MAC layer Multicast in Wireless Multihop Networks

Shweta Jain and Samir R. Das
State University of New York at Stony Brook

Abstract

Many applications in Wireless Ad-hoc networks require multicast communication. Unlike broadcast, multicast communication requires route discovery methods to build the multicast tree. Various multicast routing algorithm have been designed in recent years to facilitate multicast communication. There has been some work to develop a suitable MAC protocol for multicast traffic. In this work we explore some approaches for reliable multicast at the MAC layer. We develop a multicast extension of IEEE 802.11 standard and compare performance with 802.11 and some previous works. We implement the protocol in the popular *ns-2* simulator and experiment with multicast routing protocol.[1]. Our approach demonstrates superior performance in terms of *Packet delivery fraction* as well as *delay* compared to the original standard and certain variations of the standard. We have evaluated the protocol performance both under ideal channel conditions as well as channel subject to multipath fading.

1 Introduction

Wireless Ad-hoc networks have various applications in military, conferences, sensor networks and emergency operations. Many applications need one to many (multicast) communication where the group leader needs to send information to all members of the group. This scenario can be very common in the military where the commander needs to coordinate the activities of troops and convey important orders and messages. There is no dearth of applications that require multicast communication. Unlike broadcast, multicast cannot be achieved by a network wide flood. For example, the multicast service provider in an area would like to send DATA to only its subscribers who have paid for the service. In order to facilitate such communication the network layer needs to discover routes to all multicast group members in the network.

Many routing protocols that support multicast route discovery and tree formation have been developed in recent years [3],[2],[7],[6],[9],[10]. [23]compares various multicast routing protocol performance. While the routing layer is responsible to discover routes to destinations, it is the responsibility of the MAC layer to ensure that the data is delivered to the destination. MAC layer should be designed to provide reliable transmission of DATA to the destination at every hop in a multihop network. This reliability is usually achieved by requesting positive acknowledgment from the receiver. If positive acknowledgment is not implemented at the MAC layer, a feedback or acknowledgment mechanism must be developed in the application if the application requires some reasonable guarantee for DATA delivery. This requires the application layer to buffer data locally until acknowledgments have been received. This technique would introduce more delay in data delivery.

Besides application layer multicast data, some routing protocols may also require reliable multicast of the routing packets. *On Demand Multicast Routing protocol (ODMRP)* is a well known multicast routing protocol. The efficiency of this protocol greatly depends upon the reliable delivery of *JOIN TABLES* between members of the multicast tree. However, these join tables are sent as broadcast packets by the MAC layer, hence packet delivery is not guaranteed. An earlier work [20] has tried to achieve this reliability by passive acknowledgments. Passive acknowledgment is achieved by listening for retransmission by the neighbor nodes and transmitting the packet again if no such retransmission is heard. This method requires that each node should be able to receive retransmissions by all its neighbors which is again not guaranteed because of collisions due to simultaneous transmissions by multiple neighbors.

In order to improve the network efficiency, the MAC layer in wireless networks implements positive ACK and retransmission policies for unicast data transmission. There is an increasing research interest in designing reliable MAC protocols for multicast data as well. IEEE 802.11 is the most widely accepted MAC layer standard for both wireless LAN as well as ad-hoc networks. This protocol provides reliable delivery of unicast DATA and implements a retransmission policy in case of transmission failure. However, no such policy is implemented for broadcast and multicast data. Unlike unicast packets, multicast packets are not protected

[1]*Although we chose multicast AODV to test our protocol, our approach can be used in conjunction with any other multicast routing protocol.

by additional mechanisms like virtual carrier sensing and RTS/CTS exchange. MAC layer acknowledgment is also not supported for multicast packets. Wireless multihop network suffers from *hidden and exposed terminal* problems. The technique of virtual carrier sensing and control packet exchange alleviate the *hidden terminal* problem. This technique has been useful in increasing the reliability of unicast communication. There is a need to implement similar technique for multicast communication. Several previous works have tried to implement such techniques to provide both reliable broadcast as well as reliable multicast communication in the MAC layer. We will explore some of the recent works and propose our own solution.

The rest of the paper is organized as follows. In Section 2 we describe the medium access mechanism in IEEE 802.11 for unicast and multicast communication. Then in section 3 we describe some recent work that propose reliable MAC layer protocols for multicast and/or broadcast traffic. Section 4 describes our protocol and the design of multicast routing protocol to support multicast mac. We show performance analysis and results in section 5 followed by conclusion and future works in section 6.

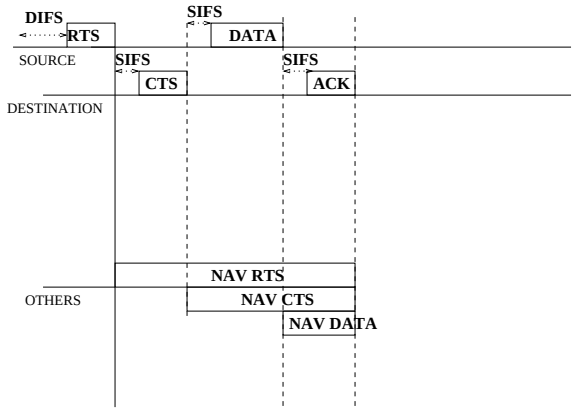

Figure 1: RTS/CTS Exchange in IEEE 802.11 for unicast transmission

2 IEEE 802.11 DCF

In this section, we will briefly review the IEEE 802.11 distributed coordinate function (DCF), which is the basic media access protocol used for unicast communication. This protocol uses Carrier Sensing Multiple Access with Collision Avoidance (CSMA/CA) to facilitate medium sharing between contending transmitters. Carrier sensing is performed by both physical and virtual mechanisms. The virtual carrier sensing is achieved by transmitting control frames to reserve the medium prior to transmission of unicast data packets. Fig 1 illustrates the 802.11 DCF mechanism. The transmitter, after sensing the medium to be idle, sends a *Request To Send* (RTS) frame with the transmitter and receiver addresses and the duration for which the medium is to be reserved. Any node other than the receiver, who hears the RTS, sets its network allocation vector(NAV) up-to the time period mentioned in the RTS, which is equal to the time required to transmit a *Clear To Send* (CTS) frame, a DATA packet, an *Acknowledgment* ACK and an additional duration equal to $3 \times SIFS$ (small inter-frame space) time. When the intended receiver receives the RTS frame, and senses the medium to be free, it replies with a CTS after waiting for an SIFS period. Any node other than the transmitter, who hears the CTS and had not heard the RTS before, would set its NAV to the time period mentioned in the CTS, which is equal to the time required to send a DATA packet, an ACK and an additional duration equal to $2 \times SIFS$. The successful reception of CTS by the transmitter indicates that the medium has been reserved and it can now transmit DATA. The transmitter waits for an SIFS duration and transmits DATA. Any node that did not receive the RTS/CTS correctly, because it was beyond the transmission ranges of both the receiver and the transmitter,but was able to sense the medium to be busy because it was within the carrier sensing range, would set its NAV to EIFS duration (extended Inter-frame Space). On receiving DATA successfully, the receiver waits for an SIFS duration and replies with an ACK. When the transmitter receives the ACK it knows that the DATA was successfully delivered. This access mechanism is used for Unicast data in IEEE 802.11. We will now describe the access mechanism for multicast and broadcast data transmission.

When the MAC layer receives a data packet with multicast or broadcast address, it uses CSMA/CA to transmit the packet. Fig 2 illustrates the access mechanism for multicast and broadcast traffic. The transmitter first senses the medium. If the medium is free it defers for DIFS (distributed inter-frame space) duration and if the medium is free until the end of the DIFS interval, the transmitter starts a back-off counter. The value of this back-off counter varies from 0 to Contention windows (CW) size. The value of CW varies from 32 to 1024 and depends upon the recent history of transmissions by the transmitter. If the transmitter had been unsuccessful in transmitting a unicast packet i.e, it has suffered collisions, its CW value will be larger. The CW size is doubled with every failure and is reset to 32 on a single success. The back-off counter value is a random number between 0 and CW. During the back-off interval the transmitter continues to sense the medium. If the medium becomes busy during the back-off interval, the transmitter freezes its back-off counter until the medium becomes free again, after which the countdown is started from

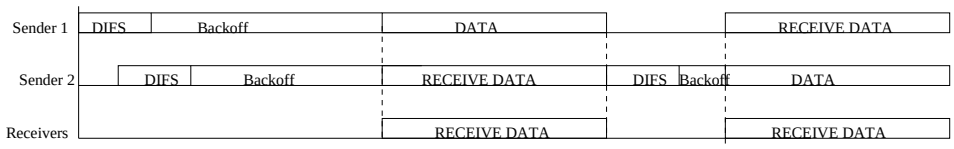

Figure 2: Access mechanism for multicast and broadcast transmission in IEEE 802.11

where it was frozen. If the medium remains free and the back-off counter becomes 0, the transmitter sends the DATA packet. All receivers that detect the transmitted packet correctly and are not in a busy state would receive the DATA packet and send it up to the routing layer. The routing layer may decide that the packet needs to be forwarded if the node is a router in the multicast tree. This station would then use the same access mechanism to forward the packet. This mechanism does not provide protection from hidden terminals neither does it guarantee that DATA was received correctly by all intended receivers. This unreliability may cause performance degradation for the application layer. Thus there is a need for a reliable MAC protocol for multicast service.

3 Background and Related Work

Some recent works have explored MAC protocols for reliable multicast and broadcast. [19],[5],[22] present solution requiring the use of busy tones and control packet exchange to achieve reliability and solution to *hidden terminal problems*. These protocols require additional hardware to send busy tones which might not be economical in real life. In [16] the authors have done an analysis of various MAC layer multicast approaches.

The Broadcast Support Medium Access (BSMA) protocol [12] is one of the first works that employ exchange of control packets to provide reliable MAC layer broadcast. Before sending DATA, the sender transmits an RTS frame and waits for CTS from all receivers, which are sent simultaneously causing collision at the sender. This protocol requires the use of a Direct Sequence Spread spectrum receiver with capture capability and assumes that the simultaneous signals can be captured by the DSSS radio. The protocol tries to avoid hidden terminal problem through this approach. Even with the availability of such radios, the receiver can capture colliding packets with a very low probability as analyzed in [15].

The protocol in [11] uses a similar approach without assuming a DSSS radio. In this work, the senders and receivers consider that a collision after RTS transmission is due to multiple CTS frames and the sender continues to transmit DATA. There is no ACK transmission, thus this approach does not provide reliable delivery. Apart from unreliable transmission, the assumption of collision is unrealistic in a dense medium where the collision may be due to another transmission and not due to CTS frames sent simultaneously.

Batch mode multicast MAC [15] is another protocol that employs control packet exchange to alleviate hidden terminal problems and achieve reliable transmission. In this protocol, the transmitter does an RTS/CTS exchange with all the next hop multicast receivers before Data transmission which is followed by a round of *Request For ACK (RAK)* and ACK transmissions. This requires the senders and receivers to reserve the medium for a relatively long interval of time $N \times (T_{RTS} + SIFSDuration + T_{CTS} + SIFSDuration) + T_{DATA} + SIFSDuration + N \times (T_{RAK} + SIFSDuration + T_{ACK} + SIFSDuration)$, where N = number of next hop multicast receivers. This approach does not fully utilize the broadcast nature of the broadcast medium, leading is wasted bandwidth. Similarly, Broadcast medium window (BMW) [13] achieves reliable broadcast by sending the broadcast packet as unicast packets to each neighbor in a round robin fashion while allowing other neighbors to receive the DATA sent without requiring acknowledgment. The sender transmits an RTS to the chosen neighbor and the neighbor responds with a CTS. The CTS contains the sequence numbers of packets that could not be received. The sender retransmits the missing packets as well as the current packet employing RTS/CTS/DATA/ACK exchanges. All other nodes may receive the packets and update their list of received data. The sender then transmits an RTS to the next neighbor and repeats this process. This approach achieves reliability but increases the data delivery latency because each neighbor needs to wait for its turn to request missing DATA from the sender.

In [21] broadcast packets are acknowledged by receivers via a bit sequence during a DIFS interval after DATA transmissions. This interval is divided into mini-slots and each receiver randomly selects a mini-slot to send a bit sequence. The size of the bit sequence depends upon the channel condition, thus limiting the number of receivers which can respond during the interval. At the end of the interval DATA is retransmitted to those nodes which did not send acknowledgment. This scheme provides reliability at a lower bandwidth cost but does not provide protection from *hidden termi-*

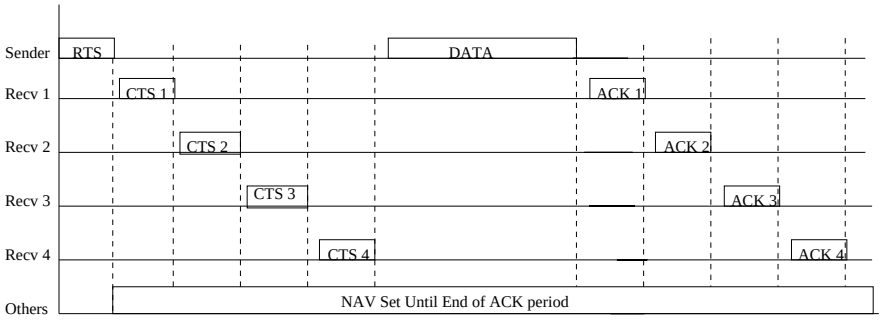

Figure 3: Multicast extension to 802.11 protocol.

nals, moreover, if the number of next hop neighbors is large,the probability of more than one node selecting the same mini-slot increases causing unnecessary retransmission of DATA.

In [8], the authors present an extension of IEEE 802.11 protocol called Multicast MAC (MMAC). In this work, the sender transmits multicast data packet to the next hop neighbors and waits to receive acknowledgments. The acknowledgment are sent according to a schedule calculated from the position index of the next hop address in the data packet. There is no upper bound to the number of next hop addresses that may be included into the data packet. Thus the data packet size increases by the number of addresses included in the header. The amount of time the sender has to wait before all the ACK frames have been received is $N \times (T_{ACK} + SIFSDuration)$, where N is the number of next hop receivers. At 2Mbps data rate $T_{ACK} = 56\mu sec$, $SIFSDuration = 10\mu sec$. Thus, for $N = 8$, the wait time is 528 μsec. If in the meantime a mobile node happens to enter the senders collision domain, it would sense an idle medium and might initiate a new data transmission. Apart from a straying node, those nodes which are beyond the receiving range but in the carrier sensing range of the sender will also be free to contend for the channel after an EIFS duration which is equal to $SIFSDuration + 8 \times ACK + DIFSDuration = 508\mu sec$ (DIFSDuration = $50\mu sec$). From these calculations it is clear that for $N \geq 8$, there is a possibility of ACK collisions at the sender leading to retransmission attempts by the sender. On the other hand, it is possible that while the receiver is waiting for its turn to send ACK, another node is trying to transmit DATA to the receiver. The receiver will not respond to any DATA transmissions before the ACK timeout period. This may cause the sender to retry several times leading to an increased contention window size and in extreme cases dropping the packet and initiating route error and discovery processes even though the route actually exists. If the sender does not receive ACK frames from all receivers, the protocol assumes that the loss is due to hidden terminals and it employs a loss recovery method.

The loss recovery method used in MMAC is similar to multicast scheme use in MACAM [14]. In both approaches the sender sends a single multicast RTS frame to all the neighbors and waits for CTS frames. The RTS frame is overloaded to contain the addresses of all the multicast next hop neighbors. Thus the RTS frame size is larger than the size of the frame in IEEE 802.11. CTS frames are transmitted in a time based priority schedule like the ACK frames. In both proocols there is no upper bound on the number of next hops that can be included in the RTS frame. Thus the RTS packet is larger than the packet size in 802.11 making the RTS frame itself prone to collisions due to hidden terminals. The effect of increased RTS size is not evaluated in these papers. Another shortcoming of these approaches is that it relies upon the fact that each next hop receiver is able to correctly receive the CTS frames sent by other receivers. Although the papers do not mention such a requirement but this requirement naturally arises due to broadcast nature of the wireless medium. The 802.11 standard requires that a node should perform Carrier sensing before sending CTS frames. If all next hop neighbors lie within each others receive range, they can receive and ignore CTS frames meant for the same multicast source. It is possible to construct scenarios in which two subsets of receivers lie beyond each others receive range (Fig 4). CTS frame sent by a receiver in one subset will not be correctly received by those in the second subset. In this scenario, some multicast receivers act as exposed nodes for other receivers. Such receivers incorrectly assume a busy medium and thus cancel CTS transmissions. The sender will send DATA to only those receivers which responded to the RTS frame and will need to retry for the remaining receivers. Fig 4 illustrates this scenario. Here, receivers 1 and 2 are beyond each others transmission range but within the carrier sensing range. When receiver 1 transmits CTS, receiver 2 would sense a busy medium and defer transmission by an EIFS period. Receiver 3 would hear the CTS correctly and determine that

the CTS was meant for the same multicast source and will go ahead and send the CTS itself. The sender will send DATA to receivers 1 and 3 alone and retry transmission for receiver 2. While this scheme is still reliable, the network incurs an extra wait period due to the unnecessary inclusion of receiver 2 in the RTS frame.

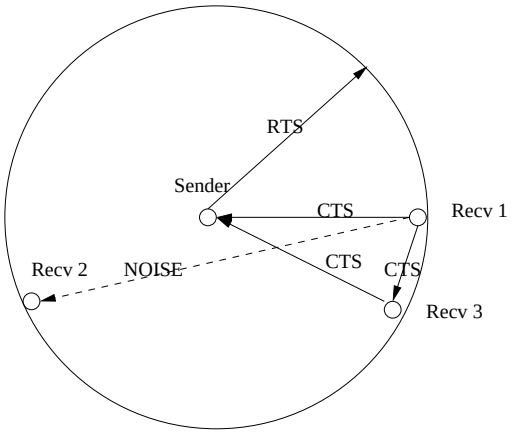

Figure 4: Neighbor unable to respond due to interference with CTS sent by another neighbor

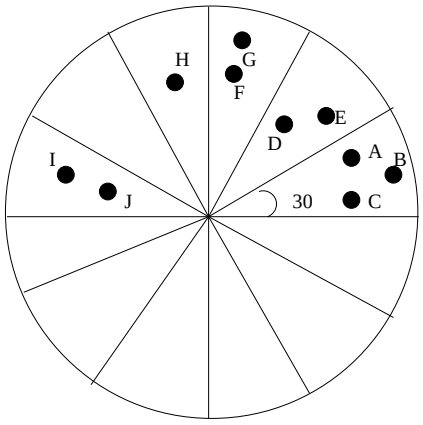

Figure 5: Clustering to group together non conflicting multicast next hop neighbors.

4 Reliable Multicast support for upper layers

We have developed MAC layer multicast as an extension to the IEEE 802.11 DCF protocol which can be used with any multicast routing protocol. We have tested our protocol with multicast AODV [3] but the scope of our protocol is not limited to any particular routing protocol. In this section we will first describe our MAC layer approach to provide reliable multicast followed by the design changes made to the multicast routing protocol.

4.1 Multicast extension for IEEE 802.11

We have implemented reliable multicast MAC within the IEEE 802.11 framework. We have used a similar approach in a previous work [18] but with a different goal of MAC layer "anycast" to achieve path diversity and thereby combat fading and adverse channel conditions. Before we describe the protocol we will describe the changes introduced to the MAC layer frames.

We modify the RTS frame to include at-most four multicast next hop neighbor addresses as in [18].This design choice helps us keep the RTS frame size within bounds. In 802.11 Wireless LAN standard, the MAC header may contain four addresses. This address space is used to specify source address, destination address, access point address etc. We can use this space to fit in four addresses for next hop nodes. These addresses are obtained from the routing layer and the position index of the address is the priority order of the neighbors. This priority may be determined by the routing layer. The CTS frame is modified to include the receiver's (node that sends the CTS in this case) order as determined from the RTS frame. This helps the original sender to differentiate between multiple CTS (CTS and ACK frames do not carry the source address). The sender maintains a log of received CTS frames. The DATA packet header is modified to include the addresses of all those nodes from which CTS was successfully received. Finally ACK frames are modified to include the receiver's order determined from the position index of its address in the received DATA frame. The sender keeps track of missing CTS and ACK frames and uses this knowledge to retransmit the data packet to only those neighbors who did not respond with CTS and ACK frames. Henceforth, we will refer to the modified control and DATA frames as RTSExt, CTSExt, DataExt and ACKExt.

When the MAC layer receives a multicast DATA packet from the upper layer it first senses the medium to determine if there is any ongoing transmission. If the medium is free, the transmitting node defers for a Distributed Inter-frame space duration (DIFS), if the medium is free during this time, the node starts counting down a back-off timer whose value is selected randomly from 0 to CW. While the node is counting down the back-off timer, it also continually senses the medium. If the medium becomes busy at any time before the timer expires, the node freezes the timer until the medium becomes free again. When the medium becomes free, the node starts counting down again after deferring for a DIFS period. This process is known as CSMA/CA and is a basic access mechanism in IEEE 802.11 MAC. When the back-off timer reaches 0 and the medium is still free, the transmitter can transmit the RTSExt frame to the multicast neighbors. The receiving nodes determine the time that they should wait before sending their

CTSExt frames from the position index of their address in the RTSExt frame. This wait time is equal to $N \times SIFSDuration + (N-1) \times CTSDuration$, where N is the position index of the address in the RTSExt frame. Thus the first node waits for SIFSDuration and the 4th one waits for $4 \times SIFSDuration + 3 \times CTSDuration$ before transmitting the CTSExt. A node transmits the CTSExt only if it does not hear any other transmission that could potentially interfere with the DATA packet that it will receive next. In our protocol, the nodes listen for CTSExt from other nodes. If the overheard CTSExt are destined for the same DATA source, it does not consider that transmission to be a potential threat to the DATA packet and continues to send its own CTSExt. The transmitting node records the addresses of nodes from which it correctly received CTSExt and sends DATAExt to only those nodes. Each node waits for its turn for sending ACKExt. The wait time in this case is $N \times SIFSDuration + (N-1) \times ACKDuration$, where N is the position index of the node address in the DATA packet. If the sender does not receive some ACKExt or CTSExt, it resends the DATA after appropriate back-off mechanism and RTSExt/CTSExt exchange with those nodes. Fig 3 illustrates the multicast extension proposed here.

Until now we explained how the protocol would act when there are 4 or less next hop neighbors. But as the multicast tree increases in size, the number of next hop children nodes in the multicast tree also increases. In case there are more than 4 children nodes, the transmitter needs to cluster the next hop neighbors into different groups of size at-most 4 each and transmit DATA to one group at a time. We will explain the clustering method next.

We have observed in previous works [8] and [14] that some next hop neighbors might act as exposed nodes to others i.e one node does not transmit its CTSExt frame if the other does. This happens if the power of the incoming CTSExt frame is less than the receive threshold at the receiver but greater than the carrier sensing threshold. In this case the receiver perceives this frame as noise and defers any transmission for Extended Interframe space time (EIFS). We try to alleviate this problem by clustering the neighbors into groups that are within each others transmission range and hence can overhear CTSExt from each other. In order to cluster nodes in this fashion we need to know the position of each node in the neighborhood. Another way to achieve the same result is by exchanging neighbor list with all neighbors and group together nodes that are each others neighbors. Position information may be available with the use of Global position system (GPS). Another method to get position information is by calculating angle of arrival and distance from the received signal strength. Any of these methods can be effectively used to form these clusters. Our method does not require the exact location information. We only need to know the approximate direction of the neighbors with respect to the transmitter. With this knowledge and from the knowledge of simple geometry we can claim that neighbors that lie within the same quadrant of the circle that approximately defines transmission range of the transmitter are each others neighbors. We re-order the neighbor list obtained from the routing table to group together nodes that are each others neighbors. In fig 3, Nodes A through G are within each others transmission range but since we cannot have groups larger than 4 we choose A through D to form the first group. Nodes E through H should form the second group while nodes I and J should form the third group. Thus the routing layer sends three copies of the same packet to the MAC layer with non-conflicting destination addresses. By clustering the next hops in this manner we ensure that given that the channel is free at all the receivers, the receivers will transmit CTSExt in response to the RTSExt. This reduces the time wasted in the RTSExt/CTSExt exchange due to inability of a node to recognize a prior CTSExt.

4.2 Design of Multicast routing

Ad Hoc On Demand Distance Vector Routing (AODV) [4] is a famous routing protocol for mobile ad-hoc networks. AODV is capable of both unicast and multicast routing. Multicast routing capabilities was incorporated into AODV in [3]. In order to run multicast algorithm, AODV is modified to include some additional data structures and routing messages. We will describe these changes below.

Each node running multicast AODV maintains a *Multicast Route Table* and a *Request Table* in addition to a *Route Table* . Members of multicast trees maintain routes to the multicast Group leaders in the Multicast Route Table. When a node hears a *Route Request*, it records the Multicast group IP address and Requesting Node IP addresses in the Request table. The requesting node typically becomes the multicast Group leader. If a node later wishes to join a multicast group and has an entry for the group leader in its Request table, it can unicast the request to join the group instead of sending a broadcast. Route Request (RREQ) packets carry two additional fields – J-flag and R-flag (join and repair flags, respectively). These flags are used for sending multicast join or route repair requests respectively. Similarly Route Reply (RREP) packets contain R-flag and U-flag (repair and update flags) to send route repair and update replies. Nodes use hello messages to maintain connectivity with neighboring nodes. Multicast group leaders periodically broadcast the *Group Hello* message to maintain the multicast group sequence number and for

disseminating this number to the multicast group. Group Hello message is also used to update distance from the group leader and for merging partitioned multicast trees.

The multicast algorithm uses a new message called *Multicast Activation* (MACT) message in addition to the RREQ and RREP messages described above. When a source node broadcasts a RREQ message for a multicast group, it often receives multiple replies. Unlike AODV, the node does not discard all replies received after the first reply. Instead, the requesting node waits for *rte discovery timeout* period after sending the RREQ. Within this timeout period, the node keeps routes with the greatest sequence number and the shortest number of hops to the nearest member of the multicast tree and disregards other routes. At the end of the timeout period, the node sends a unicast MACT message to this selected next hop. If the next hop is not a member of the multicast tree, it forward the MACT message to the best next hop. This process continues until the MACT message has reached a multicast group member. All other nodes that took part in the RREQ and RREP process delete the entry for the requesting node if they did not receive the MACT message. The MACT message thus ensures that there is only a single path to any tree node. The MACT messages are also used to prune the multicast tree. When a node decides to terminate its multicast group membership, it sends an MACT message with the P-flag (prune flag) set. The next hop upon receiving this message removes all information of the sending node from its multicast route table. If this node is not a member of the multicast group, it forward the message to the next hop. This ensures that all leaf nodes in the multicast group are multicast group members.

We have introduced certain modifications in the routing protocol to support the functioning of our MAC protocol. These modifications are not specific to multicast AODV and can be used with any other multicast routing protocol. Unlike 802.11 our MAC protocol requires the addresses of all the next hop multicast receivers. The original routing protocol does not support this functionality. The routing layer looks for next hop neighbors for the multicast group to which the packet needs to be sent. If there is a route available it forward the packet to the MAC layer. The packet header contains the address of the multicast group to which the DATA needs to be delivered. The routing layer of receiving node checks the group address in the packet. If it is a part of the multicast tree or is a router, it accepts the packet and forward it to the next neighbor. Since our MAC protocol requires the addresses of all neighbors, we modify the routing layer to include these addresses in the packet header before sending it down to the MAC layer. The routing table already contains entries for the next hop multicast neighbors. Thus by a simple lookup of the table, the routing layer is able to provide this information to the MAC layer. This modification is simple and can be implemented to any multicast routing algorithm, provided the algorithm already stores the next hop addresses in its tables.

We have made certain changes to evaluate the MAC protocols without interference from the routing protocol. These changes are not essential for the functioning of the MAC protocols. Since we have evaluated the protocol in static scenarios, we disable the propagation of route error messages. We conjecture that there is no possibility of route failure in a static scenario where all nodes are always on. Thus we disable feedback from the link layer to the routing layer to delete routes when MAC layer fails to deliver a packet after the given number of attempts. Such failure notifications are unnecessary as routes cannot be lost since the nodes are immobile and always on. These changes help us evaluate the MAC protocol fairly without interference from the routing layer.

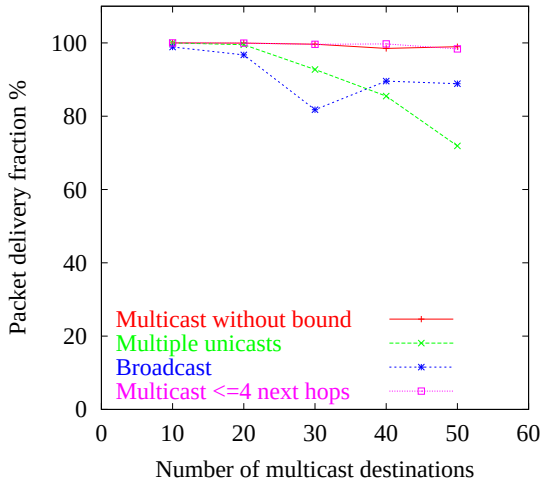

Figure 6: Packet Delivery Fraction with a two ray ground propagation model with 100 nodes.

5 Performance Evaluation

We have used network simulator *ns-2.26* to implement the multicast MAC protocol. In this section we will describe the experimental setup and results obtained. We will also briefly explain the protocols used to compare performance.

5.1 Experimental setup

We have implemented four different approaches to provide multicast at the MAC layer. We have seen earlier that there are various implementations for multicast

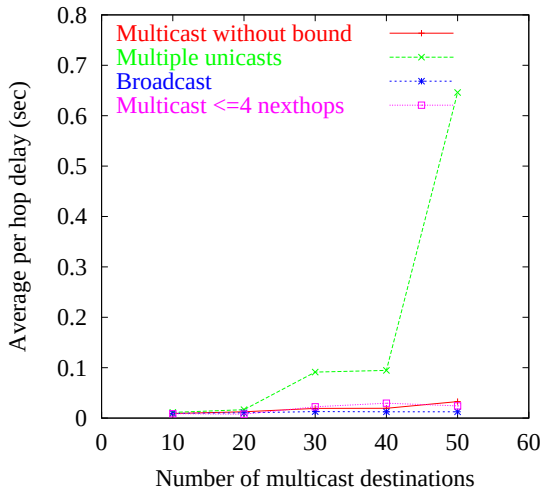

Figure 7: Average Per Hop Delay with a two ray ground propagation model with 100 nodes.

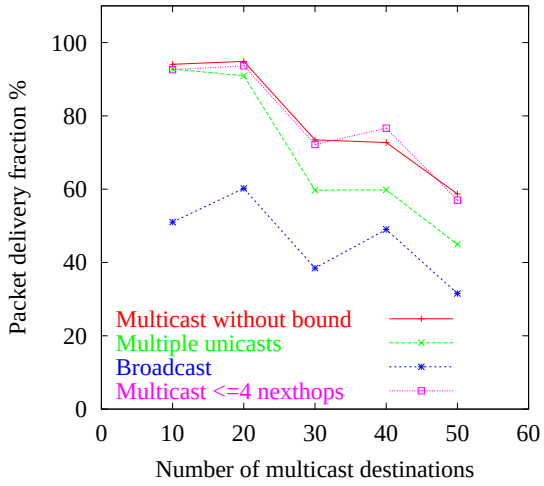

Figure 8: Packet Delivery Fraction with a Ricean fading propagation model with 100 nodes.

MAC. One method is to simply broadcast the packet after performing CSMA/CA without concerns for reliability and retransmission policy. The receivers would filter the packets depending upon the Multicast address associated with the packet. If the receiver belongs to the multicast group, it will accept the packet and re-broadcast it if it is a forwarding node. This method does not provide any reliability but is commonly used for multicast transmission under IEEE 802.11 MAC. We will refer to this method as the Broadcast MAC. One method to achieve reliable MAC layer multicast is to treat a single multicast packet as N unicast packets where N is the number of next hop neighbors that belong to the multicast tree. Each unicast packet is then transmitted using CSMA/CA with virtual carrier sensing and RTS/CTS exchange as enumerated in the IEEE 802.11 standard. This method provides reliability but it also brings about a larger delay in packet delivery. We can also send multicast packets using RTS/CTS exchange with multiple next hop addresses in RTS and DATA packets as used in MACAM and [14],MMAC [8]. This method also provides reliable transmission. Inclusion of multiple addresses in the RTS frame increases the size of the RTS frame. This is not practical in the real scenario. Control frames in IEEE 802.11 are kept small in size so that they are themselves not prone to collisions. In our implementation of MMAC / MACAM we have artificially reduced the size of the RTS frame so that it is the same size as the original RTS frame in IEEE 802.11. This is done to make a fair comparison with our protocol. In our approach we group the multicast neighbors into groups such that members of the same group are able to overhear CTSExt sent by one another. We have implemented all four methods and evaluated their performance against one another.

We have set up the experiment using a grid of size 1500x300 with 100 nodes. There is one multicast source with different number of destinations (10,20,30,40 and 50). The source sends 4 multicast UDP packets per second. The simulation runs for 900s. We use 2Mbps data rate and a nominal transmission range of 250m with the carrier sensing range of 500m. We use the two ray ground propagation model in the physical layer in one set of experiments and a *Ricean fading model* from [17] for another set of experiments to motivate the importance of using reliable MAC layer multicast. The same physical layer model was used in [1] and [18].

5.2 Results

We instrument the experiments to calculate the *Packet Delivery Fraction* and *average per hop delay* in the network. The *Packet Delivery Fraction* is calculated as

$$\frac{\text{No. of packets delivered}}{\text{No. of multicast destinations}}.$$

Similarly the *average per hop delay* is calculated as

$$\frac{\text{per packet delay}}{\text{Hop count between source and destination of the packet}}.$$

Experimental results clearly show that in the absence of a reliable MAC protocol, the network suffers from a large amount of packet loss. These losses are mainly due to collisions with other packets in the network. Fig 6 shows the *Packet Delivery Fraction* achieved with various MAC protocols. Due to the absence of a retransmission policy the broadcast MAC protocol is able to deliver only 88 % of DATA when the number if multicast members increases to 50. Unicast MAC also shows

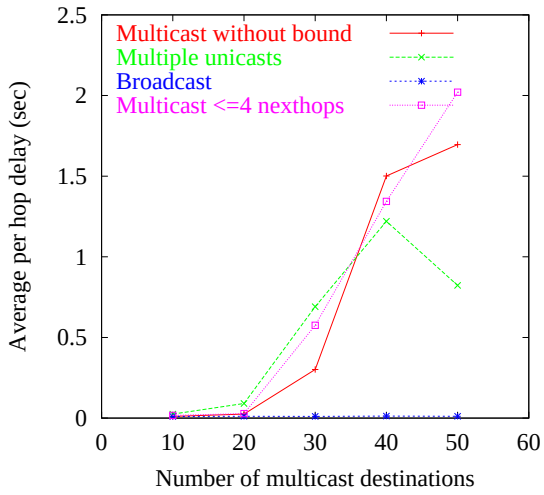

Figure 9: Average Per Hop Delay with a Ricean fading propagation model with 100 nodes.

poor performance although it ensures reliable delivery through ACK and retransmission policy. The delay incurred in sending multiple unicast packets to all multicast receivers causes packet loss in the interface queues. Multiple unicasts also contributes to the increase of the overall network load since a single multicast packet is treated as N unicast packets, N being the number of next hop receivers. The load is further increased due to contention with the receivers that are routers themselves. The *Packet Delivery Fraction* reduces to 71 % for 50 multicast members which is lower than that acheived with the broadcast mac protocol. Both reliable multicast schemes provide achieve higher *Packet Delivery Fraction* compare to the other schemes due to their ability to utilise the network bandwidth more optimaly and proper exploitation of the broadcast nature of the wireless medium. Both the protocols provide a high *Packet Delivery Fraction* of 98 % for even large number of multicast members.

Fig 8 shows the *Packet Delivery Fraction* in the presence of Ricean fading model in the physical layer. The comparative performance of all the protocols is similar in this case except for the fall in the absolute performance. We performed these experiments to further motivate the importance of reliable MAC. Here we observe that broadcast MAC which performed better than unicast MAC in the two ray ground propagation model, actually performs much worse with the fading model. The main reason for this degradation is the absence of retransmission policies which becomes of more importance in adverse channel condition.

Fig 7 plots the average per hop delay incurred in each of the four situations in the two ray ground propagation model. We observe that broadcast mac achieves the least delay which is again due to the absence of any loss recovery mechanism. Unicast MAC incurs the maximum delay mainly due to the long time packets stay in the interface queues before they can be transmitted. This queuing delay is a direct result of increased network load owing to the multiple copies of the multicast packets in the network. The multicast MAC protocols incur very low delay compared to the Unicast MAC protocol.

Fig 9 plots average per hop delay incurred in the Ricean fading scenarios. We observe that all reliable protocols incur much higher delay than in the two ray ground model. This is due to the increase in the number of retransmissions required to recover from losses due to adverse channel conditions. Multicast MAC protocols incur higher delay in these scenarios than the other protocols due to statistical reasons. It is possible that some packets that arrive at a node may be dropped from the interface queue. The probability of such drops increases when the packet has incurred high delay. These dropped packets do not contribute to the delay calculations in our experiments. The exclusion of such high delay packets attributes for lower delay in unicast MAC protocol as compared to the multicast protocols. Low delay in broadcast is again due to the absence of retransmission policies.

6 Conclusion and Future Directions

We have presented a simple extension to IEEE 802.11 protocol to provide reliable multicast MAC protocol. This approach can be easily incorporated in the standard to provide performance enhancement of multicast communication. Further work in this direction is required to implement this concept in a testbed scenario. In future, we will implement this protocol in a testbed using Berkeley motes similar to the one used in [18] to provide a proof of concept implementation.

References

[1] B. Sadeghi, V. Kanodia, A. Sabharwal and E. Knightly. Opportunistic Media Access for Multirate Ad Hoc Networks. In *Proceedings of the 8th International Conference on Mobile Computing and Networking (ACM MOBICOM'02)*, pages 24–35, September 2002.

[2] C. Chiang and M. Gerla. On-demand Multicast in Mobile Wireless Networks. In *Proc. IEEE ICNP '98*, 1998.

[3] Charles Perkins and Elizabeth Royer. Multicast Using Ad Hoc On-Demand Distance Vector

Routing. In *Proceedings of the 5th International Conference on Mobile Computing and Networking (ACM MOBICOM'99)*, August 1999.

[4] Charles Perkins and Elizabeth Royer and Samir R. Das. Ad Hoc On Demand Distance Vector (AODV) Routing. RFC 3561, July 2003.

[5] Chun-Yuah Chiu, E.H. Wu and Gen-Huey Chen. A Reliable and Efficient MAC layer Broadcast (Multicast) Protocol for Mobile Ad hoc Networks. In *Global Internet and Next Generation Networks, GlobeCom 2004*, pages 2802– 2807, December 2004.

[6] E. Madruga and J. Garcia-Luna-Aceves. Multicasting Along Meshes in Ad-Hoc Networks, 1999.

[7] Ewerton L. Madruga and J. J. Garcia-Luna-Aceves. Scalable Multicasting: The Core-Assisted Mesh Protocol. *ACM/Baltzer Mobile Networks and Applications, Special Issue on Management of Mobility*, 6(2):151–165, apr 2001.

[8] Hrishikesh Gossain and Nagesh Nandiraju and Kumar Anand and Dharma P. Agrawal. Supporting MAC Layer Multicast in IEEE 802.11 based MANETs: Issues and Solutions. In *29th Annual IEEE International Conference on Local Computer Networks (LCN'04)*, pages 172–179, 2004.

[9] J. Xie, R. Talpade, T. McAuley and M. Liu. AMRoute: Ad Hoc Multicast Routing Protocol. *ACM Mobile Networks and Applications (MONET) Journal, 7(6):*, pages 429–439, Dec 2002.

[10] Jorjeta G. Jetcheva and David B. Johnson. Adaptive Demand-Driven Multicast Routing protocol (ADMR). Internet Draft, draft- jetcheva-manet-admr-00.txt, work in progress, June 2001.

[11] K. Tang and M. Gerla. MAC Layer Broadcast Support in 802.11 Wireless Networks. In *MILCOM 2000. 21st Century Military Communications Conference Proceedings Volume 1*, Oct. 2000.

[12] K. Tang and M. Gerla. Random Access MAC for Efficient Broadcast Support in Ad hoc Networks. In *Wireless Communications and Networking Conference, 2000. WCNC. 2000 IEEE Volume 1*, pages 454 – 459, 2000.

[13] K. Tang and M. Gerla. MAC Reliable Broadcast in Ad Hoc Networks. In *Military Communications Conference, 2001. MILCOM 2001. Communications for Network-Centric Operations: Creating the Information Force. IEEE Volume 2*, Oct. 2001.

[14] Ki-Ho Lee and Dong-Ho Cho. A Multiple Access Collision Avoidance Protocol for Multicast Service in Mobile Ad Hoc Networks. In *Vehicular Technology Conference, 2003. VTC 2003-Spring. The 57th IEEE Semiannual Volume 3*, April 2003.

[15] Min-Te Sun, Lifei Huang and A. Arora and Ten-Hwang Lai. Reliable MAC Layer Multicast in IEEE 802.11 Wireless Networks. In *Parallel Processing, 2002. Proceedings. International Conference on*, Aug 2002.

[16] Prasanna Chaporkar and Saswati Sarkar. Wireless Multicast: Theory and Approaches. IEEE Transactions of Information Theory, Accepted for publications.

[17] R.J. Punnoose, P.V. Nikitin and D.D. Stancil. Efficient Simulation of Ricean Fading within a Packet Simulator. In *52nd IEEE VTS-Fall 2000*, volume 2, pages 764–767, September 2000.

[18] S Jain and S Das. Exploiting Path Diversity in the Link Layer in Wireless Ad hoc Networks. In *Proceedings of World of Wireless, Multimedia and Mobile networks, WoWMoM 2005*, 2005.

[19] S. K. S. Gupta, V. Shankar and S. Lalwani. Reliable Multicast MAC Protocol for Wireless LANs. pages 93–97, 2003.

[20] S. Lee, W. Su and M. Gerla. Ad hoc Wireless Multicast with Mobility Prediction, 1999.

[21] Shiann-Tsong Sheu, Yihjia Tsai and Jenhui Chen. A Highly Reliable Broadcast Scheme for IEEE 802.11 Multi-Hop Ad Hoc Networks. In *Communications, 2002. ICC 2002. IEEE International Conference on Volume 1*, April 2002.

[22] Weisheng Si and Chengzhi Li. RMAC: A Reliable Multicast MAC Protocol for Wireless Ad Hoc Networks. In *ICPP*, pages 494–501, 2004.

[23] Sung-Ju Lee, William Su, Julian Hsu, Mario Gerla and Rajive Bagrodia. A Performance Comparison Study of Ad Hoc Wireless Multicast Protocols. In *INFOCOM (2)*.