighbors of u and independent from each other. So the disks of the nodes in S_u are disjoint and all lie within the circle centered at u of radius d+0.5. Thus, the size of S_u is less than

$$\pi \times (d+0.5)^2 / \pi \times 0.5^2 = 4d^2 + 4d + 1$$

Lemma 5.3 holds.

From above lemmas, the d-hop Algorithm has a constant approximation ratio.

Theorem 3: The d-hop Algorithm computes a SCDS of multicast group V_m, whose size is at most

$$(8d^2 + 8d + 2 + 4/d)\times opt - 1,$$

if d is even, and

$$[(16d^3+16d^2+4d+6)\times opt + 4d^3 + 4d^2 - 2d+3]/ (2d - 1)$$

if d is odd, where opt is the size of the optimal solution.

Proof. For each node $u \in I_d$, we have $|S_u| \leq 4d^2+4d$ from Lemma 5.3. Since all nodes in S_u are in d-neighborhood of u, $|P_u| \leq (4d^2+4d)\times d$. So $|P| \leq (4d^2+4d)\times d\times |I_d|$.

In step3, a Steiner tree T is computed to connect nodes in I_d. The size of T is at most $2\times(|I_d| + opt-1) + 1$. Therefore the size of the final solution is at most $(4d^2+4d)\times d\times |I_d|+2\times(|I_d|+opt-1)+1$. Theorem 3 holds from Lemma 5.2.

The d-hop Algorithm organizes multicast nodes to a two-level hierarchical VMB. In the first level, each node u of I_d is a cluster-head and forms a d-hop multicast cluster with all multicast nodes of its d-neighborhood. For the second level, in a cluster, all nodes in S_u, called as *cluster-dominators*, dominate all multicast nodes of the cluster in one hop. The cluster-head connects all cluster-dominators of the cluster with the shortest paths. Inter-cluster, cluster-heads are connected with a Steiner tree.

B. Distributed Implementation

The distributed implementation of d-hop Algorithm relies only on local information. Each node only knows the IDs of its one-hop neighbors and the MIDs of its one-hop multicast neighbors. We also make the following common operational assumptions.

- A message sent by a node is received correctly by all its neighbors within a finite time (a *round*).
- Network topology does not change during the algorithm execution.

The distributed implementation executes in two phases. The first phase is the construction of the d-hop multicast clusters. In the second phase, we apply the distributed protocol in [11] to construct a Steiner tree to connect the cluster-heads.

In the first phase, we firstly compute a d-hop MIS of multicast group with a rank-based approach in [14], which computes MIS in ad hoc networks. By exchanging messages within d-neighbors, the multicast node with the lowest MID among candidate multicast nodes is chosen into d-MIS and then acts as a cluster-head. Based on the information obtained during message exchanges, each cluster-head selects some multicast nodes to be cluster-dominators, which dominate all multicast node of the cluster. The cluster-head unicasts messages to notify each cluster-dominator along the shortest path. All nodes on the shortest path become members of VMB, which are used to connect the cluster-head and cluster-dominators.

In the first phase, each multicast node is in one of the four states: *candidate, cluster-head, cluster-dominator* and *cluster-dominatee*. Each multicast node is initially in the candidate state and subsequently enters one of other three states. Each non-multicast node is one of the two states: *candidate* and *VMB*. Each non-multicast node is initially in the candidate state.

In this phase, three types of messages will be used:

- MESSAGE(*sender*, *TTL*, *Mlist*): a multicast node, whose MID is *sender,* broadcasts this message to all its d-neighbors. *TTL* initially equals to d. *Mlist* stores the MIDs of its multicast neighbors.
- CH(*ch, cd*): a cluster-head *ch* unicasts this message to a cluster-dominator *cd*. This message is used by a cluster-head to notify the selected cluster-dominators.
- CM(*flag, ch, sender, TTL*): a cluster-dominator or a cluster-dominatee, whose MID is *sender* and whose cluster-head is *ch*, broadcasts this message to all of its d-neighbors to declare its status. *TTL* initially equals to d. *flag* indicates the node is a cluster-dominator or a cluster-dominatee.

Each node maintains a list *Rlist,* which is initially empty. *Rlist* records the next hop for routes to multicast nodes in a cluster. *Rlist* has two fields *Destination MID* and *Next hop*. When a node receives a MESSAGE from a neighbor v and there is not an entry for the *sender* of the MESSAGE, it adds a corresponding entry into *Rlist.*

Each multicast node also maintains following local variables and data structures.

x: the number of current candidate multicast nodes in d-neighborhood with *lower* MIDs, initially equals to 0.

ch: the MID of its cluster-head.

Mlist: the list recording the MIDs of its multicast neighbors.

Dlist: a list that is used to select cluster-dominators. An entry in *Dlist* is an ordered pair of MIDs (u, v), where u is a d-neighboring multicast node with *larger* MIDs and v is a multicast node that is a neighbor of u and selected to dominate it.

Each cluster-dominatee node also maintains a local variable *cd* which stores the MID of its cluster-dominator.

The first phase of the distributed implementation is described as followings.

(1) Initially, each multicast node broadcasts a MESSAGE.

(2) Upon node v receiving a MESSAGE$(u, k, list_1)$ from its neighbor, v updates its *Rlist.* If $k > 0$, v re-broadcasts a MESSAGE$(u, k-1, list_1)$.

(3) Upon multicast node v receiving a MESSAGE$(u, k$ $list_1)$, v firstly performs the operation in (2). If u is lower than the MID of v, v increases x by one. Else if

there is no entry for u in *Dlist, u* is chosen to be cluster-dominator and a new entry (u, u) is added into *Dlist*, and then for every multicast node l in the $list_1$, v adds a new entry (l, u) to record that l is dominated by u, if l has not been recorded in *Dlist*.

(4) After d rounds, each multicast node checks its x. If $x = 0$, a multicast node u changes its state to cluster-head. For each cluster-dominator v recorded in *Dlist*, u unicasts a CH(u, v) to v according to the corresponding next hop in its *Rlist*.

(5) Upon a non-multicast node u receiving a CH, u changes its state to VMB and relays the CH according to its *Rlist*.

(6) Upon a multicast node v receiving a CH(u, v), v knows that it is selected as a cluster-dominator by cluster-head u. v changes its state to cluster-dominator, sets the local variable *ch* to u and initiates a CM(f, u, v, d), where f indicates that v is a cluster-dominator.

(7) Upon a node l receiving a CM(f, u, v, k), l re-broadcasts a CM$(f, u, v, k\text{-}1)$ if $k > 0$.

(8) Upon a candidate multicast node l receiving a CM(f, u, v, k), l firstly performs the operation in (7). If $k = d$ and the flag f indicates the sender v is a cluster-dominator, l knows that v is a cluster-dominator in its neighborhood. Then l changes its own state to cluster-dominatee, and sets two local variables *ch* to u and *cd* to v. l also broadcasts a new CM(f', u, l, d) to declare its status. Else if u is lower than the MID of l, l decreases x by one and then if $x = 0$, l becomes a cluster-head and starts to send CH messages.

In the second phase, we modify the distributed protocol in [11] to compute the Steiner tree which connects cluster-heads. After sending all CH messages, a cluster-head enters into the second phase and waits for 3d rounds. During the first d rounds, other nodes in the cluster change their status and complete their actions, described in the first phase. During the next 2d rounds, nodes are listening to the message. If a node gets some information of another cluster-head by receiving a CH from another cluster-head or a CM indicating a different cluster-head, the node unicasts a *connect* message to notify its own cluster-head. After 3d rounds, if the cluster-head has received some *connect* message, it chooses the closest cluster-head and begins to execute the *connection* step in [11]; otherwise it initiates the *discovery* step in [11]. During constructing the Steiner tree, a candidate non-multicast node or a cluster-dominatee changes its state to VMB if involved in the Steiner tree.

Finally, the resultant SCDS of multicast group consists all cluster-heads, cluster-dominators and VMB nodes.

Next, we analyze the message and time complexity of the distributed implementation above.

Theorem 4: The distributed implementation of d-hop Algorithm for constructing an SCDS of multicast nodes has a time complexity $O(Dn)$ and a message complexity of $O(n\log(n))$ if d equals to 1, otherwise $O(n^2)$.

Proof. In the first phase of distributed implementation, each multicast node initiates a MESSAGE or a CM once and all nodes in its d-neighborhood will rebroadcast the message. So the number of MESSAGE and CM are both $O(m\Delta^{d-1})$, at most $O(n^2)$. A CH message is sent along the shortest path from the cluster-head to the cluster-dominator. So each CH message will be relayed at most d times and the number of CH is at most $O(dm)$. In the second phase, the protocol in [11] has a message complexity of at most $O(n\log(n))$. Therefore, if d equals to 1, the message complexity is at most $O(n\log(n))$; otherwise the message complexity is at most $O(n^2)$.

In the first phase, the worst-case time complexity for the d-MIS occurs when all multicast nodes are arranged in ascending order and the maximum nodal degree is 2 and each multicast node is d-hop away from the closest multicast node. In this case each multicast node must wait for its d-hop multicast neighbor to declare its state. The time complexity is $O(n)$. The time complexity of the second phase is $O(Dn)$ [11], where D is the graph diameter. So the distributed implementation has a time complexity of $O(Dn)$.

VI. HIERARCHICAL STRUCTURE

Another advantage of the d-hop Algorithm is that the resultant VMB has a hierarchical structure. In this section, some approaches are proposed to maintain and update VMB to deal with mobility of nodes as well as *join* and *leave* of multicast hosts. We also concisely present a security framework to exclude malicious nodes from multicast groups. Taking the property of the hierarchical structure, all these function can be performed in a lightweight and localized manner.

A. Mobility Maintenance

In ad hoc network, free mobility of nodes will cause frequent topology changes. SCDS in UDG should be maintained as the network topology changes. The main idea of Maintenance is that the repair of the VMB is done in local vicinity and no global reconstruction is needed.

Cluster-heads have the key responsibility to maintain the structure. Cluster-heads periodically send two kinds of maintenance messages to ensure the connectivity of VMB. Intra-cluster, a cluster-head unicasts Hello messages to all its cluster-dominators along the paths of the VMB. Inter-cluster, cluster-heads unicast Hello messages to nearby cluster-heads along the branches of the Steiner tree.

We first consider the cases that a non-multicast node in the VMB moves away.

Case 1: When a non-multicast node that connects a cluster-head u and a cluster-dominator v moves away, without receiving the Hello message from u after some time, v knows the path failure. Then v broadcasts a REP message to recalculate a path to u. The *TTL* field of REP is set to a value a little bigger than d. When the VMB nodes that previously connect u and v receive the REP, they quit from the VMB. When another VMB node or u receives the REP, a message is sent back to v. Receiving the message, v sends a CONFIRM message to u along the new path. Nodes on the path become VMB nodes

Case 2: When a non-multicast node that connects two cluster-heads moves away, cluster-heads cannot receive Hello message from the other. Then cluster-heads broadcast a message to find a path to connect them.

We also consider the cases that a multicast node in the VMB moves away.

Case 3: When a cluster-dominatee moves away, no action should be done.

Case 4: When a cluster-dominator moves away, the corresponding cluster-dominatees know the absence of the cluster-dominator and wait for the Hello message from the cluster-head. If a cluster-dominatee receives a Hello message, it broadcasts a one-hop declaring message to declare itself as a new cluster-dominator. If a cluster-dominatee does not receive Hello message or declaring message, it initiates a REP message to find a path to the cluster-head. After receiving the confirm message, it becomes a new cluster-dominator and sends a declaring message.

Case 5: When a cluster-head moves away, the VMB nodes that are neighbors of the cluster-head know the accident and broadcast messages to notify the multicast nodes in the cluster. When a multicast node receives the message, it waits for some intervals in proportion to the distance to the cluster-head and then broadcasts a message to declare itself as cluster-head and to find the paths to all cluster-dominators. The new cluster-head also finds and reuses the previous paths to other cluster-heads.

B. Join and Leave

In some environments, nodes are free to join or leave a multicast group. The VMB should be updated in a lightweight manner when nodes join or leave the multicast group. When a multicast node leaves the multicast group, we apply the same strategy to update the VMB as the node moves away. If a node u wants to join a multicast group, it broadcasts a JOIN message. Receiving the JOIN message, a multicast node sends back a message with its own status. When u receives the first message, it decides its own status according to the status of the responding node and the distance.

C. Security Framework

In some applications, multicast membership is restricted to specific hosts in ad hoc network, while some malicious nodes should be excluded from the multicast group with a corresponding security framework. Based on the hierarchical structure, we design a security framework for membership authentication of hosts.

The security framework is based on a reasonable assumption that the set of malicious nodes is only a small portion of the multicast nodes and normal multicast nodes are more than malicious nodes in each local part of the network. The key of the security framework is that the most reliable multicast nodes are chosen as cluster-head and cluster-heads perform the authentication of multicast membership.

First, we briefly describe how to choose the most reliable multicast nodes to be the cluster-head. In the first d rounds, all multicast nodes broadcast its own MID to its d-neighbors. When a multicast node receive a MID, it decides to support this node or deny it and then broadcasts a message to show its decision in its $2d$-neighborhood. This ensures that each d-neighbors of a node u can receive all evaluation about u. Then the multicast node supported by the most nodes is chosen as cluster-head.

Each cluster-head monitors the behavior of multicast nodes in its cluster and can decide, or active a vote within the cluster to decide, whether to dismiss the multicast member with strange behaviors.

When a node wants to join the multicast group, it broadcasts a JOIN message. Only a cluster-head can respond to the JOIN message and decide to accept or not.

VII. SIMULATION

In this section, the simulation experiments are performed to obtain the metric, namely, the number of forwarding nodes computed by the One-hop Algorithm, d-hop Algorithm and flooding in different scenarios of ad hoc networks, which serves subsequent performance evaluation of respective algorithms.

Random graphs are generated in 1000×1000 square units of a 2-D simulation area, by randomly throwing a certain number of hosts. Three typical scenarios are selected. (1) Transmission range R is 250 units and the number of hosts n is 50. (2) R is 200 units and n is 80. (3) R is 100 units and n is 150. For each scenario, the number of multicast hosts *m* varies from 0 to *n*. For each *m*, generate a random connected graph 200 times. Calculate the 1-hop, 2-hop, 3-hop algorithm and flooding for each case; and at the end, we take the average sizes of SCDSs computed by these algorithms as the average numbers of forwarding nodes. The results are reported in Fig. 4, 5 and 6 for the three scenarios respectively. It can be shown that our algorithms compute less forwarding nodes than flooding in all of scenarios. In general, the gap between our algorithms and flooding increases as *m* increases. This indicates that our algorithms are more suitable to compute VMB in the networks where multicast nodes are distributed densely.

It can be also shown that 1-hop algorithm is better than 2-hop and 3-hop ones in most cases, especially when most nodes of networks are multicast nodes and the transmission range is large (the first scenario). On the other hand, 2-hop and 3-hop algorithms outperform 1-hop algorithm where few multicast nodes exist and the transmission range is small (the third scenario). This indicates that when multicast nodes are distributed sparsely, a *d*-hop Algorithm with a bigger *d* is more effective.

VIII. CONCLUSION

The main contribution of this paper is an in-depth study of the construction of the optimal shared VMB with the fewest forwarding nodes in ad hoc networks. MSCDS in UDG is used to model the optimal VMB, which proves a more accurate model than conventional MStT. This paper proposes a One-hop Algorithm of MSCDS with a constant approximation ratio of 10, and then extends the One-hop to a fully distributed *d*-hop algorithm, which is also with a constant approximation ratio and can organize multicast nodes to a hierarchical VMB, which induces a localized lightweight mechanism for VMB maintenance and update. A security mechanism is also presented to exclude malicious nodes from multicast group based on the hierarchical VMB. Both theoretical analysis and simulation results show that our algorithms outperform flooding.

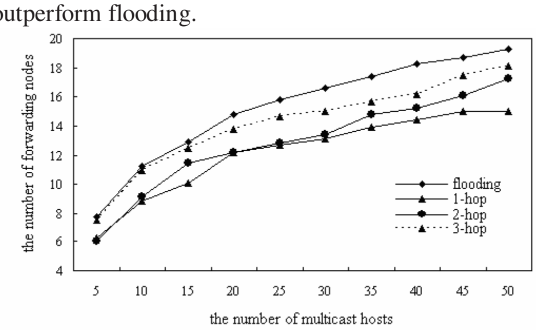

Fig. 4. Average results for *R*=250, and *n*=50.

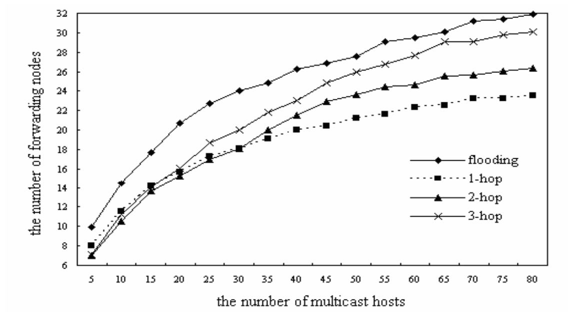

Fig. 5. Average results for *R*=200, and *n*=80.

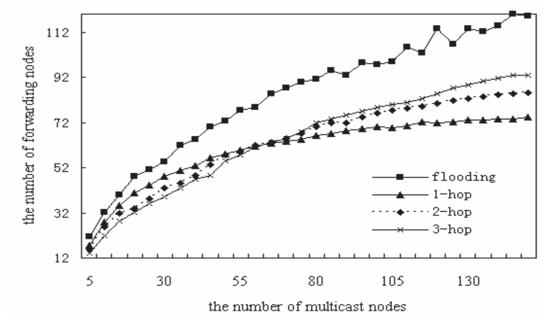

Fig. 6. Average results for *R*=100, and *n*=150.

ACKNOWLEDGEMENT

This paper is supported by National Natural Science Foundation of China [No. 60173048]. Also, we would like to thank Zen yanyan for proofreading and revising, and pointing out the clustering technique in [14].

REFERENCE

[1] C. W. Wu, Y.C. Tay, and C.-K. Toh, "Ad Hoc Multicast Routing Protocol Utilizing Increasing id-numberS (AMRIS) Functional Specification," *Internet draft*, draft-ietf-manet-amris-spec-00.txt, Nov. 1998.

[2] J. J. Garcia-Luna-Aceves and E.L. Madruga, "The Core-Assisted Mesh Protocol," *IEEE JSAC*, Aug. 1999, pp. 1380–94.

[3] K. Chen and K. Nahrstedt, "Effective Location-Guided Tree Construction Algorithms for Small Group Multicast in MANET," in *Proc. INFOCOM*, 2002, pp. 1180–89.

[4] M. Gerla, S. J. Lee, and W. Su, "On-Demand Multicast Routing Protocol (ODMRP) for Ad Hoc Networks," Internet draft, draft-ietf-manet-odmrp-02.txt, 2000.

[5] B.N. Clark, C. J. Colbourn, and D. S. Johnson, "Unit Disk Graphs," *Discrete Mathematics*, vol. 86, pp.165–177, 1990.

[6] S. Guha and S. Khuller, "Approximation algorithms for connected dominating sets," *Algorithmica*, vol. 20(4), April 1998, pp. 374-387.

[7] V. Bharghavan and B. Das, “Routing in Ad Hoc Networks Using Minimum Connected Dominating Sets,” in *Proc. Inter-national Conference on Communications’ 97*, Montreal, Canada. June 1997.

[8] I. Stojmenovic, M. Seddigh, and J. Zunic, “Dominating sets and neighbor elimination based broadcasting algorithms in wireless networks,” in *Proc. IEEE Hawaii Int. Conf. on System Sciences*, January 2001.

[9] W. Lou and J. Wu, “A cluster-based backbone infrastructure for broadcasting in MANETs,” in *Proc. Parallel and Distributed Processing Symposium,* April 2003.

[10] J.Wu and H.L.Li,“On calculating connected dominating set for networks,” in *Proc. the 3rd ACM inter-national workshop on discrete algorithms and methods for mobile computing an communications*, 1999, pp. 7–14.

[11] G. Singh, and K. Vellanki, “A distributed protocol for constructing multicast trees,” in *Proc. International Conference on Principles of Distributed Systems*, (Amiens, France, December 1998).

[12] K. M. Alzoubi, P.-J. Wan, O. Frieder, “New Distributed Algorithm for Connected Dominating Set in Wireless Ad Hoc Networks,” in *Proc. HICSS35*, Jan. 2002, pp. 297–304.

[13] M. Gerla and J. Tsai, “Multicluster, mobile, multimedia ratio network,” *ACM-Baltzer J. Wireless Networks,* vol. 1, no 3, pp. 255-265, 1995.

[14] C.R. Lin and M. Gerla, “Adaptive clustering for mobile wireless networks,” *IEEE J. Select. Areas Commun.,* vol. 15, no. 7, pp. 1265-1275, Sept. 1997.

[15] E. M. Royer and C. E. Perkins, “Multicast Operation of the Ad Hoc On-Demand Distance Vector Routing Protocol,” in *Proc. MOBICOM*, Aug. 1999, pp. 207–218.

[16] K. M. Alzoubi, P.-J. Wan, O. Frieder, “Message-optimal connected dominating sets in mobile ad hoc networks,” in *Proc. MOBICOM*, Jun. 2002, pp. 157–164.