Reliable MAC Layer Multicast in IEEE 802.11 Wireless Networks

Min-Te Sun, Lifei Huang, Anish Arora, Ten-Hwang Lai
Department of Computer and Information Science
The Ohio State University
2015 Neil Avenue
Columbus, OH 43210-1277
{msun, huangli, anish, lai}@cis.ohio-state.edu

Abstract

Multicast/broadcast is an important service primitive in networks. The IEEE 802.11 multicast/broadcast protocol is based on the basic access procedure of Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA). This protocol does not provide any media access control (MAC) layer recovery on multicast/broadcast frames. As a result, the reliability of the multicast/broadcast service is reduced due to the increased probability of lost frames resulting from interference or collisions. In this paper, we propose a reliable **Batch Mode Multicast MAC** *protocol,* BMMM, *which substentially reduces the number of contention phases, thus considerably reduces the time required for a multicast/broadcast. We then propose a* **Location Aware Multicast MAC** *protocol,* LAMM, *that uses station location information to further improve upon* BMMM. *Extensive analysis and simulation results validate the reliability and efficiency of our multicast MAC protocols.*

1 Introduction

Media Access Control (MAC) remains a fundamental research problems in wireless networks, given the difficulties caused by transmission errors, collisions, and hidden nodes. These difficulties become even more severe when support is provided for multicast/broadcast communication in wireless networks. Such support is necessary for delivering acceptable quality of service in many applications of wireless communications, such as emergency reporting or video conferencing. Moreover, even in scenarios where applications themselves do not demand multicast/broadcast, several higher layer protocols rely heavily on reliable and efficient MAC layer multicast/broadcast, for instance DSR [8], AODV [16] and ZRP [7] routing protocols. It is important to note that multicast/broadcast in the MAC layer refers specifically to the process of sending a data frame to some/all of the *neighbors* of a node. Henceforth in our presentation we treat broadcast as a special case of multicast.

Of the many random access MAC protocols for wireless networks that have been proposed so far, most primarily target unicast communications and do not yield an efficient basis for simulating multicast [9, 6]. In the few that do deal directly with multicast [2, 3], it is apparent that reliability is not a major concern. For instance, in the IEEE 802.11 specification [2], the multicast sender simply listens to the channel and then transmits its data frame when the channel becomes free for a period of time. There is no MAC-level recovery on multicast frame. As a result, the reliability of multicast is reduced due to the increased probability of lost frames resulting from interference or collisions. As another example, in [3], it is simply suggested that the sender transmits a *Request To Send* (RTS) frame immediately followed by the data frame(s). This RTS frame informs the neighbors which are idle to yield their transmissions to somewhat reduce the chance of message collisions. Again, the reliability of this scheme is low.

Recently a few multicast MAC protocols [19, 20, 21] have been proposed to enhance the reliability and the efficiency of the 802.11 multicast protocol. In this paper, we observe that even these protocols have serious reliability and/or efficiency issues. We demonstrate a reliability issue in the first two protocols [19, 20], and show that while the third protocol [21] is logically reliable, it is inefficient and can easily lead to message timeouts. Further, towards redressing these reliability and efficiency issues, we design two multicast MAC protocols based on the IEEE 802.11 *Distributed Coordination Function* (DCF) MAC protocol.

Our first protocol, *Batch Mode Multicast MAC* (BMMM), reduces the number of contention phases from n to 1, where n is the number of intended receivers in a multicast. Essentially, it provides a simple coordination mechanism for avoiding collisions in the transmissions of *Clear To Send* (CTS) and *Acknowledge* (ACK) frames, and ensures that each time the data frame is transmitted, it is received by as many intended receivers as possible. Therefore, BMMM is able to substantially reduce the average total time to complete a multicast MAC request.

Our second protocol, *Location Aware Multicast MAC* (LAMM), uses location information to further improve BMMM. Let S denote the set of intended receivers of a multicast MAC request. We show how the successful transmission of data frame to all nodes in S' using BMMM, where $S' \subseteq S$, is enough to ensure the reception of data by all nodes in S without collision. Assuming that the transmission radius is constant, we provide a necessary and sufficient condition for S'. This significantly reduces the number of

RTS, CTS, RAK and ACK frames in the BMMM protocol. We note that since the US *Federal Communications Commission* (FCC) has requested all wireless service carriers to provide the location service of emergency 911 calls [1], soon each wireless device will be able to identify its own location by means of the geolocation techniques [5]. Indeed, location information has already been used in some routing protocols [10, 4, 22, 12, 15, 17]. But to the best of our knowledge, this paper is the first effort to utilize location information at the MAC layer in wireless networks.

Using the same control and data frame formats in IEEE 802.11 specification, our protocols are able to co-exist with the current unreliable IEEE 802.11 multicast MAC protocol to provide reliable multicast MAC services when needed. To validate the performance of BMMM and LAMM, we have both analyzed and simulated our protocols along with the BSMA protocol in [20] and the BMW protocol in [21]. Our results show that our protocols are substantially more reliable and efficient than the others.

2 Existing Multicast MAC Protocols

As mentioned in Section 1, one purpose of this paper is to report our observations of performances of existing multicast MAC protocols. To this end, we begin with a brief description of CSMA/CA and three other multicast MAC protocols. Note that in IEEE 802.11, the beacon containing the station MAC address is broadcast periodically by each station to announce its presence. A station knows the neighbor's MAC addresses through the exchanges of beacon signals. Each station in the network maintains a routing table containing both the next hop information to the destination and the members of each multicast group. When a multicast request arrives from the network layer, it is assumed that the request indicates the set of neighbors required to reach all the members of the intended multicast group.

2.1 Carrier Sense Multiple Access with Collision Avoidance

The idea of *Carrier Sense Multiple Access with Collision Avoidance* (CSMA/CA) proposed in [11] has been used in many wireless MAC protocols. It works as follows.

Carrier Sense Multiple Access with Collision Avoidance Protocol (CSMA/CA): 1. A node wishing to transmit first

listens to the medium.

2. If the medium is idle, transmit the frame.

3. If the medium is busy, then continue to listen until the medium is idle; then backoff for x slots of time, where x is a random number within the contention window.

(a) If the channel is still idle when the backoff timer expires, transmit the frame.

(b) If the channel becomes busy before the timer expires, stop the timer and listen to the channel again; when the channel is detected idle, restart the backoff timer.

4. After transmission, if the node does not receive an ACK, attempt to retransmit the frame.

5. After receiving a data frame, the receiver returns an ACK.

From the above protocol, we can see that CSMA/CA protocol assumes an Ethernet-like wireless environment and intends to avoid message collision by sensing the medium status at the sender side. When the medium is busy, the node backs off its frame transmission to avoid collision. The step 1, 2 and 3(b) of the CSMA/CA protocol is commonly referred to as the *contention phase*.

The CSMA/CA protocol is known to suffer from the *hidden terminal problem*. Assume stations p and q are within each other's transmission range, and so are stations q and r; but p and r cannot hear from each other. Suppose node p wants to transmit a frame to node q while q's neighbor, r, is transmitting. Using the CSMA/CA protocol, node p will find the medium idle and transmit the frame, causing collisions at q.

Proposed in [3, 6, 9], a common method to solve the hidden terminal problem is to extend the CSMA/CA protocol with a Request-To-Send/Clear-To-Send (RTS/CTS) handshake. Before transmitting a data frame, the sender transmits an RTS frame, with an indication of the amount of time needed. Upon receiving RTS, the receiver returns a CTS, also with an indication of the time needed. All other nodes that hear RTS and/or CTS must back off for the amount of time indicated in RTS/CTS to avoid collisions. This creates a virtual carrier sense at the receiver end and avoids the hidden terminal problem. The IEEE 802.11 *Distributed Coordination Function* (DCF) MAC protocol [2] is essentially the CSMA/CA protocol with the optional RTS/CTS extension.

2.2 Description of Existing Multicast Protocols

In IEEE 802.11, the RTS/CTS extension is not used for broadcast/multicast; and the receivers are not required to return an ACK. As a result, the quality of broadcast/multicast service is not as good as that of unicast.

The protocol in [19] attempts to extend the IEEE 802.11 broadcast/multicast protocol (i.e., the basic DCF MAC) with RTS/CTS handshaking. According to the protocol, when there is a broadcast data frame to send, the sender first executes the contention phase. Once obtaining access to the medium, the sender transmits an RTS frame to its neighbors and waits for CTS frames for WAIT_FOR_CTS time units. If a node receives an RTS frame when it is not in the YIELD phase, it sends back a CTS and then waits for the data frame for WAIT_FOR_DATA time units. If the sender receives any CTS frame before the WAIT_FOR_CTS timer expires, it transmits the data frame. If the sender does not receive any CTS frame before its WAIT_FOR_CTS timer expires, it backs off and enters the contention phase again to retransmit the broadcast data frame.

In [20], the *Broadcast Support Multiple Access* (BSMA) protocol augments the broadcast MAC protocol in [19] with the NAK frame and the following additional rules:

1. After the sender transmits a data frame, it waits for WAIT_FOR_NAK time units for any possible transmission problem reported by the neighboring nodes.

2. If a receiver does not receive the data frame after it transmitted the CTS frame for WAIT_FOR_DATA time units, it transmits a NAK frame.

3. If the sender does not receive any NAK frame before its WAIT_FOR_NAK timer expires, the broadcast service is complete. Otherwise, the sender backs off and enters the contention phase again to retransmit the broadcast data.

In [21], the *Broadcast Medium Window* (BMW) protocol is introduced to provide a reliable broadcast MAC. The basic idea of the BMW protocol is to treat each broadcast request as multiple unicast requests. Each unicast is processed using the reliable IEEE 802.11 DCF MAC protocol (i.e., CSMA/RTS/CTS/DATA/ACK) with some minor modifications. In BMW, each node maintains three lists: NEIGHBOR, SEND BUFFER and RECEIVER BUFFER. The NEIGHBOR list contains the current neighbors. The SEND BUFFER list contains the ongoing broadcast messages. The node's RECEIVE BUFFER list contains the sequence numbers of the data frames received by the node.

In BMW, when a node has a broadcast data to send, it first executes the contention phase. Afterwards, the sender places the message into its SEND BUFFER and sends out an RTS frame containing the sequence number of the upcoming data frame and the MAC address of the first node in its NEIGHBOR list. When a node receives a RTS intended for it, it checks its RECEIVE BUFFER list to see if it has received all the data frames with sequence number smaller than or equal to the upcoming one. If all the data frames (including the upcoming one) have been received, the receiver sends a CTS with appropriate information to suppress the sender's data frame transmission. Otherwise, the receiver sends a CTS frame with all the missing data frame sequence numbers. The sender, upon receiving the CTS frame from one of the neighbors, checks the CTS frame to transmit all the missing data frames and waits for an ACK. After receiving the data frame, the receiver sends an ACK. The sender moves on to serve the next node on the NEIGHBOR list if either the returned CTS frame indicates all the data frames have been received or an ACK has been received. If all nodes in the NEIGHBOR list have been served, the sender removes the message from its SEND BUFFER.

3 Problems with Existing Multicast MAC Protocols

In [19], the MAC protocol does not coordinate the transmission of the CTS frames. This brings about a serious problem: the collision of CTS frames. Since this protocol is an extension of the IEEE 802.11 DCF MAC, the intended receivers will each send a CTS frame immediately after a short inter-frame spacing (SIFS) if it is not in the YIELD phase. This SIFS is predefined and has the same value for all nodes. For a multicast RTS, if more than one intended receiver replies with a CTS frame, these CTS frames are destined to collide with each other at the sender. To take care of this problem, an assumption is made in [19] that the sender's radio has the *Direct Sequence* (DS) capture ability. That is, when a node receives multiple frames at the same time, the frame with the strongest power can be captured by the node. However, according to [3], in order to successfully capture the frame, the strongest frame's Signal-to-Interference Ratio (SIR) needs to be at least 10dB. If there are only two nodes sending CTS frames with the same power, this requires that one node is at least 1.5 times far away from the sender than the other node is. When more than two nodes are sending CTS frames simultaneously, this SIR is difficult to achieve. In [23], it is reported that when nodes are distributed uniformly, the "capture" effect occurs with a probability at about 0.55 when there are two competing nodes. This probability quickly drops to 0.3 at the presence of 5 nodes and then further drops to 0.2. Thus, in the protocol of [19], the sender is most likely to miss the CTS frames due to collision and the sender's low capture ability. According to the protocol, the sender will back off its transmission of data frame and, after another contention phase, send another RTS. The above process may repeat for several times until a CTS capture is successful or the message times out thus increasing the total time of multicast.

At a first glance, it seems easy to alleviate CTS collisions by allowing each intended receiver to defer its CTS transmission for a random amount of time. That is, instead of sending a CTS after a SIFS time gap, each receiver i defers for a random number, x_i, of time slots, where x_i is an integer within a contention window, say $[0..w]$. This random defer time scheme can reduce the chance of CTS collision at the sender, *if the value of w is not too small*. Unfortunately, as explained below, this is not easy to achieve. According to IEEE 802.11, after the medium is idle for a time equal to *DCF Inter-Frame Spacing* (DIFS), every node can contend for access to the wireless channel. Thus, to implement the above random defer time scheme, the value of w should be *less* than $\frac{DIFS-SIFS}{length\ of\ a\ slot}$ so as to ensure that each CRT is sent before any node has a chance to send any non-CRT frame.

According to the IEEE 802.11 specification, for the *Frequency Hopping Spread Spectrum* (FHSS) medium, the lengths of SIFS and DIFS are 28 and 128 microseconds, respectively, and each slot is 50 microseconds. Thus, the maximum value allowed for w is 1[1], which evidently is too small for the above random defer time scheme to work effectively. One may argue that we can increase the size of w by increasing the value of DIFS. That may be true, but it certainly is unacceptable because a longer DIFS value would considerably slow down all communications in all IEEE 802.11 networks.

[1] Actually, there is a value PIFS, defined to be 78 microsecond for FHSS systems. If a medium is idle for PIFS time, a specific node can send a beacon frame to switch to the point coordinated mode. If this feature is available in the wireless network in question, the only value available for w would be 0!

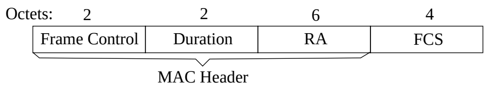

Figure 1. RAK Frame Format

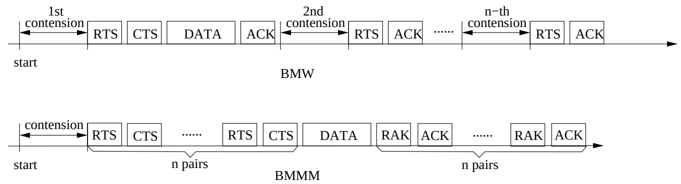

Figure 2. BMW vs BMMM

The BSMA protocol in [20] is essentially the same as the protocol in [19], except that it includes the NAK frame. That means that the BSMA protocol has exactly the same CTS collision problem. The additional NAK frame in [20] does not help resolve the collision of the CTS frame. In fact, since the transmissions of NAK frame are not coordinated either. The same collision problem exists when more than one node send the NAK frames.

The protocols in [19, 20], whether with or without NAK [19, 20], are unreliable in that when a multicast is done, they do not know whether every intended receiver has received the data. These protocols do not improve much in reliability over the current IEEE 802.11 multicast service. In the above sense, the BMW protocol in [21] is reliable because, if necessary, the sender will retransmit the data frame until it has received an ACK from every intended receiver. However, BMW is inefficient for following reasons:

- Contention phase — The BMW protocol requires at least n contention phases for each multicast data frame. Not only is each contention phase lengthy in time, but also the sender has to contend with other nodes for access to the medium. It is possible that some other node wins the contention and thereby interrupts and prolongs the ongoing multicast process.

- Timeout — In many applications (e.g., routing), multicast is time sensitive. If the multicast request can not be fulfilled within a certain amount time, the multicast request will be considered unsuccessful by the higher layer. For such applications, the prolonged multicast process as pointed out above can easily lead to a timeout in the higher layer.

4 Batch Mode Multicast MAC Protocol

In BMW, the sender uses at least n rounds of DCF-like unicasts for a multicast request intended for n neighboring nodes. Each round requires one contention phase before an RTS frame can be sent. If we consolidate the n contention phases into one, then the required time to serve a multicast can be greatly reduced. This is the primary idea of our *Batch Mode Multicast MAC Protocol* (BMMM).

To achieve this goal, the design issue is how to coordinate the transmissions of the control frames, including RTS, CTS and ACK, with no modification of the frame format in IEEE 802.11 specification. First, we want to ensure that there is no collision among control frame transmissions. Second, if one of the sender's neighbors has data to send, it should not pass its contention phase when the sender is exchanging control frames with its intended receivers. To avoid the collisions among CTS and ACK frames, the sender needs to provide a simple coordination among the intended receivers. To prevent a neighbor from passing its contention phase, the protocol needs to ensure the medium will not idle for long when a multicast request is processing. To meet the above requirements, we design the protocol such that the sender instructs its intended receivers (of a multicast) to transmit the control frame in order. The sender uses its RTS frames to sequentially instruct each intended receiver to transmit a CTS. To coordinate ACK transmissions from receivers, a new control frame is required. We therefore propose a new control frame type called RAK (Request for ACK). The RAK frame, as shown in Figure 1, has the same format as the ACK frame. It contains frame control, Duration, receiver address (RA) and frame check sequence (FCS). With the help of the RAK frame, the sender can coordinate the ACK transmissions in the similar manner it coordinates the CTS transmissions. That is, after the transmission of the data frame, the sender uses the RAK frames to sequentially instruct each intended receiver to transmit an ACK. Figure 2 illustrates the difference between BMW and BMMM when the data is received by all intended receivers successfully at the first try.

Our BMMM protocol has several advantages:

- Our protocol greatly reduces the number of contention phases, which will be shown by our analysis and simulation. The time decreased by the reduction of contention phases is much larger than the time increased by the introduction of RAK frames because the transmission of each RAK frame takes one time slot while one contention phase generally takes much more than one time slot. Therefore, our protocol significantly decreases the required time to serve a multicast request.

- Our multicast protocol does not modify any control frame format. This allows our multicast MAC protocol to co-exist with the other IEEE 802.11 protocols, including the unreliable IEEE 802.11 multicast MAC protocol. A multicast request can specify if it needs a reliable service or not from the upper layer to select the appropriate multicast MAC protocol to use.

- In our protocol, the sender transmits RTS frames periodically before sending data and transmits RAK frames periodically after sending the data. This means that the medium will never be idle for more than $2 \cdot SIFS + (T_{CTS} or T_{ACK})$, which is less than DIFS. Since any neighbor wishing to transmit data must listen to ensure the channel is free for at least DIFS, having sender transmitting RTS and RAK frames periodically prevents any neighbor from passing its contention phase.

Batch_Mode_Procedure(Input: S**, Output:** S_{ACK}**)**

s executes the contention phase
for each $p_i \in S$
  s sends RTS containing p_i's MAC address and
    Duration := $(\|S\| - i) \cdot T_{RTS} + (\|S\| - i + 1) \cdot T_{CTS} + T_{DATA} + \|S\| \cdot (T_{RAK} + T_{ACK})$
  s waits CTS from p_i for T_{CTS}
if s received at least one CTS frame
  s sends DATA frame
  for each $p_i \in S$
    s sends RAK containing p_i's MAC address
    s waits ACK from p_i for T_{ACK}
else /* no CTS was received */
  s backs off and starts the sender's protocol again
let $S_{ACK} \subseteq S$ be the set of nodes from which S has received an ACK

Sender's Protocol:

if s has a multicast message to send to the nodes in S and it is not in yield state
  while $S \neq \emptyset$
    Call Batch_Mode_Procedure(S, S_{ACK})
    $S = S \setminus S_{ACK}$

Receiver's Protocol:

if a node p receives an RTS frame
  if p's MAC address matches the RTS's receiver address and it is not in yield state
    p sends out CTS containing Duration := Duration in RTS - T_{CTS}
if a node p receives a RAK frame
  if p's MAC address matches the RAK's receiver's and p has received the data frame and it is not yielding
    p sends ACK containing Duration = Duration in RAK - T_{RAK}
if a node q receives a control frame (RTS/CTS/RAK/ACK) not intended for it
  q yields for Duration time specified in the control frame

Figure 3. The Batch Mode Multicast MAC Protocol (BMMM).

5 Location Aware Multicast MAC Protocol

We pointed out in Section 1 that a node's geographic location can be easily obtained from the Global Position System (GPS). Considering the transmission radius of the IEEE 802.11 (up to 500 feet for 802.11b), the GPS location information is accurate enough to be used for our purpose. In IEEE 802.11 specification, the frame body of the beacon frame format is well enough to accommodate the GPS location information (< 30 bits). If we including the location information in beacons, neighbors will learn each other's location. Location information has been used in some routing protocols. In this section, we investigate the possibility of utilizing location information in medium access control.

The BMMM protocol in Section 3 reduces the number of contention phases by putting $\|S\|$ pairs of RTS/CTS together. The nodes that successfully received the data frame are expected to each return an ACK after it receives a RAK. When the size of S is large, it may be desirable to reduce S's size by considering only a subset of it. That is, when running the BMMM protocol, we send RTS only to the addresses of nodes in a subset, S', of S, and expect only those nodes to return a CTS and, later after receiving the data frame and its RAK frame, return an ACK. Without an explicit ACK from each node in $S \setminus S'$, the sender of course has no way to know whether the nodes in $S \setminus S'$ have received the data frame. But is it possible that by receiving only the ACKs from those nodes in $S' \subseteq S$, the sender is able to conclude that all nodes in S have received the multicast data frame without collision, assuming the transmission error is caused primarily by the collision? In this section, we show that for some subsets S', this is indeed possible. We also establish a necessary and sufficient condition that characterizes all such subsets S'.

Let $A(s)$ denote the coverage area of s, and let $A(S) = \bigcup A(s_i)$ where $s_i \in S$.

Definition 1 *Let S be a set of nodes. A subset S' of S is said to be a* cover set *of S if $A(S') = A(S)$.*

The following theorem characterizes the above mentioned S'.

Theorem 1 *Let S be the set of all intended receivers of the multicast data frame. In the Batch_Mode_Procedure(), suppose that the sender receives an ACK from every node in a subset S' of S. A node p in $S \setminus S'$ is guaranteed to have received the data frame without collision if and only if S' is a cover set of S.*

Due to length limit, all the proof are omitted in this paper. According to Theorem 1, given a set of intended receivers S, if a minimum cover set of S can be found, then not only the size of the RTS frame but also the number of CTS and ACK frames can be greatly reduced. Computing the minimum cover set is itself an interesting and nontrivial computational geometric problem. In [18], we show that if the sender knows the locations of nodes in S, the minimum cover set of S can be computed in $O(n^{4/3})$ time, where n is the av-

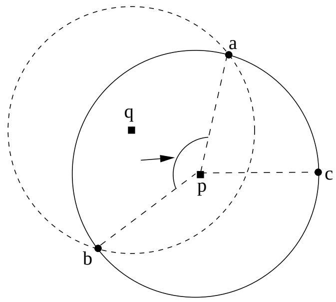

Figure 4. Cover Angle of node p for node q

erage number of intended receivers for a multicast request. For ease of reference, we state this result in the following as a theorem.

Theorem 2 ([18]) *The minimum cover set of a neighbor set S can be computed in $O(n^{4/3})$ time.*

Theorem 1 assumes that every node in S' returns an ACK and all the ACKs are received correctly. The next theorem indicates that even if not every nodes in S' returns an ACK or if not all ACKs are received correctly, we can still tell which nodes in $S \setminus S'$ must have received the data frame without collision.

Theorem 3 *Let S be the set of all intended receivers of the multicast data frame. In the Batch_Mode_Procedure(), suppose that the sender receives an ACK from every node in a subset $S_{ACK} \subseteq S'$, where S' is a cover set of S. Under the assumption that the primary transmission error is caused by collision, a node p in $S \setminus S'$ is guaranteed to have received the data frame without collision if and only if $A(p) \subseteq A(S_{ACK})$.*

To apply Theorem 3, we need an algorithm for easily checking if $A(p) \subseteq A(S_{ACK})$. That is, we want an efficient algorithm to identify if the transmission area of a node is completely covered by the transmission areas of a set of nodes. This is also a non-trivial computational geometry problem. In the following we describe an *angle-based scheme* for this purpose. Before stating the algorithm, we first define the cover angle for the neighboring nodes.

Definition 2 *Assume two nodes p and q are neighbors to each other. Let a and b denote the intersections of $A(p)$ and $A(q)$ boundaries, and c be the intersection of the straight horizontal line passing through p and $A(p)$ boundary to the east of p, the* cover angle *of p for q is defined to be $\angle apb$ and is denoted as $[\angle cpa, \angle cpb]$, where $\alpha = \angle cpa$ and $\beta = \angle cpb$ (both counter-clockwise). If two nodes are at the same location, their cover angle is defined to be $[0, 360]$. If two nodes are more than R away from each other, their cover angle is defined to be $\emptyset$.*

The arrow in Figure 4 indicates the cover angle of p for q to be $[\angle cpa, \angle cpb]$.

Assuming that the transmission radius is R, the cover angle $[\alpha, \beta]$ of two neighbors can be easily calculated.

Assume that a cover angle $\angle apb$ of p for q is non-zero, according to the definition of the cover angle, it is not difficult to see that the sector of $A(p)$ from $\overrightarrow{pa}$ to $\overrightarrow{pb}$ falls inside $A(p) \cap A(q)$, hence is covered by $A(q)$. The cover angle provides an efficient way to identify if the transmission area of a node p is completely covered by a set of nodes C.

Theorem 4 *Assume that all nodes have the same transmission radius R. Given a node p and a set of nodes C, if the union of the cover angles $\cup_i[\alpha_i, \beta_i]$ is $[0, 360]$, where $[\alpha_i, \beta_i]$ is p's cover angle for $s_i \in C$, the transmission area of p, $A(p)$, is completely covered by C.*

The *Location Aware Multicast MAC* (LAMM) protocol is a refinement of the Batch Mode Multicast MAC protocol. Assume the intended receiver set of the multicast data frame to be S and the ACK set to be S', we denote MCS(S) the minimum cover set computation procedure that takes S as input and returns the the minimum cover set of S using the algorithm in [18]. Also, we denote UPDATE(S, S') the angle-based procedure that takes both S and S' and returns the set of nodes in S that are not completely covered by S'. The Location Aware Multicast MAC protocol can be formally expressed as following using the Batch Mode Procedure in the BMMM protocol. (The receiver's Location Aware Multicast MAC protocol and the Batch Mode Procedure are exactly the same as in Figure 3.)

Sender's Location Aware Multicast MAC Protocol

if s has a multicast message to send to the nodes in S
  while $S \neq \emptyset$
    Batch Mode Procedure(MCS(S), S_{ACK})
    S = UPDATE(S, S_{ACK})

6 Analysis

BMMM, LAMM, BMW and BSMA all use the RTS-CTS approach. After sending an RTS frame, if the sender does not receive any CTS frame, it will enter another contention phase to retransmit the RTS. This process is repeated until the sender hears at least one CTS signal and thus sends the data frame. In this section, we analyze the expected number of contention phases needed before the sender sends data for each of the four protocols.

There are five reasons why the sender may not receive a CTS frame from a certain receiver:

- transmission errors in the RTS frame
- collision of the RTS frame
- the receiver is yielding to some other message transmission
- transmission errors in the CTS frame
- collision of the CTS frame

For BMMM, LAMM and BMW the sender receives one CTS frame a time so there is no collision of CTS frames. For BMW, CTS frames may collide.

Combine the first four factors and let q be the probability that the sender does not receive a CTS from a given node due to one or more of these four reasons. Let S be the set of intended receivers and $n = ||S||$ be its size. Let S' denote

Parameters	BMMM	LAMM	BMW	BSMA
$q = 0.05, n = 5, \|\|S'\|\| = 4$	1.00	1.00	1.05	3.27
$q = 0.05, n = 10, \|\|S'\|\| = 6$	1.00	1.00	1.05	4.08

Table 1. Expected number of contention phases before the sender sends data

the minimum cover set of S. The probability that at least one CTS frame is successful for BMMM, LAMM and BMW is $1-q^n$, $1-q^{\|S'\|}$ and $1-q$, respectively. If p is the probability for the sender to hear at least one CTS signal, the expected number of contention phases before the sender receives at least one CTS signal is $1+(1-p)+(1-p)^2+(1-p)^3+\cdots = 1/p$. Therefore, the expected numbers of contention phases for BMMM, LAMM and BMW are $\frac{1}{1-q^n}$, $\frac{1}{1-q^{\|S'\|}}$ and $\frac{1}{1-q}$, respectively.

But in BSMA protocol the CTS frames may collide with each other. [20] assumes the DS (direct sequence) capture ability of radio. Since the capture probability is low in general, the sender may still be unable to successfully detect the strongest CTS signal out of the colliding CTS signals. Therefore, to compute the probability of receiving one CTS frame, we need to take the capture probability into account. [23] gives a formula for computing the capture probability C_k for k concurrent signals. For the sender, the probability of receiving exactly k CTS signal is $C(n,k)(1-q)^k q^{n-k}$. Therefore, the probability of receiving one CTS frame is $\sum_{k=1}^{n} C(n,k)(1-q)^k q^{n-k} C_k$ and the expected number of contention phases is its reciprocal. We show two sets of probabilities in Table 1. The BMMM, LAMM and BMW protocols use much fewer contention phases before a data frame can be sent.

Next we analyze the overall numbers of contention phases needed by BMMM, LAMM and BMW. As we already know, the BMW protocol needs at least n contention phases. Considering our BMMM and LAMM protocol, suppose in a round of RTS/CTS/DATA/ACK transmission, the probability for an intended receiver to receive the data successfully is p. Let n denote the number of intended receivers and f_n be the expected number of contention phases for a multicast message that has n receivers. We can easily compute $f_1 = 1+(1-p)+(1-p)^2+(1-p)^3+\cdots = 1/p$. For $n = 2$ we have $f_2 = 1+C(2,1)p(1-p)f_1+C(2,2)(1-p)^2 f_2$ and the solution is $f_2 = \frac{3-2p}{p(2-p)}$. For $n = 3$ we have $f_3 = 1 + C(3,1)p^2(1-p)f_1+C(3,2)p(1-p)^2 f_2+C(3,3)(1-p)^3 f_3$ and so on. The expected numbers of contention phases are shown in Figure 5 for different n and $p = 0.9$. As can be seen, the expected number of contention phases is far less than n and it increases much slower than the linear function. Therefore, BMMM and LAMM use much less a number of contention phases than BMW. This conclusion will be testified by simulation in the next section. Furthermore, the lines of the expected number of contention phases in Figure 5 coincide with the lines of the average number of contention phases in Figure 9(a) very well.

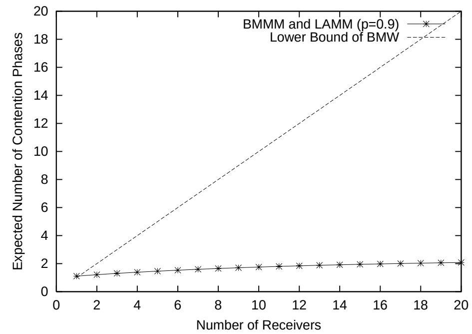

Figure 5. Expected number of Contention Phases in BMW, BMMM and LAMM

Parameter	Value
Signal Time	1 slot
Data Transmission Time	5 slots
Simulation Time	10000 slots
Time Out	100 slots
Radius	0.2
Unicast Message Ratio	0.2
Multicast Message Ratio	0.4
Broadcast Message Ratio	0.4
Message Generation Rate	0.0005/node/slot
Reliability Threshold	90%

Table 2. Parameters Used for Simulations

7 Simulation

To evaluate our multicast MAC protocols, we developed our own wireless LAN simulator. The protocols we simulated include our BMMM and LAMM, as well as BSMA [19] and BMW [20]. We randomly placed 100 nodes in a unit square. Each node is able to send unicast, multicast and broadcast data frames to its neighbors using the simulated protocol. We assume that the time is slotted so that the event (e.g., message sending and receiving) happens at the beginning of a slot. To ensure that BSMA in [20] works as designed, we adopted the direct sequence capture ability for the radio channel. The probability of capturing a collided CTS frame was set according to [23]. All the simulation results were the means of 100 runs of simulations with different random seeds. A multicast message transmission is considered *successful* if the message reaches a certain percentage of the intended receivers. We call such a percentage the *reliability threshold*. If a multicast message either reaches less than the reliability threshold of the intended receivers or times out before completion, the transmission is considered unsuccessful. The following table lists the parameters we used in our simulations, if not mention otherwise:

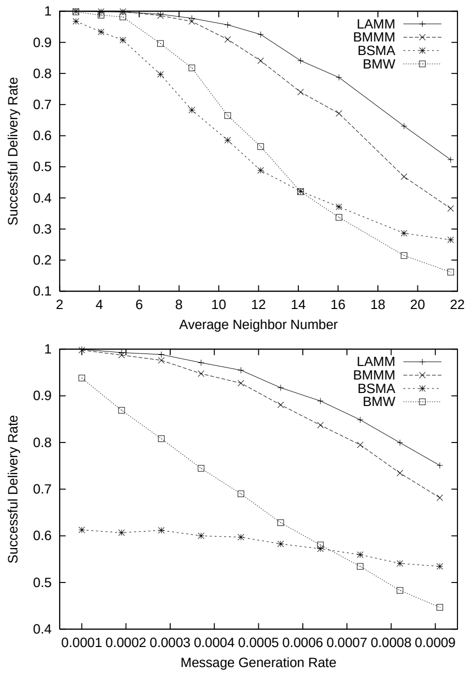

Figure 6. Successful Delivery Rate vs (a) Nodal Density (b) Message Generation Rate

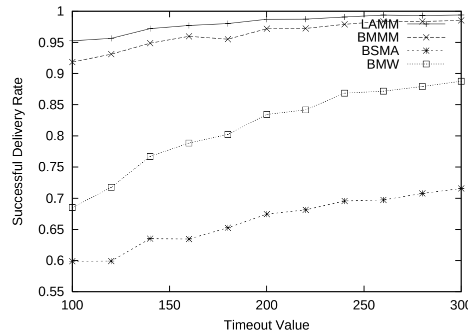

Figure 7. Successful Delivery Rate vs Timeout

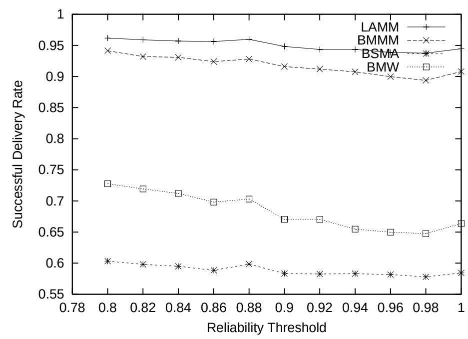

Figure 8. Successful Delivery Rate vs Reliability Threshold

7.1 Reliability

Reliability is the first feature we used to evaluate the protocols. To measure the reliability, we define the *successful delivery rate* to be the number of successful message transmissions divided by the total number of requests. A MAC protocol is reliable if it has a high successful delivery rate. Figures 6 (a) and (b) show the successful delivery rate under different nodal density and message generation rates. The x-axis is the average number of neighbors in Figure 6 (a) and the message generation rate in Figure 6 (b). The y-axis is the successful delivery rate for both Figures 6 (a) and (b).

As can be seen, the successful delivery rate of all protocols degrade when either the number of neighbors or the message generation rate increases. This is due to the fact that the more traffic in a transmission area, the more collisions may occur. Because of collisions, many messages time out before they can reach their destinations. In both Figures 6 (a) and (b), our LAMM and BMMM protocols enjoy the highest and the second highest successful delivery rate, respectively.

As we have pointed out, time-out is one of the major causes of the unsuccessful message transmissions. In Figure 7, we show how timeout value affects the successful delivery rate for different protocols. The x-axis is the timeout value ranging from 100 slots to 300 slots and the y-axis is the successful delivery rate in Figure 7. As expected, the larger the timeout value is, the higher successful delivery rate a protocol produces. No matter what timeout value is used, our BMMM and LAMM protocols constantly produce much higher successful delivery rate than BSMA and BMW protocols.

In the successful delivery rate definition, reliability threshold plays an important role. Next we illustrate how reliability threshold affects the successful delivery rate for different protocols in Figure 8, where the x-axis is the reliability threshold and the y-axis is the successful delivery rate. As can be seen, no matter what reliability threshold we use, our BMMM and LAMM protocols always produce much higher successful delivery rate than BMW and BSMA protocols.

7.2 Efficiency

The second feature we examine is the efficiency. In our simulations, We measure efficiency by the average number of contention phases and the average message completion time.

Figures 9 (a) and (b) show the average number of contention phases (i.e., CSMA/CA) for each protocol under different nodal density and message generation rates. The x-axis is the average number of neighbors in Figure 9 (a) and the message generation rate in Figure 9 (b). The y-axis is the average number of contention phases per message for both (a) and (b). In both Figures 9 (a) and (b), BMW requires the highest number of contention phases. Our BMMM and LAMM protocols are able to produce slightly lower aver-

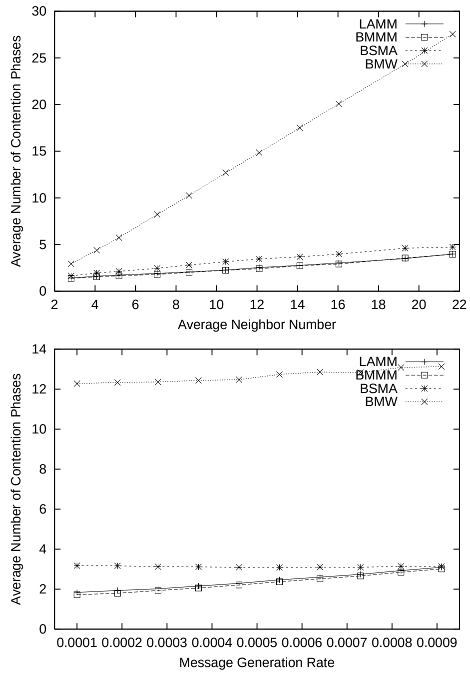

Figure 9. Average Number of Contention Phases vs (a) Nodal Density (b) Message Generation Rate

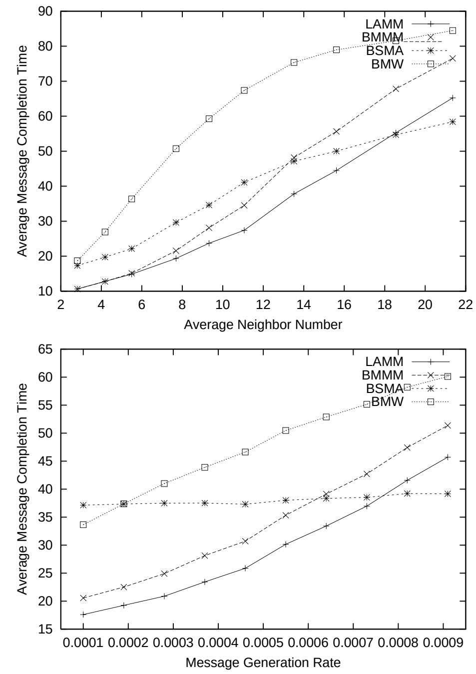

Figure 10. Average Message Completion Time vs (a) Nodal Density (b) Message Generation Rate

age number of contention phases than BSMA under different nodal density and message generation rates.

Figures 10 shows the average time required to complete a multicast for different protocol. The x-axis is the average number of neighbors in Figure 10 (a) and the message generation rate in Figure 10 (b). The y-axis is the average message completion time for both Figures 10 (a) and (b). As expected, Figures 10 (a) and (b) show that the LAMM protocol requires less time to complete a multicast than BMMM, which in turn requires less time than BMW. This is because of the excessive number of contention phases and control frames in BMW and the smaller number of CTS and ACK frames required by LAMM than by BMMM.

7.3 Comments on Simulation Results

The successful delivery rate is closely related to the average message completion time. The longer average message completion time, the more likely a multicast message may time out and thus lower the successful delivery rate. This explains why the rank of LAMM, BMMM and BMW protocols in Figures 10 (a) and (b) is the same as the rank in Figures 6 (a) and (b).

This simple relationship between the successful delivery rate and the average message completion time can not be applied to the BSMA protocol. In BMW, BMMM and LAMM, when a message is completely multicasted, all intended receivers are guaranteed to receive the message without collision. This is not true for BSMA because in BSMA a message can be considered completely transmitted even when the message has not yet reached its intended receivers. This property explains why BMMM and LAMM have a higher successful delivery rate than BSMA even in the case when BSMA's average message completion time is shorter than BMMM and LAMM as shown in Figures 6 and 10.

From the simulation results, we can draw the following conclusions:

- No matter what metric is used, the BMW protocol always incurs the highest overhead. Under most situations, it has the lowest successful delivery rate. Therefore the BMW protocol is not a good multicast MAC protocol.
- As we have said, BSMA itself is not logically reliable. Although BSMA uses just a slightly larger number of contention phases than BMMM and LAMM, it shows much lower successful delivery rate than the latter two even if we allow radio direct sequence capture ability.
- The proposed BMMM and LAMM protocols have the best performance in terms of successful delivery rate and number of contention phases.

8 Conclusion

In this paper, we investigated the existing wireless multicast MAC protocols and showed that they are either unreliable or inefficient. We proposed two reliable multicast

MAC protocols: The Batch Mode Multicast MAC protocol and the Location Aware Multicast MAC protocol that can co-exist with the current unreliable IEEE 802.11 multicast MAC protocol. Based on the IEEE 802.11 DCF unicast MAC protocol, BMMM coordinates the receiver's control frame transmissions by sender's RTS and RAK frames. It not only avoids the control frame collisions but also prevents any neighbor from passing its contention phase. This helps noticeably reduce the number of contention phases for a multicast request. As a result, it decreases the average total time required to complete a multicast request and reduce the chance of message timeout. LAMM uses two location-based procedures to further improve upon BMMM. Our analysis and simulation results showed that both BMMM and LAMM exhibit improved reliability and efficiency, with LAMM generally outperforming BMMM.

We conclude with a pointer to future work. Throughout this paper, our focus has been on resolving the hidden terminal problem for multicast. Another problem that is challenging in wireless medium access control is the exposed terminal problem. To the best of our knowledge, no multicast MAC protocol has addressed the exposed terminal problem. With the help of location information, we hope to find an efficient multicast MAC protocol that solves both the hidden and exposed terminal problems.

References

[1] http://www.fcc.gov/e911/

[2] Editors of IEEE 802.11, Wireless LAN Medium Access Control (MAC and Physical Layer (PHY) specification, Draft Standard IEEE 802.11, 1997.

[3] V. Bharghavan et al, “MACAW: A Media Access Protocol for Wireless LAN's,” Proc. ACM SIGCOMM 1994, pp. 212-225, Aug. 1994.

[4] P. Bose, et al, “Routing with guaranteed delivery in ad hoc wireless networks,” Proc. ACM DIAL M 1999, pp. 48-55, Aug. 1999.

[5] G. M. Djuknic and R. E. Richton, “Geolocation and Assisted GPS,” IEEE Computers, Feb, 2001.

[6] C. Fullmer and J. J. Garcia-Luna-Aceves, “Floor Acquisition Multiple Access (FAMA) for Packet Radio Networks,” Proc. ACM SIGCOMM 1995, pp. 262-273, Aug. 1995.

[7] Z. J. Haas and M. R. Pearlman, “The Performance of a New Routing Protocol for the Reconfigurable Wireless Networks,” Proc. IEEE ICC 1998, pp. 156-160, Jun. 1998.

[8] D. Johnson and D. Maltz, “Dynamic Source Routing in Ad Hoc Wireless Networks,” T. Imielinski and H. Korth, Eds. *Mobile Computing*, Ch. 5, Kluwer, 1996.

[9] P. Karn, “MACA - A New Channel Access Method for Packet Radio,” ARRL/CRRL Amateur radio 9th Computer Networking Conference, pp. 134-140, Apr. 1990.

[10] B. Karp and H. T. Kung, “GPSR: Greedy Perimeter Stateless Routing for Wireless Networks,” Proc. ACM MOBICOM 2000, pp. 243-254, Aug. 2000.

[11] L. Kleinrock and F. A. Tobagi, “Packet Switching in Radio Channels: Part I - carrier sense multiple access modes and their throughput-delay characteristics,” IEEE Trans. Commun., vol. COM-23, no. 12, pp. 1400-1416, 1975.

[12] Y. B. Ko and N. H. Vaidya, “Location-Aided Routing (LAR) in Mobile Ad Hoc Networks,” Proc. ACM MOBICOM 1998, pp. 66-75, Oct. 1998.

[13] J. Kuri and S. K. Kasera, “Reliable Multicast in Multi-Access Wireless LANs,” ACM/Kluwer Wireless Networks, vol 7, pp. 359-369, 2001.

[14] S. Lin and D. J. Costello Jr., “Error Control Coding: Fundamentals and Applications,” Prentice-Hall, Englewood Cliffs, NJ, 1983.

[15] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, “The Broadcast Storm Problem in a Mobile Ad Hoc Network,” Proc. ACM MOBICOM 1999, pp. 151-162, Aug. 1999.

[16] C. E. Perkins, E. M. Royer and S. R. Das, “Ad Hoc on Demand Distance Vector (AODV) Routing,” http://www.ietf.org/internet-drafts/draft-ietf-manet-aodv-06.tex, IETF Internet Draft, Jul. 2000.

[17] M. Sun, et al, “Location Aided Broadcast in Wireless Ad hoc Networks,” Proc. IEEE GLOBECOM 2001, pp. 2842-2846, Nov. 2001.

[18] M. Sun and Ten-Hwang Lai, “Location-aided Geometry-based Broadcast for Wireless Ad hoc Network Systems,” submitted for publication.

[19] K. Tang and M. Gerla, “MAC Layer Broadcast Support in 802.11 Wireless Networks,” Proc. IEEE MILCOM 2000, pp. 544-548, Oct. 2000.

[20] K. Tang and M. Gerla, “Random Access MAC for Efficient Broadcast Support in Ad Hoc Networks,” Proc. IEEE WCNC 2000, pp. 454-459, Sep. 2000.

[21] K. Tang and M. Gerla, “MAC Reliable Broadcast in Ad Hoc Networks,” Proc. IEEE MILCOM 2001, pp. 1008-1013, Oct. 2001.

[22] J. Wu and H. Li, “On Calculating Connected Dominating Set for Efficient Routing in Ad Hoc Wireless Networks,” Proc. ACM DIAL M, pp. 7-14, Aug. 1999.

[23] M. Zorzi and R. R. Rao, “Capture and Retransmission Control in Mobile Radio,” IEEE JSAC, vol. 12, No. 8, pp. 1289-1298, Oct. 1994.