NEW TECHNOLOGIES OF MULTICASTING IN MANET

SHUHUI YANG * AND JIE WU

Abstract. As the technology of mobile ad hoc networks (MANETs) develops, many new kinds of applications in this field emerge. The group-oriented services which take advantage of the broadcasting nature of wireless network are of much importance. Therefore, multicasting protocols in MANET are receiving increased attention. This paper gives a general survey of the state-of-the-art technologies in multicasting protocols over MANET after a short discussion of the more traditional tree-based and mesh-based multicast routing protocols. These new technologies are analyzed and classified according to their underlying principles. At last, we give a brief introduction to two other multicasting protocols, which are different from the others due to their special focuses.

Key words. Backbone-based approach, mobile ad hoc network (MANET), multicasting protocol, overlay network, stateless multicast.

1. Introduction. A mobile ad hoc network (MANET) [1] [12], is a dynamic self-configurable wireless network, which has no fixed infrastructure or central administration. These characteristics make MANETs suitable for mission-critical applications, such as disaster recovery, crowd control, search and rescue and automated battlefield communications, yet make the routing in MANETs very difficult. Nodes can move arbitrarily, network topology can change frequently and unpredictably, and the bandwidth and battery power are limited. For these reasons, the development of routing protocols in MANET is extremely challenging. Multicast plays an important role in MANET. Many ad hoc network applications need the nodes to work as a group to carry out a given job. This kind of application is efficient due to the broadcast nature of wireless network for it can improve the efficiency of the wireless links. As a result, multicast routing has become a research focus recently, and various multicasting protocols in MANET have been proposed.

Multicasting is the transmission of data packets to more than one node sharing one multicasting address. The senders and receivers form the multicast group. Actually, there could be more than one sender in a multicast group, so it is group-oriented computing. In wired networks, some well established routing protocols can provide efficient multicast, but when it comes to MANETs, these protocols may fail due to some unique characteristics of MANETs. When designing protocols for ad hoc network multicast, some key issues should be kept in mind. These include constant update of delivery paths, dynamic group membership, and as little state information as possible. Several multicast routing protocols have been proposed for MANET, which can be classified as either *tree-based* or *mesh-based* according to the kind of routes they create. In the former, all the routes forms a tree infrastructure with the source node as the root, thus there is only one single path between every pair of sender and receiver. Obviously, it's very efficient since the routing information needs to be maintained is very little. In the latter, a mesh infrastructure is maintained as the routing information, that is, more than one path between each sender and receiver pair exists, so it is more robust but less efficient. Both use similar routing discovery and maintenance methods. They use some kind of 'scoped flooding' to find the routes, and store the route information in the routing table of the intermediate nodes (always noted as *forwarding nodes*). The data delivery just follows this routing information. Routes are periodically updated and recovered when failure occurs. More recently, there are trends toward methods which aim to reduce the state information in the network and thus to reduce the overhead. Examples are *overlay-based approach*, *backbone-based approach*, and *stateless approach*, which all reduce the protocol

*Department of Computer Science and Engineering, Florida Atlantic University, Boca Raton, FL 33431 Email:{syang,jie}@cse.fau.edu.

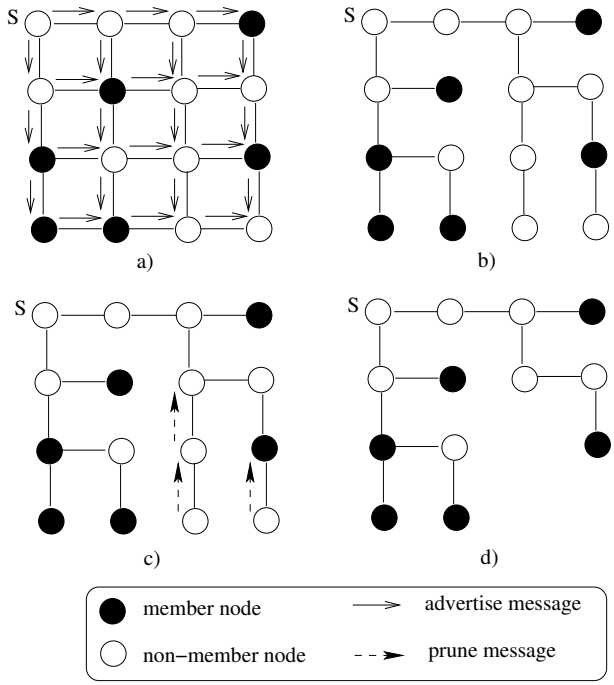


FIG. 2.1. *Multicast tree construction of DVMRP. (a) Broadcast of* Advertise *message. (b) Source-based shortest path tree among all nodes. (c) Unicast of* prune *message. (d) Source-based multicast tree.*

states in some way.

The rest of the paper is organized as follows. Section II presents an overview of the more traditional tree-based and mesh-based multicasting protocols. Section III introduces the latest technologies for multicasting in MANETs with analysis and classification. Section IV gives a brief comment on two new multicasting protocols which focus on reliable multicasting. Concluding remarks are made in Section V.

2. Multicasting Protocols in Mobile Ad Hoc Networks. MANET is a highly dynamic environment, so the traditional well established multicasting protocols cannot be deployed directly to it. Some modification and extension should be made while considering all the constraints, such as dynamic network topology, limited bandwidth and power. The new protocols should avoid global flooding, should dynamically build the routes, and should update both routes and memberships periodically.

2.1. Tree-based Multicasting Protocols. This tree-based concept is borrowed from the multicasting protocols in wired networks. Since efficiency can be achieved and robustness is not a critical issue in the stable wired network, most multicast methods are tree-based, either *source-* or *shared-tree-based*. The former one will construct a multicast tree among all the member nodes for each source node; usually this is a shortest path tree. This kind of protocol is more efficient for the multicast, but has too much routing information to maintain and has less scalability. The latter one constructs only one multicast tree for a multicast group including several source nodes. Every source uses this tree to do multicast. Usually the shared tree constructed is a minimum spanning tree. Since the path between a sender and a receiver is not necessarily the shortest path, the shared-tree-based protocol is less efficient than the source-based protocol in doing multicast, but it reduces the overhead greatly by

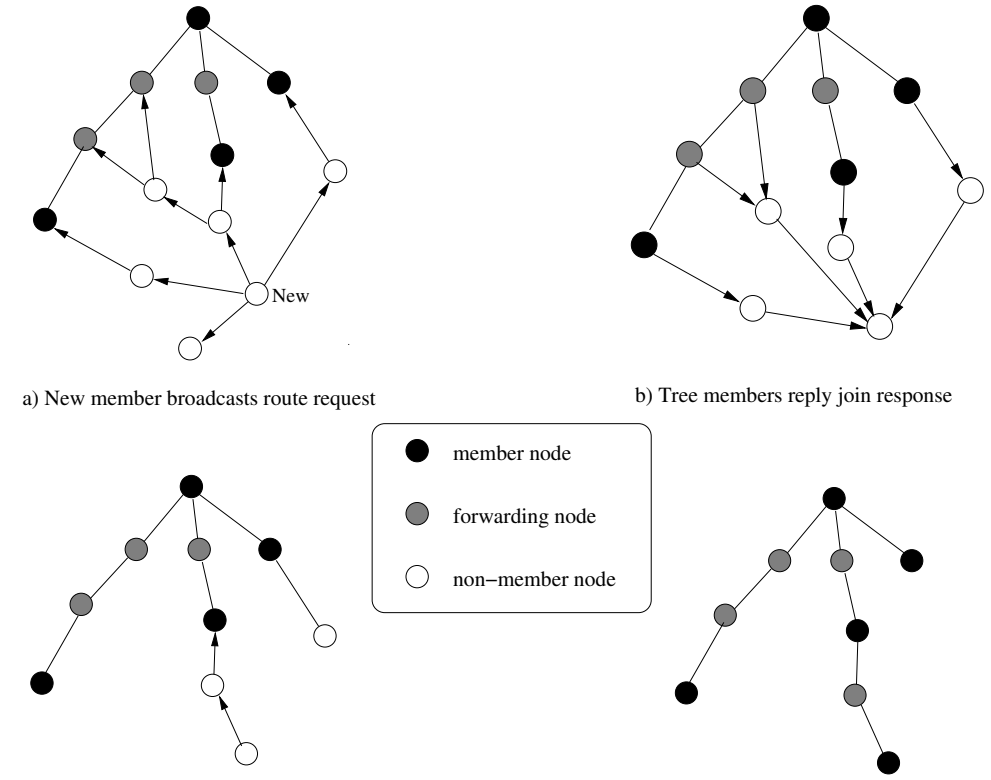


FIG. 2.2. *Member join procedure of MAODV.*

maintaining less routing information. To let these multicasting protocols work in MANET, some modification and extension should be made. The following two protocols are developed for MANET.

DVMRP (Distance Vector Multicast Routing Protocol) [18]. This protocol uses reverse path algorithm to construct a source-based multicast tree in a multicast group for each source node. In DVMRP, there are two main phases to construct a multicast tree. A source node will broadcast an *Advertise* message at the first phase. Every intermediate node will forward this message if the link from which it gets the message is on the shortest path back to the source. At the same time, every node can decide its parent node, which is the previous node on its shortest path to the source. Therefore, the parent-child relationship is established and a shortest path tree rooted on the source node among all the nodes is constructed (see Figure 2.1 (b)). The second phase is pruning. Every leaf node, if it is not a group member, will send a *prune* message to its parent node. The intermediate parent node will send a *prune* message to its parent if there is no group member in the subtree rooted by itself, as shown in Figure 2.1 (c). In this way, the subtrees without member nodes are pruned by being deleted from their parent's forwarding table. Therefore, the source-based multicast tree is established (see Figure 2.1 (d)). Later, when a non-member node wants to join this multicast group, it will send a *graft* message to its parent. The parent will forward the *graft* message upstream until the message reaches an on-tree (this group's multicast tree) node. In this way, this branch is grafted on the source-based multicast tree.

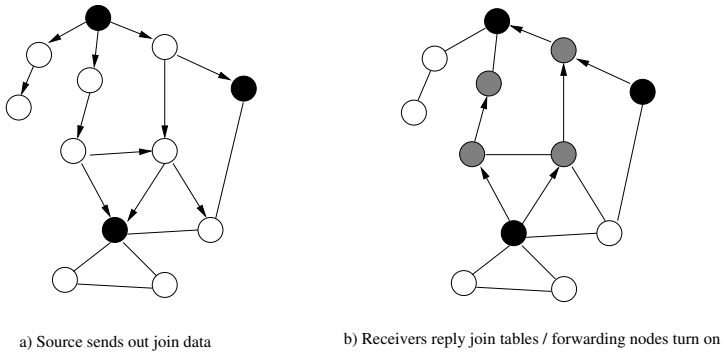


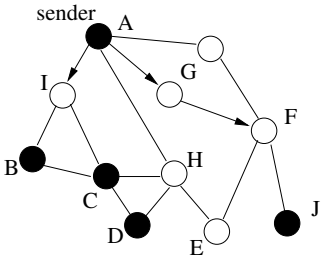
FIG. 2.3. *Creation of forwarding mesh in ODMRP.*

MAODV (Multicast On-demand Distance Vector routing protocol) [19]. This protocol uses broadcast to find the route in an on-demand way and constructs a shared routing tree. The node who wants to join a multicast group or has data to send will broadcast a *Route Request (RREQ)* message. This message will be rebroadcasted by all the intermediate nodes until it reaches an on-tree (this group's multicast tree) node. This on-tree node can then reply a *Request Response (RREP)* message by unicast along the reverse path to the sender. The sender node may get more than one RREP, if so, it will select the best one based on sequence number and hop count, then unicast an *activation message* along this selected path. Every intermediate node on this path will be a forwarding node. It sets up entry in its routing table to add the sender and itself on the tree (see Figure 2.2). In this way, the multicast tree has only a single path to any tree node. This protocol uses hard state in its routing table, that is to say, the state information is updated when failure occurs, contrary to soft state, in which routing table is updated periodically. When a link failure occurs, it will be detected and some kind of repair will be done.

There are some other tree-based protocols, such as MOSPF (Multicast extension to Open Shortest Path First Protocol) [17] and LAM (Lightweight Adaptive Multicast) [10].

2.2. Mesh-based Multicasting Protocols. Correspondingly, the mesh-based method is much more suited for MANET, which demands more robustness of the protocol. That is, when a route fails, which is common in mobile ad hoc networks, there should be another route to deliver the data. It is the redundancy of the routes that provides the fault tolerance.

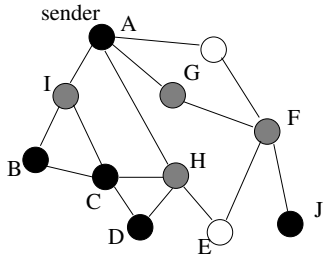
ODMRP (On-Demand Multicast Routing Protocol) [14]. This protocol uses the concept of 'forwarding node' to do the multicasting. It finds some nodes (group member nodes or non-member nodes) to be 'forwarding node' in the whole network, and only these nodes will forward multicast messages. The source on-demand establishes the routes by broadcasting the *JoinData* message with TTL. This message is periodically generated to refresh both the membership and routes. Every intermediate node will add the upstream node's ID in its own routing table upon receiving this message. The message will be forwarded until it reaches a group member. The group member then creates a *JoinTable* message and broadcasts this message to its neighbors. Every neighbor node will know that itself is on the path between the source and the group member if the next hop ID in one of the entries of the *JoinTable* message meets its own ID. This neighbor node then establishes itself as a *forwarding node*. It sets the *Forwarding Group Flag* on. Then it builds its own *JoinTable* message based on routing table and propagates it on until the message reaches the source via the shortest path (see Figure 2.3). The mesh of forwarding nodes is established in this way. This forwarding group supports the shortest paths between any member pairs. The source can send data to all



a) Transmission of forwarding tables

receiver	next hop
B	I
C	I
D	H
J	G

c) Forwarding table of node A



b) Creation of forwarding group

receiver	next hop
J	F

d) Forwarding table of node G

FIG. 2.4. *Creation of forwarding group in FGMP.*

the group members with the help of the mesh. Only the forwarding nodes will forward the multicast data. It is a soft state protocol and there is no need for the group members to send explicit messages to leave the group. Members can stop working at any time, and this change can be detected by the periodic refreshment.

FGMP (Forwarding Group Multicast Protocol) [3]. This protocol also uses the concept of *forwarding group* to keep track of nodes which participate in data transmission, but not the traditional links. The source just broadcasts data, and only the forwarding nodes will rebroadcast it. This protocol actually is a kind of 'limited scope' flooding. Each forwarding node maintains only two parameters. One is a flag which indicates the forwarding node's status of *on* or *off*, the other is a timer. The forwarding node is only effective before the timer expires. This protocol is the twin method to ODMRP, only the way to create the forwarding group is different. The creation and maintenance of the forwarding group can be done in a *sender-initiate* way or a *receiver-initiate* way, that is, either a sender node advertises itself and begins the mesh construction procedure or a receiver does so. They are quite the same, generally. The former is more efficient when there are less senders than receivers in the multicast group. In receiver advertising method, each receiver will flood its membership periodically. The sender collects this information to create and update a *member table*. The sender creates the *forwarding table* based on the *member table* and some preexisting routing tables. Then it broadcasts the *forwarding table* to all the neighbors. Only the neighbors which are in the next hop list of the *forwarding table* will accept it and create their own forwarding tables. They rebroadcast the table until it at last reaches the receiver (see Figure 2.4). By this forwarding table transmission, the forwarding nodes are selected. Some newly developed methods use dominating set to create and maintain the forwarding group.

There are some other mesh-based protocols, such as CAMP (Core-Assisted Mesh Protocol) [6] and NSMP (Neighbor Supporting ad hoc Multicast Routing Protocol) [13].

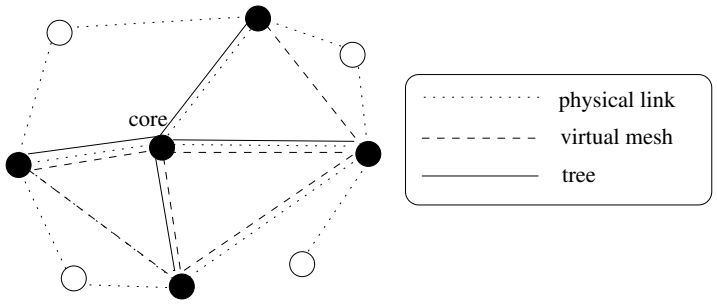


FIG. 3.1. *Virtual multicast tree in AMRoute.*

2.3. Hybrid Multicasting Protocol. There is a hybrid approach named AMRoute (Ad hoc Multicast Routing) [22]. This protocol is a combination of tree-based and mesh-based methods to seek both efficiency and robustness. It has two main procedures. One is mesh creation, the other is tree creation. It first creates the virtual mesh links among the group members based on the physical links. A logic core will be selected from the members in this procedure. It then uses this mesh to establish the multicast tree. The logical core will initiate the tree creation. The tree can stay unchanged even if the topology of the network changes, as long as the links between the core node and tree members still work with the help of virtual mesh. The neighbors on the multicast tree are connected by the underlying unicast tunnels which have the responsibility to deal with dynamic network topology. Both the tree and mesh are quite static (see Figure 3.1).

More detailed information about these two kinds of multicasting protocols can be found in a recent survey [4]. In [15], some performance evaluation and comparison of these protocols is presented.

3. New Technologies of Multicast Routing Protocols in MANET. The multicasting protocols in MANET discussed above are much more traditional. They have some disadvantages. When the network size or number of source nodes increases, their performance will decrease significantly. Take ODMRP for instance, which can exhibit a high package delivery ratio even at high mobility; it will suffer from higher control overhead when the application scales. With some analysis, we can find that, in ODMRP, the source node initially floods a *JoinReq* package to the whole network, then the expected group members will send back *JoinReply* packages along the reverse paths to the source. All the intermediate nodes will become *forwarding nodes* to establish the routes. The state information is maintained by periodically flooding the *JoinReq* control package. This protocol provides the robust routes at the expense of increased overhead. This will lead to inefficiency when the network or group is large. The new trend of design of multicasting protocols in MANET is to reduce such overhead. Some efforts have been made in this field. We classified these new technologies into the following four categories.

- *Overlay-based multicasting*. This method constrains the protocol states within the group members, to avoid the explosion of state information if it is kept in all the forwarding nodes.
- *Backbone-based multicasting*. In these protocols, the state information is confined within the virtual backbone only.
- *Stateless multicasting*. In these protocols, there is no need to maintain any states in the forwarding nodes at all.

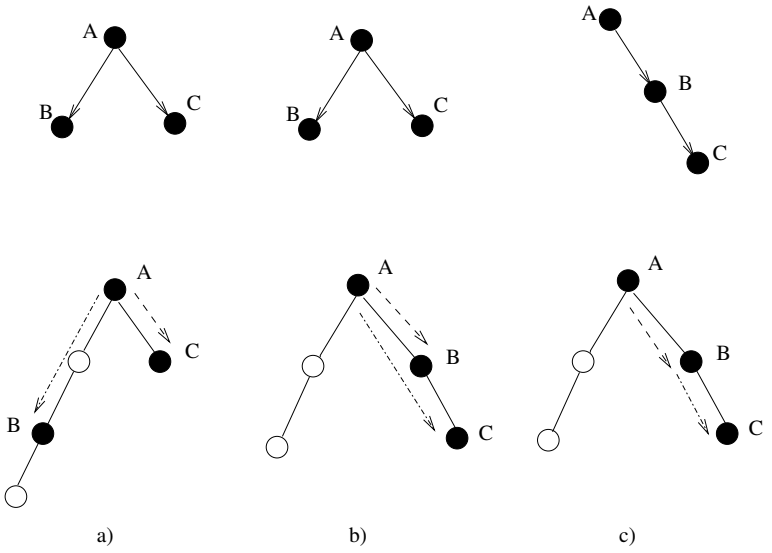


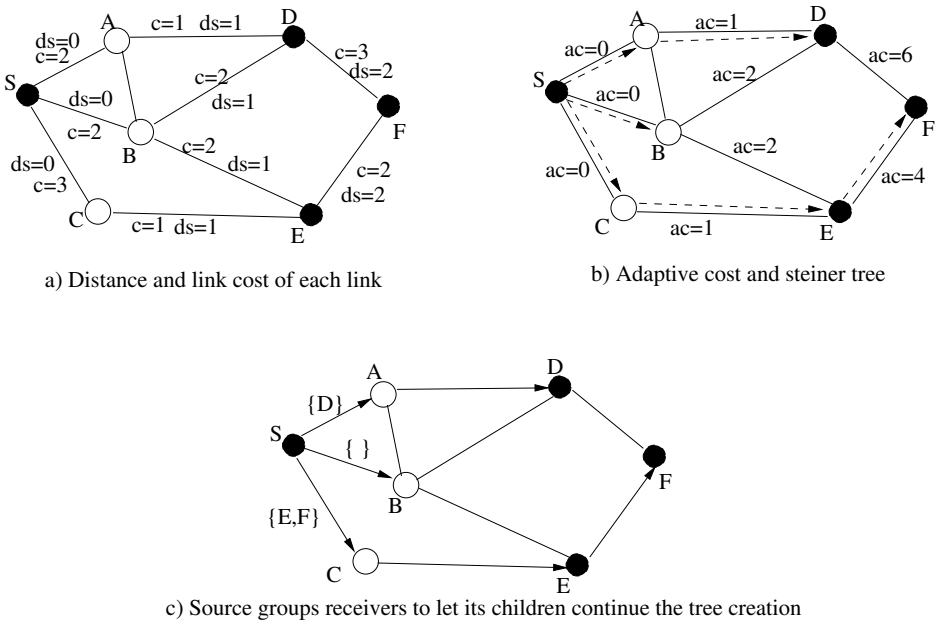
FIG. 3.2. *Efficiency of Overlay Multicast. (a) Initial network and overlay topology. (b) Network and overlay topology after movement of node B. (c) Optimization of the overlay network after movement of node B.*

- *Other multicasting protocols.* The protocols of this category use an explicit way to reduce the control overhead of some existing protocols. They are a form of direct extension of the traditional ones.

3.1. Overlay-based Multicasting Protocols. The management of state information of multicasting protocols includes periodic updating of the routing tables to maintain the correctness of the routing structure, either a tree or a mesh. Since the traditional protocols involve both the group member nodes and non-member nodes, say, forwarding nodes, to maintain the state information, they will encounter the problem of scalability. One way to address this problem is to constrain the protocol states only in the group member nodes. In the overlay multicast, a virtual infrastructure is built on the top of the underlying physical links among only group members. The links in the virtual network are unicast tunnels in the physical network. The virtual network will deal with the multicast functionalities while the underlying physical links' job is to provide a best-effort unicast datagram service.

The AMRoute [22] discussed before is this kind of protocol in some ways. It tries to use the infrastructure of both tree and mesh to achieve better performance, meanwhile the overlay network forms. But a big problem of the overlay multicast method is that the relatively static upper network may cause redundancy in date delivery in the existence of the change of underlying topology. We can see this in Figure 3.2. In (b), there is a redundant data delivery link between node *A* and *C* if the overlay network remains unchanged. The more efficient solution is (c). AMRoute can't solve this kind of problem because the shared upper tree structure is always built based on the static virtual mesh information.

In [8], a new overlay multicast protocol for MANET called PAST-DM is put forward. It can effectively alleviate the overlay network problem mentioned above. It uses the frequently updated link states to maintain the virtual mesh, and develops a novel *source-based steiner tree* algorithm to construct the upper source-based tree infrastructure. Both these aspects can help it avoid the redundant link problem in some ways. Just like AMRoutes, the multicast begins with the creation of the virtual mesh on the top of the physical links. All the member nodes periodically exchange *GroupReq* messages to discover and keep track of the neighbors.



a) Distance and link cost of each link

b) Adaptive cost and steiner tree

c) Source groups receivers to let its children continue the tree creation

FIG. 3.3. *Tree construction in PAST-DM.*

By doing so, each member node can get the group topology; it then represents the topology as a link state table. The group members can form a dynamic virtual mesh. Each source of the group will construct its own source-based data delivery tree based on its local link table using the *source-based steiner tree* algorithm. If we denote the distance between the source node s and node n as $ds(n)$, then the distance between source node to a virtual link (n_1, n_2) can be defined as $ds(n_1, n_2) = min[ds(n_1), ds(n_2)]$. If $c(n_1, n_2)$ denotes the cost of virtual link (n_1, n_2), then the *'adaptive cost'* of this link to the source is given as $ac(n_1, n_2) = ds(n_1, n_2) * c(n_1, n_2)$. The source can create its steiner tree by selecting the smallest *'adapted cost'* links. In Figure 7, we can see that all the neighbors of source node have 0 as the adaptive value, so they will be first-level children of the source. The tree grows by including the nearest receiver to itself gradually. The source doesn't need to compute the whole tree. It can group the receivers (all the other group members) and send each first-level child a unicast packet with the subgroup receivers in the packet header. The child will create the sub-tree rooted at itself according to the most recent information and do the data forwarding (See Figure 3.3 (c)). Because the multicast tree can be updated during the next incoming link state exchange, the newly constructed tree is always the most efficient one.

3.2. Backbone-based Multicasting Protocols. The backbone-based multicasting protocols use another method to constrain the state information. They select some nodes to form a virtual backbone of the network, and the state information can only be held in the backbone nodes.

The adaptive backbone-based multicast protocol for ad hoc networks [9] is such a protocol. It is based on a two-tier hierarchical approach. The backbone is composed of the core nodes which will forward the multicast data among themselves; the maintenance and forwarding of the membership are done only in the inside local group rooted at a core. This combines both the effectiveness of the flooding scheme and the efficiency of the tree scheme. There are also some other backbone-based multicast protocols in MANET [7], [20], but they

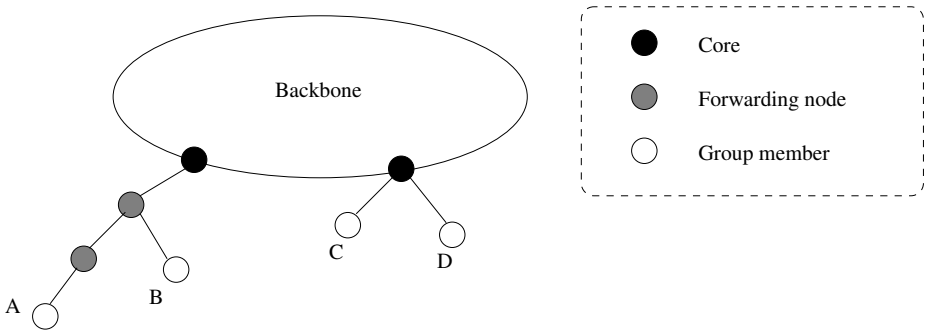


FIG. 3.4. *Structure of the backbone network and forwarding tree.*

all use the concept of dominating set to create the backbone. That is, every node in the network is just one hop away from a core or it is a core itself. This method has the problem that the number of cores is in the same order of the number of nodes. They can't effectively reduce the control overhead by constraining the multicast in backbone nodes. The adaptive backbone-based approach creates the *Adaptive Dynamic Backbone* (ADB) which allows the noncore nodes to be far away from the cores as long as the mobility in the nodes' local area is not that high. The ADB creates a forest with several trees of various depth. If the local area of a tree is relatively static, the depth of the tree is allowed to be large, otherwise it has to be small (see Figure 3.4). So it is adaptive according to local current network environment.

At first, every member node will set itself to be a core and send out a *Hello* message to its neighbors. When a node receives this *Hello* message from other nodes, it will calculate the *height* value, which is some kind of metric such as detected link failure frequency, remaining power or degree of connectivity, *et al*. As an example, we can use the three-parameter tuple *($nlff^{-1}$,degree,id)* as the *height* value (*nlff* is normal link failure frequency). Then it will decide whether it should remain a core or become another core's child based on the *height* values. If it chooses to be another core's child, it will record the core's information and when it sends out its next *Hello* message, it will indicate its core's information and also replace its original link failure frequency with a new accumulated *nlff*. In this method of message exchange, the cores are selected to form the backbone. The trees rooted at cores won't be too high with the limitation of 'accumulated *nlff* constraint' or 'hop limit constraint'. The height of a tree depends on the mobility of nodes in that area. Every node also maintains the *core forwarding table*. It can get some information about other cores which are not its parent core via *Hello* messages of its neighbors who do not belong to the same core. Then it just records the routing information to those other cores. In a *core-forward-update* period, every child sends its *core forwarding table* to its parent. At last these tables reach the cores, thus the cores know the routes to other cores. When a node wants to join a multicast group, it just sends out a *Join-Request* to its parent. If the parent is not already part of the forwarding tree, it keeps forwarding the message upstream and sets itself as a *forwarding node*, otherwise, it will ignore the message. In this way, a multicast forwarding tree is constructed, which is rooted at the core and spans all members in this local group. When sending data, the group member will locally broadcast the packet, and the forwarding nodes will rebroadcast the packet. When the data reaches the core, the core will relay it in the backbone range by encapsulating the packet into a *CORE-FORWARD* message and forwarding it to the next hop core(s) in its *core forwarding table*. When other cores get the message, they remove the *CORE-FORWARD* header and do the group level multicast as described before.

The backbone topology is much more simple and stable compared with that of the whole

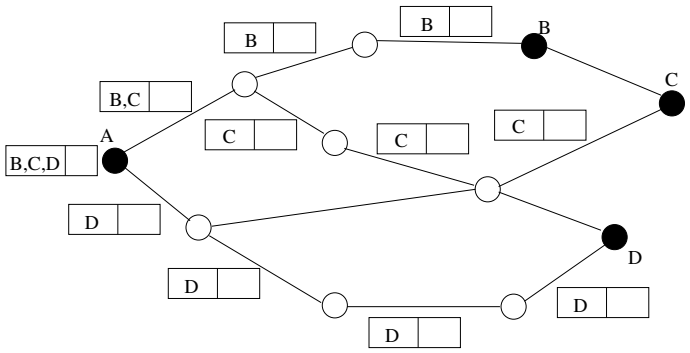


FIG. 3.5. *Transmission of DDM packets.*

network. But the cores are 'hot spots' of the network, and this will put limits on the horizonal scalability of the network, and the number of the local groups, because all the multicast packets will pass the same set of core nodes.

3.3. Stateless Multicasting Protocols. The multicasting protocols discussed so far have some state information maintained at the forwarding nodes to keep the routing structure. Recently, there is a shift towards designing stateless protocols for MANETs. These protocols do not require the routing information to be kept at forwarding nodes at all to further reduce overhead. The following two protocols use different methods to achieve this goal.

Differential Destination Multicast (DDM) [11]. This is a stateless MANET multicast routing protocol for small communication groups. In DDM, the source node explicitly mentions the destination addresses by encapsulating them in the headers of the data packets. The intermediate nodes with the DDM agents on them will take charge of the delivery of the packets. They will look into the header of the packet to find out the destinations, then query the underlying unicast protocol to find out the next hop information (see Figure 3.5). In this way, the forwarding nodes won't need to keep any multicast routing information. But it is obvious that this protocol only suits the small size communication group because with the growth of the group, the packet header will become larger and larger and lose efficiency.

In DDM, the source node controls the membership information. This protocol has two modes. One is stateless, just as described above, in which the forwarding nodes rely on the underlying unicast protocol to forward every packet; the other is soft state. In soft state mode, the forwarding nodes will remember the destinations of the last packet sent and the corresponding next hop information. The following packets sent by the same source do not need to have the *destination headers*. They can be forwarded based on the in-band information at the forwarding nodes. Only when the destination list has some change, does the source need to notify the forwarding nodes; thus, the name *differential destination*.

When a node is interested in some multicast group, it will unicast a *Join* message to the source node. The source node will decide whether to accept it according to some admission policy. It will then unicast the *ACK* message to the node and add it to the *member list (ML)*. This ML list kept in the source node should be refreshed periodically. The source will set a *POLL* flag in the outgoing packet every several packets. The destination nodes will unicast the *Join* message to the source on receiving this special packet to show its existence.

This protocol is not a general purpose multicast protocol, for it can't work well with large group size due to the header-encoded mechanism. But it can excel in the horizonal scalability, which is the growth of the number of the multicast groups.

Effective Location-Guided Tree Construction Algorithm (LGT) [2]. LGT is another

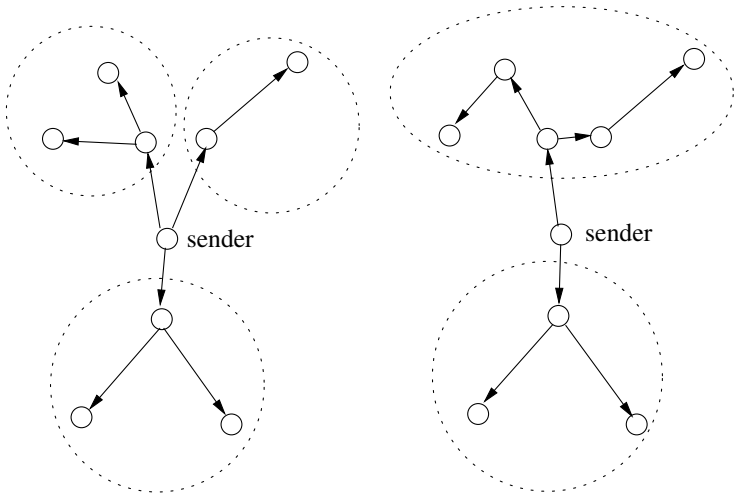
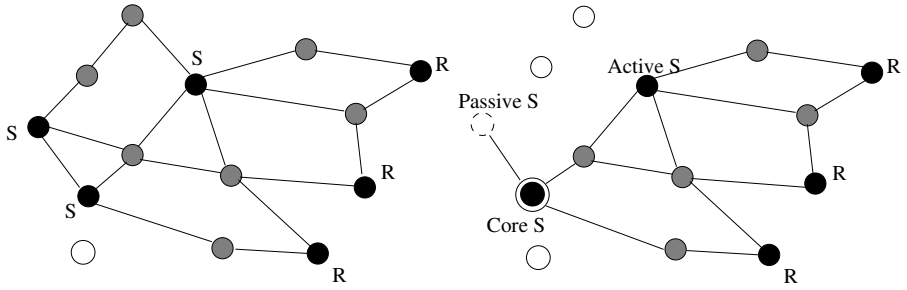


FIG. 3.6. *Example of LGK & LGS tree.*

multicast protocol in MANET for small communication group. In this protocol, an upper overlay packet delivery tree is created on top of the underlying unicast protocol. The multicast packet is encapsulated in a unicast envelope and unicasted among the group members. The difference between LGT and DDM is that in DDM, the distribution tree is not controlled by the upper transport layer, but in LGT, the tree is constructed with the flexibility of adding upper layer's packet routing in order to minimize the overall bandwidth cost of the tree. Also the DDM requires all nodes to cooperate, but the LGT needs only the group member nodes to do so.

The operation of this protocol is similar to DDM. The source node will control the membership of the group and send out packets with destination-included headers. The forwarding nodes will construct a data distribution tree based on the destination list they get from data packets. The tree is composed of all the group nodes. Then the forwarding nodes use the underlying unicast protocol to forward the packets. The main part of this protocol is the design of the tree construction algorithm. Two of these algorithms are developed. One is *location-guided k-ary tree (LGK)* algorithm, the other is *location-guided steiner tree (LGS)* algorithm. Both utilize the geometric location information of the destination nodes as the heuristics to construct the tree without knowing the global topology of the whole network. These algorithms are distributed; each node is responsible for the construction of the outgoing branch of its level and also data forwarding.

In LGK, the source node selects the k nearest destination nodes as its children and groups the remaining nodes to each child according to geometric proximity. It then sends a copy of the packet to each child. This child will do the same procedure to forward the packet until all destination nodes get the packet. In this way, the packets are forwarded along the k-ary tree to the destination via unicast routing. The LGS tree is constructed using a modified version of the *Takahashi-Matsuyama* heuristic. The difference is that it uses the geometric distance as the measurement of closeness and only the group member nodes can be used as the tree nodes. The source will select the nearest neighbor to form the first link of the tree. Then at each iteration, the nearest unconnected destination to the tree will be added onto the tree (see Figure 3.6). These two algorithms are used under different conditions to achieve better



a) ODMRP forwarding group b) DCMP forwarding group

FIG. 3.7. *Forwarding nodes in ODMRP & DCMP.*

performance. The LGS tree has lower bandwidth cost when the location information of the nodes is up-to-date. When that location information becomes difficult to keep up-to-date, the difference of cost between the two algorithms is not that significant. So the LGK should be used to take advantage of its lower distribution delay and computational complexity.

3.4. Other Multicasting Protocols. In [5], some modification is made to ODMRP. It dynamically classifies the source nodes into *active* and *passive* modes; only the *active* ones perform the function of periodic routing refreshment. In this way, the new protocol named dynamic core based multicast routing protocol (DCMP), can successfully reduce the control overhead by reducing the number of forwarding nodes. The simulation shows that it can increase the multicast efficiency by $10 - 15\%$ compared with ODMRP.

In DCMP, source nodes are classified into three categories, *active sources*, *core active sources*, and *passive sources*. The active sources are the same as the sources in ODMRP. They will broadcast *JoinReq* messages when they have data to send or to maintain the routing. The *passive sources* won't do this. When they have data to send, they will forward the data to the *core active sources* to which they belong, which are selected from the *active sources* to act as agents to one or more *passive sources* and are in charge of forwarding the *passive sources*' data to group members.

At first, every source is *active source*. It can broadcast the *JoinReq* packet when it has data to send, just like the operation in ODMRP with the exception that the packet has an additional flag called *CoreAcceptanceFlag*, which indicates whether this source can support some other *passive sources* or not, according to its own parameter *MaxPassSize*. For example, A broadcasts a *JoinReq* packet. When an *active source*, say, B, receives this packet, it will change its status to *passive source* if the following three conditions are met: (a) the *CoreAcceptanceFlag* in this packet is set to on; (b) the distance between these two sources A and B is less than the predefined parameter *MaxHop*; (c) the ID of source B (*ToBePassive source*) is less than the original source A (*ToBeCore source*). If so, it will reply a *PassReq* packet to A, and come into the *ToBePassive source* mode. After that it will neither become *core active source* for other nodes nor send *PassReq* to other nodes. The *ToBeCore source* node A will decide to be B's core or not, then send back again another *Confirm* packet to let B know. In this method of message exchange, the intermediate nodes will create routing in their routing tables. When the *passive source* has data to send, it will forward it to its *core active source* along the already existing path. Due to the mobility of nodes, it is possible for the *passive source* to move too far away from its *core active source*. It can detect this by receiving the *core*'s *JoinReq* after its movement. It then will send the *PassReq* packet with the *CoreReq* field reset to the *core*. Then both of them together with the intermediate nodes know about

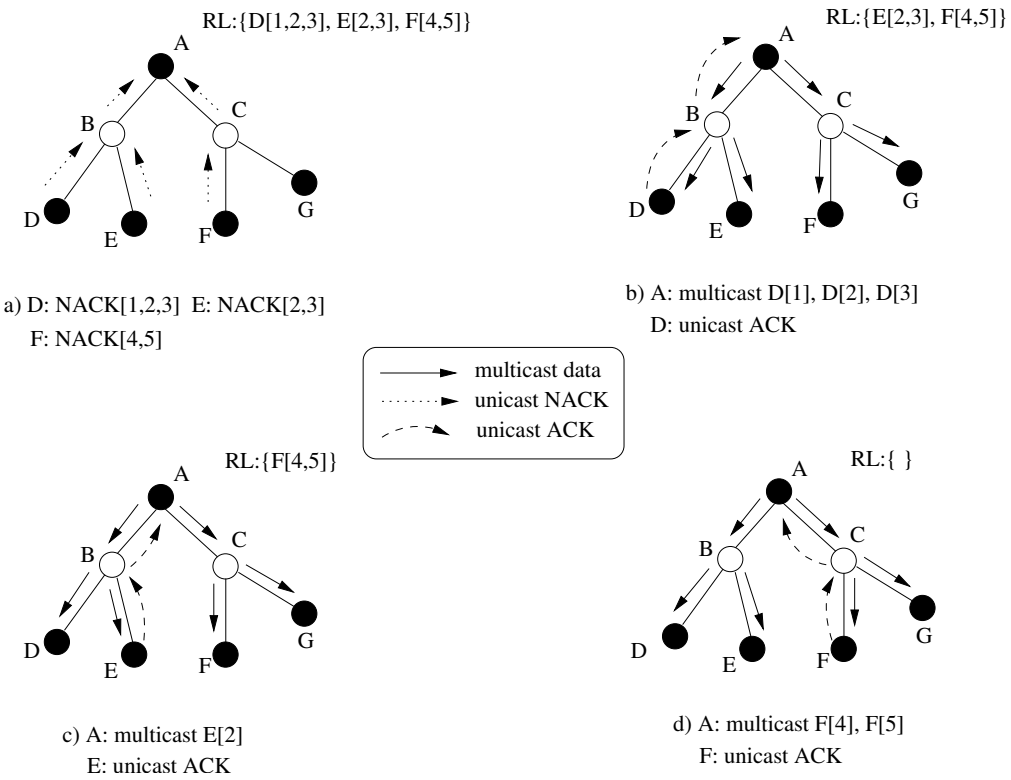


FIG. 4.1. *RALM operation.*

the status change. In Figure 3.7, we can see that the number of forwarding nodes is less than in ODMRP.

The numbers of *passive active sources* and *core active sources* are bounded by the parameters *MaxPassSize* and *MaxHop*. If the number of *passive active sources* is large, the control overhead can be reduced sharply, but the routing mesh will be of less robustness because only the *active sources* are in charge of the maintenance of the routes, which may be deficient. This tradeoff should be considered when setting the parameters.

4. Reliable Multicasting Methods in MANET. Some efforts have been made to develop reliable multicasting protocols. There are three ways to provide some extent of reliability to multicast in the network layer. One is NACK-based method, the second is flooding, the third is the gossip method, which is 'flooding with some limitation'. In wired network multicast, all these methods have been studied and exploited to provide reliability. But little work has been done for MANET multicast. In this section, we will introduce two reliable multicast protocols. One is NACK-based, the other uses the gossip-based probabilistic method.

4.1. Reliable Transport Multicasting Protocol. Reliability is a very important aspect of multicasting protocols in MANET. However, little work has been done in this field. In [21], the *Reliable Adaptive Lightweight Multicast Protocol (RALM)* is put forward to address both the reliability and congestion control problems of multicasting protocols in MANET. This protocol uses the round-robin fashion to deliver data to one multicast member per time in case of data loss to guarantee the reliability and uses send-and-wait approach to do congestion control.

The source initially multicasts data at the rate specified by the application until it detects some data loss by receiving the *NACK* messages unicasted from some member nodes. Then begins the recovery phase. The source node adds the senders of *NACK* to its *Receiver List*. It will then select nodes from the *Receiver List* to do the recovery one by one until the list is empty. The source selects a receiver from the list; it then retransmits the data missed by this receiver by multicasting and puts this receiver's ID in the packet header. Only this receiver is allowed to respond to the source by unicasting an *ACK* message after it has gotten all the missed data. Other receivers who have missed the same data packets can get the data too because of the multicasting. They are not allowed to send *ACK* at this time, instead, they will send back *ACK* when next selected by source to do recovery. The source will re-multicast the lost data after a predefined time expires in case it's lost again, until finally it gets the receiver's *ACK*. This receiver can then be deleted from the source's *Receiver List*. The source node will repeat this procedure until the list is empty. It can then revert to the original application sending rate. In this way, the source node is self-clocked and can ensure it won't send data faster than the *ACK* is received when congestion occurs. As in Figure 4.1, if sender node *A* receives *NACK* from node *D*, *E* and *F* and their lost data information (D lost data 1, 2, 3; E lost 2, 3; F lost 4, 5), its receiver list will be 'D,E,F', (see Figure 4.1 (a)). Then it randomly selects one, say *D*, and re-multicasts the data missed by *D*. *D* will reply an *ACK* message after it gets all these packets. Then *D* is removed from *A*'s list (see Figure 4.1 (b)). Other destinations won't reply even if they miss the same data. For instance, *E* has missed some part of data with *D*. It will reply *ACK* message after the source has begun to resend lost data by denoting *E*'s address in the multicast packet header, (see Figure 4.1 (c)). Then *E* will be deleted from the receiver list. The sender selects the next node from the list until it's empty.

RALM is a reliable, rate-based, congestion controlled multicasting protocol. The focus is to provide reliability of the multicasting protocol, not the routing approach itself as the others do. It only suits the small multicast group because the round-robin approach limits its scalability.

4.2. Probabilistic Reliable Multicast in Ad Hoc Networks. In [16], a *Route Driven Gossip (RDG)* method is developed to provide the probabilistic reliable multicast in MANET. The deterministic protocols for multicast in MANET suffer from the tradeoff between reliability and scalability very much due to the instability of the MANET. As a matter of fact, these existing deterministic protocols provide no reliability guarantees at all. The protocols establish exact routes to let messages go to every member in the group with large overhead. Due to the network's congestion, link layer failure or nodes movement, still not every member can get all the messages. On the contrary, if we let the multicast protocol be probabilistic, such as gossip-based, maybe we can achieve better reliability with less overhead. The main idea of using gossip to do routing is that there is no deterministic pre-established routes, every member node has routes to some other member nodes. Therefore, when a member node gets a message, it will forward the message to a small part of the group. After several rounds, this message spreads through the whole group and all the members can get it. This kind of protocol uses controlled redundancy to achieve high reliability.

There already exist some gossip-based multicast protocols for both MANET and wired networks. The novel RDG is aimed at the more practical specification of probabilistic reliability. It uses a pure gossip scheme. All the messages, multicast packets, negative acknowledgements, and membership information will be gossiped. It does not require the multicast primitive at the network layer, and can be deployed on the top of any on-demand routing protocols (use DSR as example).

There are three sessions in the RDG protocol. The *Join session* is for a node to join a multicast group if it has data to send or wants to receive that group's data. The node floods

TABLE 4.1
Active views of members. A '⋆' at V_{ij} means j is in the active view of i.

V	1	4	6	8	10
1	⋆	⋆	⋆		
4	⋆	⋆	⋆	⋆	
6	⋆		⋆		⋆
8		⋆		⋆	⋆
10			⋆	⋆	⋆

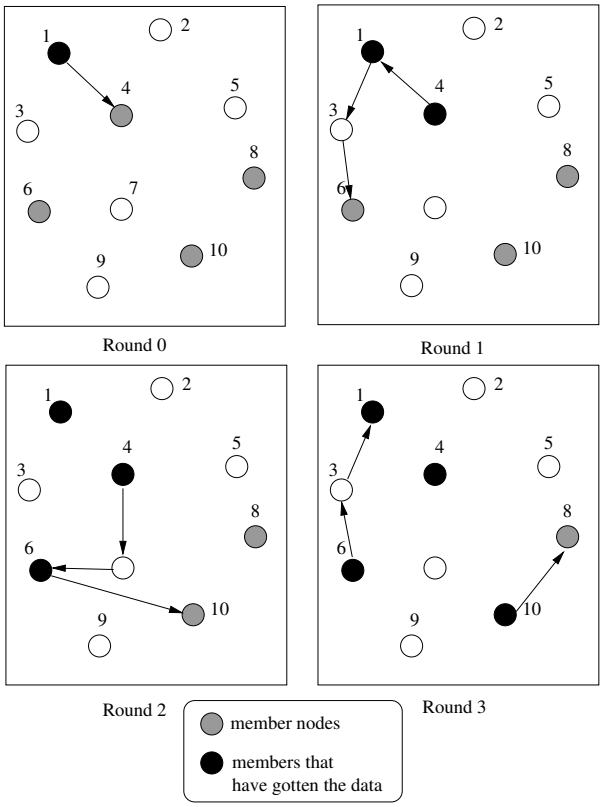


FIG. 4.2. *Example of the execution of RDG protocol.*

a *GroupRequest* message to the network to search for the group's members. This message is the same with the *GroupRequest* message in DSR except that RDG uses multicast group ID as the target address. Each member who gets this message will update its *View* (data structure to store the membership information) by adding this new member and reply the *GroupReply* message with probability P_{reply}. The initiate node also updates its own *View* by receiving *GroupReply* messages. In this way, every member can have a partial view of the multicast group and routes to reach some other members. Multicasting is done in the *Gossip/Leave session*. Each node has a buffer to store data. It will periodically generate a gossip message to F other nodes randomly chosen from its *View*. This gossip message includes some data selected from its own data buffer and also indicates the data it has missed. This selected data will be removed from its buffer after being gossiped for τ_q times. Then members who receive this gossip message will update their own data buffer and reply with the data the sender

requests. When a member wants to leave, it also sends out a gossip message in the same way to let others delete its information from their *Views*. When the amount of information in a member's *View* drops below a predefined threshold, this member has to reinitiate a *Join* session. Even if this protocol is not deterministic but probabilistic, through the gossip method, it can achieve the probabilistic reliability and only has a modest degradation when it suffers scalability and mobility. As an example, there is a group of 10 nodes, 5 of them are members of a multicast group. We define F to be 1 , τ_q to be 2. Table 4.1 is the *Views* of all the members at a certain time. Figure 4.2 is the propagation process of a single data among them using gossip initiated by node 1. We can see that even if none of the members has a global view of the membership, all the members can have the data packet after 3 rounds.

RDG uses a probabilistically controlled flooding technique, which is gossiping, to deliver data packets to the group members. It is in some way a stateless multicasting protocol, too.

5. Conclusions. Multicasting protocols in wired network have been well established. In the mobile ad hoc network, it is really a challenge to design an effective multicasting protocol due to the dynamic, non-infrastructure nature of the MANET. Although lots of work has been done in this field, few can provide a general purpose satisfied protocol.

In this paper, we first give a brief summary on the more traditional multicasting protocols, then lay our emphasis on the new trends and technologies in this field. We classify these new technologies into four categories according to their underlying principles. They use the *overlay networks* or *backbone* to limit the spread of state information, or even let the protocols be *stateless*. Above all, they all apply themselves to reducing the control overhead in the old methods to increase the performance. At last we take a look at two protocols which have their special efforts in the aspect of reliability. One tries to use the reliable probabilistic method to replace the deterministic one to achieve better performance, the other aims at the NACK-based reliable method, designing the congestion control and error recovery mechanism for multicasting protocol. The future work in the design of multicasting protocols in MANET should try to reduce the overall control overhead to support the scalability and robustness. Some key issues of this goal, such as the control of the spreading of the state information and the reliability mechanism, need further study.

Acknowledgments. This work was supported in part by NSF grants CCR0329741, ANI0073736, and EIA0130806.

REFERENCES

[1] *Internet Engineering Task Force (*IETF*) Mobile Ad Hoc Networks (*MANET*) Working Group Charter*.
[2] K. CHEN AND K. NAHRSTEDT, *Effective location-guided tree construction algorithms for small group multicast in* MANET, in Proc. of INFOCOM 2002, 2002.
[3] C. CHIANG, M. GERLA, AND L. ZHANG, *Forwarding group multicast protocol (*FGMP*) for multihop, mobile wireless networks*, Cluster Computing, 1. No. 2 (1998), pp. 187–196.
[4] C. M. CORDEIRO, H. GOSSAIN, AND D. AGRAWAL, *Multicast over wireless mobile ad hoc networks: Present and future directions*, in IEEE Network, Special Issue on Multicasting: An Enabling Technology, January/February 2003.
[5] S. DAS, B. MANOJ, AND C. MURTHY, *A dynamic core based multicast routing protocol for ad hoc wireless networks*, in ACM MOBIHOC, June 2002.
[6] J. J. GARCIA-LUNA-ACEVES AND E. L. MADRUGA, *The core-assisted mesh protocol*, IEEE Journal on Selected Areas in Communications, Special Issue on Ad-Hoc Networks, 17. No.8 (1999), pp. 1380–1394.
[7] M. GERLA, C. CHIANG, AND L. ZHANG, *Tree multicast strategies in mobile, multihop wireless networks*, in ACM Mobile Networks and Applications, vol. 4, 1999, pp. 193–207.
[8] C. GUI AND P. MOHAPATRA, *Efficient overlay multicast for mobile ad hoc networks*, in IEEE WCNC'03, 2003.

[9] C. JAIKAEO AND C. SHEN, *Adaptive Backbone-based Multicast for Ad hoc Networks*, in IEEE International Conference on Communications (ICC 2002), New York City, NY, April 2002.
[10] L. JI AND M. S. CORSON, *A lightweight adaptive multicast algorithm*, in GLOBECOM, 1998, pp. 1036–42.
[11] L. JI AND M. S. CORSON, *Differential destination multicast - a* MANET *multicast routing protocol for small groups*, in Proc. of INFOCOM 2001, 2001, pp. 1192–1202.
[12] J. JUBIN AND J. D.TORNOW, *The* DARPA *packet radio network protocols*, in Proc. of the IEEE, vol. 75. No. 1, January 1987, pp. 21–32.
[13] S. LEE AND C. KIM, *Neighbor supporting ad hoc multicast routing protocol*, in ACM Mobihoc, 2000, pp. 37–50.
[14] S. LEE, W. SU, AND M. GERLA, *On-demand multicast routing protocol in multihop wireless mobile networks*, in ACM Mobile Networks and Applications, special issue on Multipoint Communication in Wireless Mobile Networks, 2000.
[15] S. LEE, W. SU, J. HSU, M. GERLA, AND R. BAGRODIA, *A performance comparison study of ad hoc wireless multicast protocols*, in Proc. of INFOCOM 2000, 2000, pp. 565–574.
[16] J. LUO, P. EUGSTER, AND J. HUBAUX, *Route driven gossip: Probabilistic reliable multicast in ad hoc networks*, in Proc. of INFOCOM 2003, 2003.
[17] J. MOY, *Multicast extention to* OSPF, in Internet draft, 1998.
[18] T. PUSATERI, *Distance vector routing protocol*, in draft-ietf-idmr-dvmrp-v3-07, 1998.
[19] E. ROYER AND C. PERKINS, *Multicast operation of the ad-hoc on-demand distance vector routing protocol*, in Mobile Computing and Networking, 1999, pp. 207–218.
[20] P. SINHA, R. SIVAKUMAR, AND V. BHARGHAVAN, MCEDAR*: Multicast core extraction distributed ad-hoc routing*, in Proc. of the Wireless Communications and Networking Conference, 1999.
[21] K. TANG, K. OBRACZKA, S. LEE, AND M. GERLA, *Reliable adaptive lightweight multicast protocol*, in IEEE International Conference on Communications (ICC 2003), 2003.
[22] J. XIE, R. R. TALPADE, A. MCCAULEY, , AND M. LIU, AMR*oute: Ad hoc multicast routing protocol*, ACM Mobile Networks and Applicaions, 7. No. 6 (2002).