Reliable Multicast in Multi-access Wireless LANs

JOY KURI[1], AND SNEHA KUMAR KASERA[2]

[1] *Center for Electronic Design and Technology,*
Indian Institute os Science,
Bangalore 560012, India
E-mail: kuri@cedt.iisc.ernt.in

[2] *Bell Labs Research,*
Lucent Technologies,
Holmdel, NJ 07733
E-mail: kasera@research.bell-labs.com

Multicast is an efficient paradigm for transmitting data from a sender to a group of receivers. In this paper, we focus on multicast in single channel multi–access wireless local area networks (LANs) comprising several small cells. In such a system, a receiver cannot correctly receive a packet if two or more packets are sent to it at the same time, because the packets "collide." Therefore, one has to ensure that only one node sends at a time. We look at two important issues. First, we consider the problem of the sender acquiring the multi–access channel for multicast transmission. Second, for reliable multicast in each cell of the wireless LAN, we examine ARQ–based approaches. The second issue is important because the wireless link error rates can be very high.

We present a new approach to overcome the problem of feedback collision in single channel multi–access wireless LANs, both for the purpose of acquiring the channel and for reliability. Our approach involves the election of one of the multicast group members (receivers) as a "leader" or representative for the purpose of sending feedback to the sender. For reliable multicast, on erroneous reception of a packet, the leader does not send an acknowledgment, prompting a retransmission. On erroneous reception of the packet at receivers other than the leader, our protocol allows negative acknowledgments from these receivers to collide with the acknowledgment from the leader, thus destroying the acknowledgment and prompting the sender to retransmit the packet.

Using analytical models, we demonstrate that the leader–based protocol exhibits higher throughput in comparison to two other protocols which use traditional delayed feedback–based probabilistic methods. Last, we present a simple scheme for leader election.

Keywords: Wireless Local Area Networks, Reliable Multicast, Multi-access Channel, Feedback Collision

1 Introduction

Multicast is an efficient paradigm for transmitting data from a sender to a group of receivers, also called “group members.” Multicast incurs lower network and end-system costs than broadcast to all nodes in the network or unicast to individual group members. Several applications including information dissemination, multimedia conferencing, shared whiteboards, distance learning, multi–party games and distributed computing use (or will use) multicast communication.

Future networks will include large numbers of portable devices moving among wireless cells. Several of these devices (or receivers) in a cell might be interested in receiving multicast data sent from a local or a remote sender. For efficient utilization of the wireless bandwidth and for better performance, it will be important to have multicast communication support in these cells. To distinguish between the original sender of the multicast data and the node that multicasts data on the wireless link in a cell, we will refer to the first one as the “original sender” (e.g., a remote node multicasting stock quotes) and the latter as the “sender” (e.g., base-station of the cell).

In this paper we focus on multicast in single channel multi–access wireless local area networks (LANs) comprising several small cells. In such a system, a receiver cannot correctly receive a packet if two or more packets are sent to it at the same time, because the packets “collide.” Therefore, one has to ensure that only one node sends at a time. We look at two important issues. First, we consider the problem of the sender acquiring the multi–access channel for multicast transmission. Second, for *reliable* multicast in each cell of the wireless LAN, we examine ARQ–based approaches. The second issue is important because the wireless link error rates can be very high. When the original sender multicasts data to a large number of wireless receivers which might be far away from it, recovery from wireless link errors exclusively from the original sender will be highly inefficient. Instead, *local* error recovery from the sender, the base–station, on the wireless link, will help in increasing throughput, reducing delay and bandwidth consumption.

Acquiring the shared channel for transmission in a cell involves sending a request and getting a positive feedback from the recipient. This works well for unicast but cannot be simply extended to multicast. This is because an uncontrolled feedback from several group members will result in a feedback collision at the sender. The same problem also arises when a sender expects feedback from the receivers for ensuring reliable multicast communication. Again, uncontrolled acknowledgments (ACKs) or negative acknowledgments (NAKs) from several group members will result in a collision at the sender, delaying any error recovery and wasting bandwidth. Traditional delayed feedback–based probabilistic methods could be used for reducing the feedback collision to some extent but they are not very efficient either.

We present a new approach to overcome the problem of feedback collision in single channel multi-access wireless LANs, both for the purpose of acquiring the channel and for reliability. Our approach involves the election of one of the multicast group members (receivers) as a “leader” or representative for the purpose of sending feedback to the sender. To illustrate our approach, we consider the reliable transmission of a packet. On erroneous reception of the packet, the leader does not send an acknowledgment, prompting a retransmission. On erroneous reception of the packet at receivers *other* than the leader, our approach allows negative acknowledgments from these receivers to collide with the acknowledgment from the leader, thus destroying the acknowledgment and prompting the sender to retransmit the packet. The ACKs and/or NAKs are sent immediately after packet transmission is over; so there is no waiting involved as in delayed feedback–based methods, thereby avoiding wasted channel bandwidth and improving performance. This approach can be potentially integrated with the current wireless LAN standard (IEEE 802.11).

Using analytical models, we analyze the throughput behavior of the leader–based protocol and two other protocols which use traditional delayed feedback–based probabilistic methods. We demonstrate that the leader–based protocol exhibits higher throughput. Last, we present a simple scheme for leader election.

The remainder of this paper is structured as follows. In the next section we examine related work. In Section 3, we examine why it is necessary to make the wireless link reliable. In Section 4, we describe the problem setting. We propose the leader–based protocol for channel access and error recovery, as well as two other protocols based on traditional probabilistic approaches in Section 5. Section 6 contains our performance study. In Section 7 we discuss leader election. Conclusions and directions for future work are contained in Section 8.

2 Related Work

Multicast is being recognized as an efficient communication paradigm and is getting increasing attention from the mobile and wireless network community. In [16], designs for efficiently supporting multicast for mobile hosts on the Internet have been presented. In [12], an approach for supporting host mobility using IP multicasting as the sole mechanism for addressing and routing packets to mobile host has been considered. Both these proposals focus on mobility aspects and are concerned with network layer and routing issues. They do not deal with error recovery or with multi-access channels.

As far as multi-access wireless LANs are concerned, most of the existing work [2], [4], [5], [8] has focussed upon point-to-point unicast communication. The problem of acquiring the shared channel for multicast has been mentioned in [2] but no solution has been proposed. Recently, Bharghavan [3] has proposed a token–based solution for multicast in multi-access wireless LANs. Here, the base station of a cell in the wireless LAN distributes tokens to potential senders in the cell. When the base-station wishes to multicast, it does not give any token to other members of the cell for the purpose of acquiring the channel. Our work differs from Bharghavan's work in the following significant ways. First, we do not give control to any particular node for co–ordinating transmissions; rather all nodes including the base-station contend for the channel. Second, in addition to the problem of acquiring the channel we also provide solutions for reliable multicast.

In [14], it has been noted that the transmission of multicast, as proposed in the current IEEE 802.11 standard, is less robust due to absence of positive acknowledgment for multicast. Our leader–based protocol addresses this concern.

3 Importance of Reliable Wireless Links

In this section we discuss the importance of providing reliable wireless links for multicast communication. For the class of multicast applications having strict end–to–end delay requirements (for example, multimedia conferencing), error recovery on an end–to–end basis is usually not an option because it takes too long. However, link–level error recovery operates on a considerably smaller time scale (assuming that the quality of the links is not too bad), and is therefore a viable approach. Investing in link–level error recovery is worthwhile because it improves the quality of the links as seen by the applications and consequently improves the quality of multimedia applications as seen by the end user. For end–to–end *reliable* multicast communication applications such as multicast file transfer, dissemination of stock quotes and shared whiteboards, wireless link–level reliability saves time as well as both network and end-system resources.

To focus on the wireless links, we consider a loss–free wired network (see Figure 1) and assume that losses take place only on the wireless links. Figure 2 shows the average number of retransmissions required for correct reception of a packet from an original sender as a function of the number of receivers. The loss probabilities shown in Figure 2 are for individual wireless links. The number of retransmissions plotted on the y–axis is obtained by using the expressions derived in [13]. We see that as the number of wireless receivers that use a reliable multicast application grows, the number of retransmissions also increases. The increase is more for higher loss probabilities.

The need for additional transmissions due to errors in the wireless links puts unnecessary processing burden on the original sender. These additional transmissions go over the entire wired multicast tree and also the wireless links, wasting bandwidth and also leading to processing of unwanted redundant retransmissions at those receivers ([10]) which might have already received the packet. If the base-stations were to take the responsibility of supplying retransmissions rather than the original sender, then the load of supplying retransmission gets distributed across base-stations. Each base–station needs to supply only a few retransmissions (this is the case when there are only a small number of wireless receivers in Figure 2) which are restricted only within the area controlled by the base–station.

In summary, the impact of recovery from wireless link errors only from the original sender is much more severe for multicast applications. *Local* error recovery done from base–stations, which are upstream and closest to the point of wireless–link losses, is much more efficient.

In some loss scenarios it might not be possible to ensure full wireless link reliability at the link–layer (as discussed in Section 5.4). In these scenarios, the lost packets have to be recovered at the transport layer. For end–to–end reliable multicast applications, recovery at the transport layer will also be needed to ensure end–to–end reliability. The recovery processes at the link and transport layers might interfere with each other [6]. This interference could be reduced if the delay in link layer recovery is small. Hence one of the goals of any link layer error recovery scheme should be to keep the delay minimal.

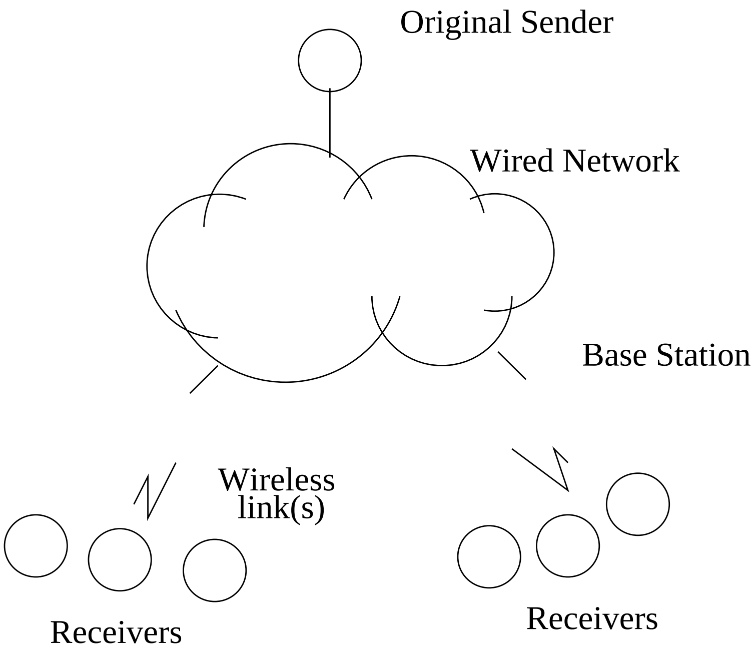

Figure 1: Multicast Network.

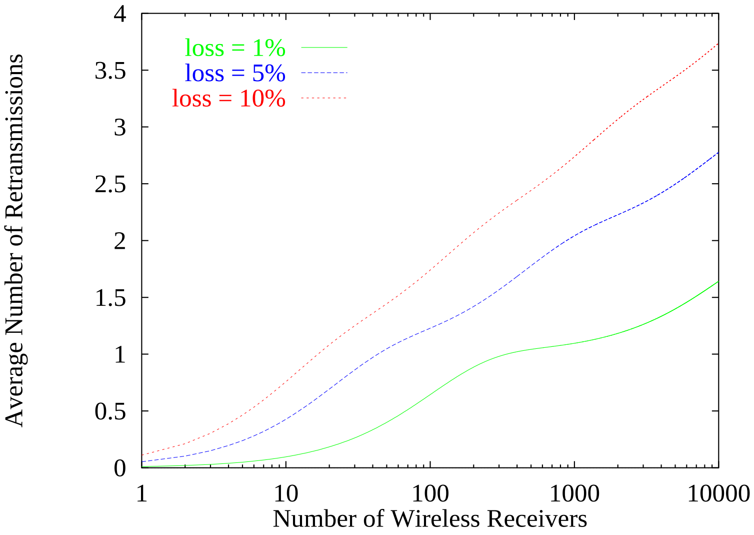

Figure 2: Average number of retransmissions.

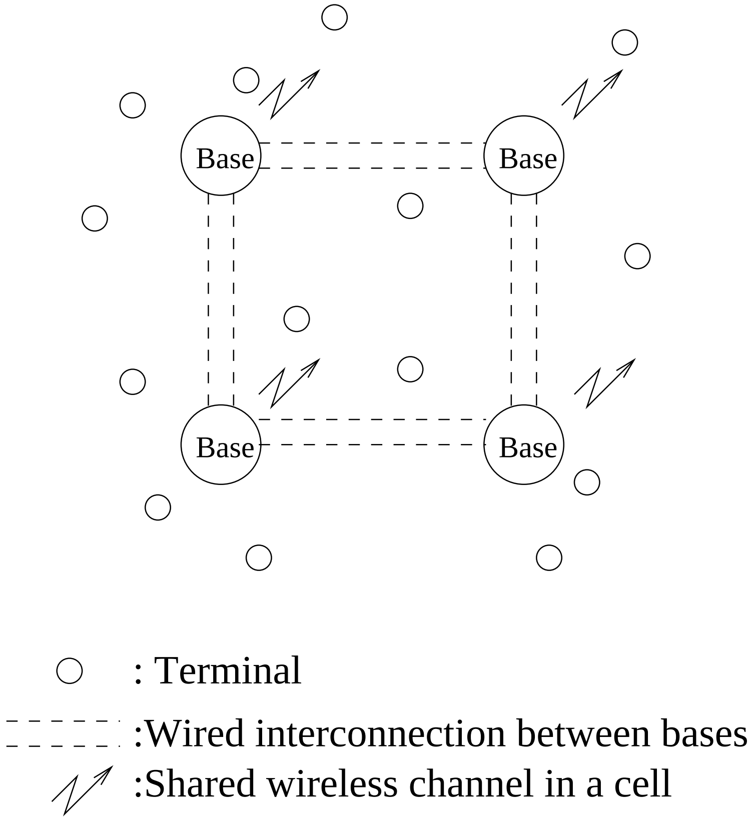
Base
Base
Base
Base
○ : Terminal
:Wired interconnection between bases
:Shared wireless channel in a cell

Figure 3: Top view of the system, showing bases, terminals,the shared wireless channel in each cell and the wired interconnection between bases.

4 Problem Setting

We consider multicast communication in a microcell–based wireless network supporting mobile terminals (Figure 3). Each microcell (henceforth called "cell") is administered by a base station located at the center of the cell. The mobile terminals in a cell communicate with each other and the base station. All multicast communication is directed from base station to the terminals. The group of mobile terminals receiving multicast from the base station are also called receivers.

Time is measured in terms of a basic unit called the "slot." Thus, time evolves in discrete steps: 1st slot, 2nd slot, $\ldots$, n^{th} slot, and so on. System events, like transmission/reception of a packet, occur at integer–valued slot times. It is important to ensure that all the entities in a cell — the base and the terminals — identify the beginnings and ends of slots unambiguously and simultaneously. This is the problem of synchronization and, for the purposes of this paper, we assume that perfect synchronization is achieved.

There are significant differences between the wired and wireless LAN transmission media, which make it impossible to port traditional wired–LAN MAC strategies like

CSMA/CD to wireless LANs. In a multi–access wireless LAN, collision detection is not practical. This is because the dynamic range of the signals on the medium is very large, so that a transmitting station cannot effectively distinguish incoming weak signals from noise and the effects of its own transmission [15]. In order to prevent loss of bandwidth due to finding out about a collision (possibly due to an ACK/NAK) after the entire packet has been transmitted, a transmitter needs unambiguous and conclusive evidence that it has acquired the channel before starting transmission. In the wireless context, this evidence can be provided by means of a handshaking mechanism implemented using short fixed–size signaling packets: Request–to–Send (RTS) and Clear–to–Send (CTS) [8], [9].

We now briefly describe the RTS–CTS mechanism for unicast transmission. When a base or a terminal wishes to transmit, it sends an RTS packet to the intended recipient; this RTS packet contains the length of the proposed transmission. If the recipient hears the RTS, it replies immediately with a CTS; the CTS also contains the length of the imminent data transmission. Upon hearing the CTS, the initiator goes ahead with the transmission. Any terminal overhearing an RTS defers all transmission for an interval sufficient for the associated CTS to be sent and heard. Any terminal overhearing the CTS defers for the length of the oncoming data transmission. After a data packet is received, the recipient provides link–level ARQ feedback, by means of an ACK.

The RTS–CTS mechanism also helps in combating the hidden terminal problem [8]. When a transmitter about to transmit senses no carrier in its vicinity, it cannot conclude that the shared channel is unused, because another transmitter *hidden* from it may be transmitting at that instant. With the RTS–CTS mechanism, the hidden terminals can hear the CTS and defer using the channel. In this paper we consider that all terminals in a cell are within the range of one another and the base station. All terminals have a consistent view of what is going on in the cell and that there are no hidden terminals. A discussion on the impact of hidden terminals on our work is presented in Section 5.4.

The IEEE 802.11 Media Access Control standard uses RTS–CTS exchange. It is important that the RTS–CTS control structure be retained when multicast functionality is overlaid. Consequently, when adding multicast functionality, we devise ways of extending the access control mechanism rather than modifying its basic structure.

While the RTS–CTS mechanism, described above, for co–ordinating access to the channel and supplying link–level ARQ feedback works well enough for unicast transmissions, it runs into problems straight away in the context of multicasting. With the above protocol, each of the members in a multicast group would respond with a CTS to a multicast–RTS from the base, leading to a CTS collision at the base. A similar collision problem can also be expected with respect to the feedback (ACK or NAK) provided by the link–level ARQ mechanism.

Standard probabilistic approaches can be used to tackle the CTS collision problem. In the "delayed feedback" scheme, terminals hearing a multicast–RTS send a CTS with a random delay, hoping to avoid a CTS collision. Another possibility is the "probabilistic feedback" scheme, where each receiver sends a CTS immediately, but only with a certain probability. We will also consider protocols based on these ideas. To tackle the ACK/NAK collision problem, a contention–based approach is possible, where receivers contend for the channel to send feedback. However, the probabilistic and contention–based approaches suffer from problems of their own, as will be seen in subsequent sections. This motivates us to develop a new protocol, that is leader–based, that addresses these specific problems satisfactorily.

5 Protocols

We now propose three generic protocols, one leader–based and two that are based on random timers and probabilistic measures, for reliable multicast over a multi–access wireless LAN. All these protocols are for a single sender, the base–station, sending reliably to a group of receivers within a cell. We assume that the basic support for link level multicast, such as link level multicast address, is available at both the base-station and the receivers. The receivers which subscribe to the multicast address are said to belong to the multicast group corresponding to the multicast address.

5.1 Leader–Based Protocol

We now present our leader–based protocol for reliable multicast over a multi-access wireless LAN. This protocol assumes that one of the receivers of the multicast has been chosen to be a leader for the purpose of supplying CTS and ACK in response to RTS and data packets (of length l, say), respectively. We will discuss the leader election process separately in Section 7. The leader–based error recovery protocol, termed LBP, is specified as follows:

[**A**] Base ⟶ Receivers (Slot 1)
Send multicast–RTS.

[**B**] Receivers ⟶ Base (Slot 2)
Leader: if ready to receive data, send CTS.
if not ready to receive data (e.g., due to insufficient buffers), do nothing.
Others: if ready to receive data, do nothing.
if not ready to receive data, send NCTS (*Not* Clear to Send)[1].

[**C**] Base ⟶ Receivers (Slot 3)
If a CTS was heard in slot 2, start multicast transmission.
If no CTS was heard in slot 2, back off and go to Step A.

The next step is executed only when multicast transmission occurs in Step C.

[**D**] Receivers ⟶ Base (Slot $(l+3)$)
Leader: if packet received without error, send ACK.
if in error, send NAK.
Others: if packet received without error, do nothing.
if in error, send NAK.

[1] Note that a version of LBP without NCTS is perfectly possible; however, incorporating NCTS provides richer semantics.

LBP uses both ACKs and NAKs from receivers as feedback to the sender. It makes an interesting use of collisions associated with the multi–access channel. It allows collision of an ACK with one or more NAKs to ensure that the sender does not get a positive feedback if one or more group members receive erroneous transmission.

The next two subsections describe the other two protocols that do not assume the presence of any leader. We propose these protocols mainly for comparison purposes. Both these protocols incorporate only negative acknowledgment based error recovery and are similar in principle to the error recovery protocols proposed for wired networks.

5.2 Delayed Feedback–Based Protocol

In the delayed feedback–based protocol, the CTS collisions are sought to be avoided using a random timer. This protocol, termed DBP, is specified as follows:

[**A**] Base $\longrightarrow$ Receivers

1. Send multicast–RTS.
2. Start a timer (timeout period T), expecting to hear a CTS before the timer expires.

[**B**] Receivers $\longrightarrow$ Base

1. On hearing RTS, start timer with an initial value chosen randomly from $\{1,2,\ldots,L\}$.
2. Decrement timer by 1 in each slot.
3. If a CTS is heard before timer expires, freeze timer (CTS suppression).
If no CTS is heard before timer expires, send CTS.

[**C**] Base $\longrightarrow$ Receivers

If no CTS is heard within T, back off and go to Step A.
If a CTS is heard within T (at a random time), start data transmission.
After finishing transmission, prepare to transmit next packet and go to Step A (no waiting for feedback).

The next step is executed only when multicast transmission occurs in Step C.

[**D**] Receivers $\longrightarrow$ Base

If packet received without error, do nothing.
If in error, contend for the channel to send NAK.

5.3 Probabilistic Feedback–Based Protocol

The probabilistic feedback–based protocol, termed PBP, is similar to DBP with one important difference. In PBP, instead of waiting for a random number of time slots to send a CTS, the group members send out a CTS in the slot following the RTS ($T = L = 1$), with a certain probability. This probability is chosen based on the number of group members. As in the case of LBP, the receivers in PBP could send NCTS with probability 1 if they are not ready.

5.4 Discussion

We now present a qualitative discussions of the three protocols described above. In comparison to LBP, a successful RTS–CTS exchange would take longer in both DBP and PBP. This is because DBP and PBP have to deal with the possibility of CTS collisions. DBP delays feedback to reduce the possibility of collision. PBP does not delay feedback but might have to go through several rounds of RTS–CTS exchange due to CTS collision or due to receivers not sending any CTS at all. This additional delay and failed exchanges reduce channel utilization.

As DBP and PBP are NAK–based, the link level buffer requirements in DBP at the base-station as well as the receivers are higher. At the base-station, a packet has to be kept for longer to ensure that most of the retransmission requests can be serviced. At a receiver, more buffer will be required to buffer out–of–order packets so that upper layers get ordered delivery. Another problem with DBP and PBP is the choice of right parameters for waiting times and probability of sending feedback. This choice is dependent upon the number of group members. The group members are not likely to have an estimate of the group size. It is possible for the sender to do this estimation and send out the right parameters with the RTS to save them from implementing complex estimation mechanisms.

In all the three protocols, we have not considered the case where the RTS is received only by some but not all group members. In this case it is possible that the RTS–CTS exchange will go through and the sender will successfully transmit the packet which might not be received by the receivers that did not receive the RTS. These packets could be recovered at the upper layers if required by the applications (for e.g., reliable multicast applications). If control packets are not lost, LBP guarantees in–sequence delivery, which DBP and PBP cannot.

A flexible flow control feature is built into LBP and PBP by means of NCTS. This is flexible because it allows prevention of data transmission even if one receiver is not ready. DBP can try to do this (by refusing to send CTS or by sending NCTS) but have no guarantee of success, because somebody else's CTS may initiate transmission.

Finally, we consider the impact of hidden terminals on the operation of LBP. In LBP, the leader acts as a representative of the receivers in the multicast group. As long as there are no hidden terminals in the cell, the leader behaves as a true representative. But in the presence of hidden terminals, the situation is complicated because the leader might see a multi–access wireless channel *different* from that seen by other terminals.

In the following, we use the term "hidden terminal" to mean a terminal hidden from the leader. Let the leader be denoted by L, the hidden terminal by H and let M denote a member of the multicast group other than the leader.

The impact of the hidden terminal H is felt when a collision occurs between the multicast–RTS (m–RTS) transmitted by the base and a unicast–RTS (u–RTS) transmitted by H; note that if the m–RTS or the u–RTS is transmitted *alone* in a slot, viz., there is no RTS–collision, then no problem arises. Suppose, first, that H does not belong to the multicast group. We consider the sequence of events following a collision in slot n, say. There are two scenarios to be looked at:

(a) H is within range of at least one member, say M, of the multicast group.

1. M is unable to hear the m–RTS from the base because of the interference caused by H.

2. Consequently, in slot $(n+1)$, M does nothing; however, this is exactly what M's response would be if it had heard the m–RTS and was ready to receive multicast data.
3. M may hear the m–CTS from the leader in slot $(n+1)$, depending on whether M and the leader are within range of each other.
4. From slot $(n+2)$, M hears the multicast transmission from the base.

Note that the u–RTS transmitted by H is not replied to by the base (because the base itself transmitted a packet in the same slot), and so unicast transmission by H cannot begin. Further, H will sense carrier from slot $(n+2)$ onwards and refrain from sending another u–RTS till the multicast transmission ends. Thus, the hidden terminal H is unable to disrupt multicast transmission. However, the downside is that in step (2.) above, M does not have the possibility of sending an $m--NCTS$ to indicate that it is not ready to receive multicast transmission.

(b) H is not within range of any member of the multicast group.

In this case, H cannot affect any member of the multicast group, and multicast transmission proceeds as usual. Also, owing to the reasons given in (a) above, the unicast transmission attempted by H is "killed" and not re–attempted before the multicast transmission is over.

Suppose next that H itself belongs to the multicast group. Then, both the scenarios discussed above are again applicable, with the addition of a new feature, viz., that H will hear multicast data from slot $(n+2)$ onwards, without having heard the initial m–RTS from the base. Of course, H had no chance to expect a multicast transmission because of the collision it caused in slot n, but the multicast data is still available for it.

There is another way in which hidden terminals can affect the operation of LBP. Suppose that the m–RTS and m–CTS exchange has been completed and the base is about to begin multicast transmission. Consider a hidden terminal H that did not hear the m–CTS. It is possible that H sends a u–RTS packet immediately after m–CTS; that is, a collision occurs between the u–RTS from H and the multicast packet from the base. All group members within the range of H will fail to hear the multicast packet. Note, however, that even in the case of DBP and PBP, multicast transmission can be disrupted in this way. As usual, unicast transmission from H cannot begin because the u–RTS is not replied to.

In summary, there are two ways in which H can affect the operation of LBP. The first occurs when there is a collision during the control packet exchange (m–RTS and u–RTS collide). This does not prevent start of multicast transmission, but a member within range of H is deprived of the possibility of aborting the multicast transmission by sending NCTS. The second way in which multicast transmission can be affected occurs when there is a collision affecting the multicast data. All group members within range of H fail to hear the transmitted data; however, the same holds true for the other protocols DBP and PBP.

We end this section by noting that we have considered the case where the base station multicasts data to the terminals in a cell. Mobile terminals in a cell may want to multicast data to other terminals within the cell. In our framework, a mobile terminal can do this by sending unicast data to the base station which in turn multicasts the data to the multicast group members in the cell using the above protocols. The problem with this approach is that if all terminals and the base station contend for the multi–access channel with equal priority, a backlog could build up at base station. Development of mechanisms for restricting multicast from terminals or for providing higher priority to base station multicast in comparison to terminal multicast will be an important future work.

One possibility is to allocate equal priority to each *session* with data to send, rather than allocating equal priority to the base station and terminals. This would translate to a higher priority for the base station, as it is likely to have several sessions with data to send. We would also like to investigate the issue of fairness between multicast and unicast transmissions. We note that multicast can require more retransmissions than unicast *even in the wired environment*, because the probability of a "successful" transmission reduces as the size of the multicast group increases. Thus, the issue of fairness is applicable not only in the scenario considered in this paper, but also in general as well.

6 Performance Study

In this section, we compare the performances of LBP, DBP and PBP. We consider a scenario where multicast traffic is the *only* traffic present in the cell. We also assume that control packets (e.g., RTS, CTS, ACK, NAK) are never lost. Time is measured throughout in terms of a basic unit called the "slot."

The basic criterion used for studying the performances of LBP, DBP and PBP is the mean "channel holding time" associated with a tagged data packet (also referred to as the "cost" corresponding to that packet). This is a natural criterion to use because the reciprocal of the mean channel holding time provides a measure of throughput. The channel holding time is obtained by summing up the time, to access the channel and to actually transmit data or feedback, associated with successful transmission of the tagged data packet to all group members.

We consider the idealized case of the error–free channel first. This is evidently in favor of DBP and PBP, since no retransmissions are necessary. We derive analytical expressions for the mean access periods under DBP and PBP. In the subsequent section, that considers a lossy channel, we derive a *lower bound* to the mean channel holding time under DBP. This lower bound is valid for a completely general loss model.

6.1 Error–free channel

Performance of DBP

In DBP, a receiver hearing a multicast–RTS from the base starts a timer with a value chosen at random (uniformly) from the set $\{1,2, \ldots, L\}$. We assume that the value L is made available to the receivers by the base; for example, it may be carried in a field in the RTS packet. The receiver whose timer expires sends a CTS. Upon hearing the CTS, other receivers whose timers have not yet expired suppress their own CTSs. A CTS collision occurs if two or more receivers happen to choose the same initial value for their timers.

Since the receivers send the CTS after a delay, the base must wait for some time to hear the CTS. This is the base's timeout period of T slots. If a base does not hear a CTS

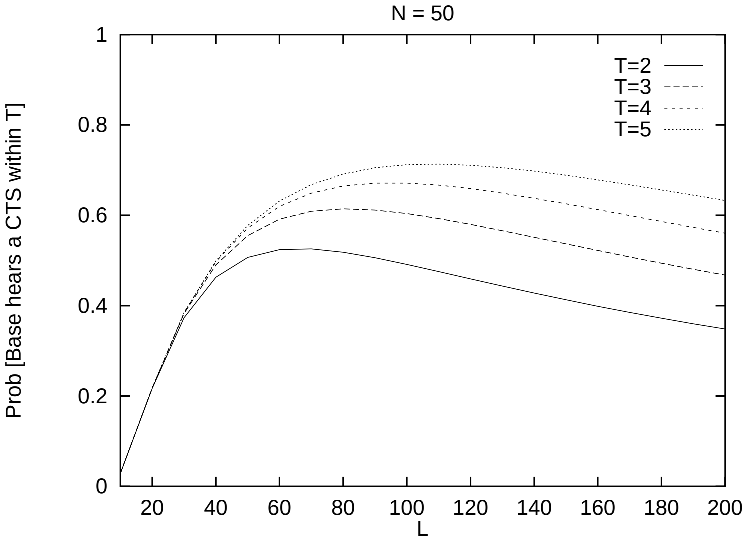

Figure 4: Variation of p_h with L, keeping N and T fixed.

within time T, it assumes there was a collision, and tries again. We choose $T < L$. This is because if T is large, then a lot of time is wasted before the base times out. On the other hand, choosing a moderately large L helps in avoiding a CTS collision within T.

Probability of receiving a CTS

The first question that arises in this scenario is: given the number of receivers N, L and T, what is the probability that the base hears a CTS within time T? Let this probability be denoted by p_h. p_h can be expressed as follows (see [11] for the derivation):

$$p_h = \frac{N}{L} \sum_{i=1}^{T} \left(\frac{L-i}{L} \right)^{N-1} \quad (1)$$

In Figure 4, we show how p_h varies with L, when N and T are held fixed. In all cases we find that p_h first increases, hits a peak and then decreases as L is increased. When L is small, the chances of CTS collision increase. When L is large, the chances of *no* receiver sending a CTS within the timeout period T go up. The best values of p_h are therefore found in the middle.

Average length of the access period

In DBP, after sending out a multicast–RTS, the base waits for the timeout period T to hear a CTS. If no CTS is heard within T, the base backs off and restarts the whole process by sending out a multicast–RTS again. The back–off feature is intended to resolve contention for the channel; i.e., it is intended to come into play when the RTSs sent out by the contenders collide. The failure to hear a CTS is interpreted by the base as contention for the channel among senders.

However, in DBP, the lack of a CTS can be caused simply by colliding CTSs, even when there is absolutely *no contention* for the channel. So, a CTS collision causes the base to unnecessarily back off. This will clearly increase the average length of the access period.

In order to create a situation favorable to DBP, we make the following assumption:

> *Assumption S: If no CTS is heard within the timeout period T, the base does not back off.*

Under this condition, we ask the question: on the average, how long does the base spend in the access period?

Let T_a^{DBP} be the random variable representing the total time spent by the base in the access period, measured from the instant when it is ready to send the first RTS. We assume that it takes 1 slot to transmit the RTS or any other control packet. Let $\mathcal{A}$ denote the event that the base hears a CTS within T slots of sending the first RTS, and $\overline{\mathcal{A}}$ denote the complementary event. Then we have

$$T_a^{DBP} = \begin{cases} 1+\tau & \text{if } \mathcal{A} \text{ occurs} \\ (1+T)+W_a & \text{if } \mathcal{A} \text{ does not occur,} \end{cases}$$

where $\tau \leq T$ is the (random) time at which the CTS is heard if $\mathcal{A}$ occurs, and, W_a is the time spent in the access period after the first timeout.

Now the distribution of W_a is the same as the distribution of T_a^{DBP}, i.e.,

$$W_a \overset{d}{=} T_a^{DBP},$$

where $X \overset{d}{=} Y$ denotes that random variables X and Y are equal in distribution. Noting that $\text{Prob}(\mathcal{A}) = p_h$, we obtain

$$E(T_a^{DBP}) = E(\tau/\mathcal{A}) + \frac{(1-p_h)}{p_h}T + \frac{1}{p_h} \tag{2}$$

In Figure 5, we present some examples of how $E(T_a^{DBP})$ varies with the parameters L and T. The number of receivers N is chosen to be 30. For a fixed T, $E(T_a^{DBP})$ first decreases, reaches a minimum and then increases again as L is increased. This is because $E(T_a^{DBP})$ is high when p_h is low and vice versa (Equation 2), and Figure 4 shows that p_h is low at the extremes of L and high in between.

Performance of PBP

Next we consider protocol PBP. In this case, after hearing the multicast–RTS, a receiver sends a CTS in the next slot with probability p. The base waits for 1 slot after sending the RTS. If exactly 1 member happened to reply then the access period is complete. If the base does not hear a CTS then it has to restart the process by sending the multicast–RTS again.

So, the minimum time spent in the access period is 2 slots, 1 to send the RTS and 1 to hear the CTS. Let p_o be the probability that the access period lasts 2 slots. Clearly,

$$p_o = Np(1-p)^{N-1}$$

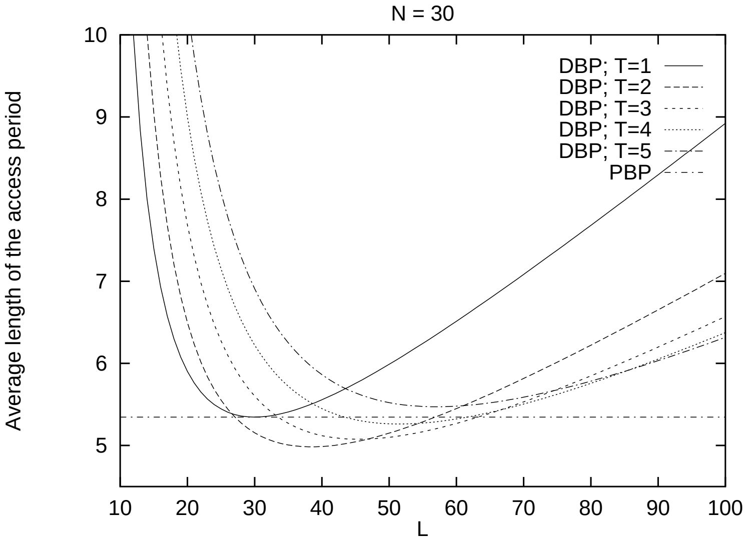

Figure 5: Comparison of the expected time spent in the access period under DBP and PBP.

Under Assumption S, the number of attempts necessary for the access period to be complete is geometrically distributed with parameter p_o. Hence the mean time spent in the access period, $E(T_a^{PBP})$, is given by $2/p_o$. To minimize this time, we choose p so that p_o is maximized; this is achieved for $p = 1/N$, giving the following expression for the mean time:

$$E(T_a^{PBP}) = \frac{2}{\left(1 - \frac{1}{N}\right)^{N-1}} \tag{3}$$

The value N can be transmitted to the receivers from the base in a field of the RTS packet, for example.

Comparison of DBP and PBP

Fig 5 also shows the mean time spent in the access phase under DBP and PBP (Equations 2 and 3). Under PBP, the value of N determines the mean time, while under DBP, we have two additional parameters, L and T, that must be assigned values. We see that, with appropriate values for L and T, DBP can have a shorter mean access period than PBP. Therefore, in the rest of the paper, we abandon PBP and consider DBP only.

Cost under DBP versus Cost under LBP

Consider DBP. When the channel is error–free, no NAKs are necessary because no packet is received in error. Then, a sample path of events on the channel may look like Fig 6: The base transmits a multicast–RTS (time taken: 1 slot) and then waits for the timeout period T to hear a CTS. After possibly several attempts, the base hears the CTS and transmits the packet.

We focus on a tagged packet and consider the mean time required to transmit the

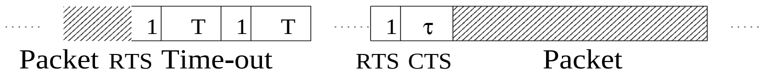

Figure 6: Events on the error–free channel; DBP.

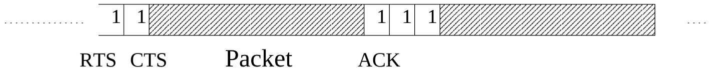

Figure 7: Events on the error–free channel; protocol LBP.

packet, including the time spent in the access period. We consider this time to be the "cost" associated with the tagged packet. The cost to transmit a packet gives a measure of the efficiency of the protocol. Let the data packet transmission time be C slots. Then, from Figure 6, we find that the cost of a packet under DBP is: $E(T_a^{DBP}) + C$, where $E(T_a^{DBP})$ is obtained from Equation 2.

On the other hand, consider the events on the channel under LBP, shown in Figure 7. Here, a packet transmission is preceded by 2 slots: 1 for the multicast–RTS, immediately followed by the CTS. In addition, a packet transmission is followed by an ACK packet which also occupies 1 slot. Thus, the cost of a packet transmission under LBP is: $(C+3)$. So, a comparison of packet transmission costs between LBP and DBP reduces to finding the best values of $E(T_a^{DBP})$. Assuming that it takes 20 slots to transmit a data packet ($C = 20$), we arrive at Table 1.

N	Best T, L	DBP Min cost	LBP Cost	% gain
2	2,3	23.83	23	3.50
5	2,7	24.58	23	6.41
10	2,13	24.82	23	7.89
20	2,26	24.94	23	7.79
30	2,38	24.98	23	7.94
40	2,51	25.00	23	8.01
50	2,64	25.02	23	8.06

Table 1: Comparison of packet transmission costs under DBP and LBP; ($C = 20$).

From Table 1, it is clear that the performance of LBP is better than the best performance achievable with DBP.

6.2 Lossy channel

When the channel is lossy, packets are received in error and retransmissions are required. Since data packets are usually appreciably larger than control packets like RTS, CTS, ACK etc, the probability of a data packet being in error is larger than that for control packets. We assume that the control packets are never lost.

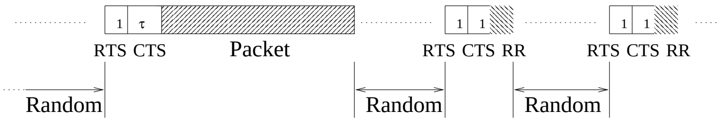

Figure 8: Events on the lossy channel; DBP (RR = repeat–request)

Consider the behavior of DBP. After a data packet is transmitted, the base and the receivers start contending for the channel. The receivers that are contending need to recover a packet that had been received in error earlier. The base tries to access the channel to transmit the next data packet. Under these circumstances, a sample path of events on the channel may look as in Figure 8. On this sample path, a packet transmission from the base is followed by two repeat–request transmissions from the receivers. The dotted portion represents the time intervals over which the base and the receivers were contending for the channel.

We note that the number of receivers contending for the channel now varies randomly, depending on the past history of the sample path. Thus, an analytical expression for the cost under DBP cannot be obtained in a simple way. However, to show that LBP performs better than DBP, we first find *a lower bound* of the cost under DBP. Then we show numerically that the cost under LBP is less than this lower bound.

A General Lower Bound of the cost under DBP

Consider Figure 8 again. Let the random interval before the transmission of the packet be denoted by X. Now the average of X, $E(X)$, cannot be less than $E(T_a^{DBP})$, because $E(T_a^{DBP})$ is the average length of the access phase under the *best* circumstances for DBP: (i) no receiver contending for the channel and (ii) no backing off for the base. Thus, $E(X) \geq E(T_a^{DBP})$. Also, the time interval immediately preceding a repeat–request transmitted by a receiver is *at least* 2 slots long: 1 slot for the unicast RTS from the receiver and 1 slot for the CTS from the base.

Let n_{av} be the average number of times that a packet has to be transmitted before all receivers get it error–free. Then, under DBP, the cost associated with a tagged packet is lower bounded by:

$$n_{av}(E(T_a^{DBP}) + C) + (n_{av} - 1)(D + 2), \tag{4}$$

where D is the size of the repeat–request packet transmitted by the receivers, and $(n_{av} - 1)$ is the average number of repeat–requests sent, assuming perfect repeat–request suppression.

We note that the above argument holds for a completely *general* loss model. The value of n_{av} will be different for different loss models.

Lower Bound of Cost under DBP versus Cost under LBP

Under LBP, however, the cost associated with a tagged packet is given simply by:

$$n_{av}(C + 3). \tag{5}$$

Now the value of $(E(T_a^{DBP}) + C)$ in expression 4 depends on the parameters L and T used in DBP. However, the *minimum* value of $(E(T_a^{DBP}) + C)$ is already available in the fourth column of Table 1 ("Min cost (DBP)"). From Table 1, we observe that[2]

$$(E(T_a^{DBP}) + C) \geq (C + 3).$$

Now from expressions 4 and 5, and the fact that $n_{av} \geq 1$, we find that the cost of a packet transmission under LBP is less than even the lower bound of the cost under DBP, for a perfectly general loss model.

To obtain an idea of the *minimum* improvement that can be expected, we consider a simple loss model in which losses seen by receivers are independent. The average number of transmissions required to ensure that all receivers receive a packet, n_{av}, for this simple loss model can be be found in [13]. Using the expressions from [13] and setting $C = 20$ and $D = 1$, we compare the cost under LBP with the lower bound under DBP in Table 2. The impact of spatial and temporal correlation in loss can be found in [11].

Loss			*Min* Cost	Cost	*Min* % Gain
Prob	N	n_{av}	DBP	LBP	LBP/DBP
0.05	10	1.43	36.69	32.82	10.55
	20	1.69	44.31	38.94	12.11
	30	1.86	49.10	42.83	12.79
	40	1.97	52.22	45.36	13.15
	50	2.05	54.35	47.08	13.38
0.10	10	1.76	45.90	40.43	11.91
	20	2.08	55.20	47.91	13.21
	30	2.25	59.99	51.77	13.70
	40	2.36	63.09	54.28	13.96
	50	2.44	65.47	56.21	14.14

Table 2: Cost under LBP compared with the *lower bound* of the cost under DBP; $C = 20$, $D = 1$.

The performance study in this section was motivated by the desire to compare DBP and PBP, that utilize standard probabilistic approaches to mitigate the CTS and ACK collision problems, and LBP that makes use of a leader to tackle these problems in a novel manner. The protocols were compared in a situation where multicast traffic is the only traffic present in the cell. From the columns "Cost under LBP" in Tables 1 and 2, it is clear that the throughput under LBP is higher than that under DBP (we recall that the throughput is the reciprocal of the cost). Being based on the probabilistic approach, DBP and PBP end up wasting channel bandwidth in trying to co–ordinate access to the channel. On the other hand, LBP provides efficient medium access and comprehensively outperforms the other protocols.

[2] This is yet to be formally proved.

Group Number	Link–level Address of Leader
G_1	A_1
G_2	A_2
...	...
G_i	×
...	...
G_k	×

Table 3: Group–leader table maintained at the base.

7 Leader Election

In this section, we discuss the leader election process. We assume that upon joining or leaving a group, a terminal sends explicit link–level join–group or leave–group messages to its base station.

Let $\mathcal{G} = \{G_1, G_2, \ldots, G_k\}$ be the set of possible groups to which a terminal may subscribe. The base station maintains a table containing each group and the corresponding leader (if any) as in Table 3. A × in the "Address" column means that the corresponding group has no leader. When the base starts up, the entire "Address" column contains ×'s.

When a terminal T sends a link–level join–group message to join group G_i (say), the base checks the table to find out if group G_i already has a leader. If it does, and T itself is not the leader, the base replies with the message that T will be a non–leader for group G_i. If group G_i does not have a leader already, then the base replies with the message that T will be a leader for group G_i.

When a terminal T sends a link–level leave–group message to leave group G_i (say), the base checks the table to see if T is the leader of group G_i. If T is not the leader, the base does nothing. If T is the leader, the base erases the entry in the column corresponding to G_i. In other words, we now have a × in the column corresponding to G_i.

However, a difficulty arises if the leave–group message sent by the leader leaving group G_i is not heard at the base station for some reason. Then, the base wrongly believes that G_i has a leader even though the leader has already signed off. In such a case, when the base sends out a multicast–RTS for group G_i, it will hear no CTS. After several unsuccessful attempts, the base will erase the leader entry corresponding to G_i, and stop forwarding packets addressed to this group. If there are other group members that are still interested in G_i, they will eventually time out and start the process of subscribing to group G_i afresh.

In order to maintain the group–leader table, the base station has to store two addresses per multicast group that is active in the wireless LAN. This amounts to about 10 bytes (4 bytes for IPv4 address and assuming that link layer address is 6 bytes long) of memory. Even if we assign a few more bytes for keeping any other state associated with a leader corresponding to a multicast group (for e.g., the number of unsuccessful retransmission attempts) only a few kilobytes of memory is required to maintain leader information of hundred multicast groups. We also note that it is possible to reduce the amount of control traffic flow for leader election purposes when a higher layer group management protocol like the IGMP (Internet Group Management Protocol, [7]) is running above the link layer. In this case, explicit *link–level* join–group messages may be suppressed, and leader election carried out by "snooping" IGMP packets. Under IGMP, receivers send explicit *IGMP–level* join–group messages upon joining a group. These join–group messages must pass through the base station. Hence, it is possible for the base station to become aware of one or more group members in the cell. The base–station can then assign one of these members the task of a leader by sending a message to this member.

8 Conclusion

In this paper we proposed a new approach for reliable multicast in a multi–access, cell–based wireless LAN. Our approach addressed two important issues, one of acquiring the wireless multi–access channel for multicast and the other of error recovery for reliability.

We proposed a leader–based protocol that deliberately allows responses from the leader and other members to possibly collide. We showed how the collision event itself can be used to convey retransmission requests.

The leader–based protocol provides very efficient solutions to the CTS and ACK/NAK collision problems. In addition, it is very simple to implement and can be potentially integrated with the current wireless LAN standard (IEEE 802.11). Comparison with traditional delayed feedback–based and probabilistic protocols showed the superior performance of the leader–based protocol. Simple mechanisms for leader election were also discussed. An emulation of the leader–based protocol is currently under consideration. In the future, we would also like to study the impact of mobility on the leader–based approach.

Acknowledgments

We would like to thank Professor Krithi Ramamritham for discussions and suggestions which served to influence and improve this work.

References

[1] A.S. Acampora and M. Naghshineh, "Control and Quality-of-Service Provisioning in High-Speed Microcellular Networks," *IEEE Personal Communications,* Vol. 1, No. 2 (Second quarter, 1994) ,pp. 36-43.

[2] V. Bharghavan, A. Demers, S. Shenker and L. Zhang, "MACAW: A Media Access Protocol for Wireless LANs," *Proceedings of ACM SIGCOMM Conference,* August 1994.

[3] V. Bharghavan, "A New Protocol for Medium Access in Wireless Networks," *Internal Technical Report,* University of Illinois at Urabana-Champagne, 1998.

[4] H. Chhaya and S. Gupta, "Performance Modeling of Asynchronous Data Transfer Methods of IEEE 802.11 MAC Protocol," *Wireless Networks,* Vol.3(1997) No.3, August 1997.

[5] B.P. Crow, I. Widjaja, J.G. Kim and P. Sakai, "Investigation of the IEEE 802.11 Medium Access Control (MAC) Sublayer Functions," *Proceedings of IEEE Infocom,* April 1997.

[6] A. DeSimmone, M.C. Chuah and O.C. Yue, "Throughput performance of transport layer

protocols over wireless LANs," *Proceedings of IEEE Globecom,* November 1993.

[7] W. Fenner, "Internet Group Management Protocol, Version 2," *RFC 2236,* November 1997.

[8] Wireless Medium Access Control and Physical Layer Working Group, "P802.11 IEEE Draft Standard-Wireless LAN," *IEEE Standards Department,* D3, January 1996.

[9] P. Karn, "MACA — A New Channel Access Method for Packet Radio," *ARRL/CRRL Amateur Radio 9th Computer Networking Conference,* September 1990.

[10] Sneha K. Kasera, Jim Kurose and Don Towsley, "Scalable Reliable Multicast Using Multiple Multicast Groups," *Proceedings of ACM Sigmetrics Conference,* June 1997.

[11] Joy Kuri and Sneha K. Kasera, "Reliable Multicast in Multi–access Wireless LANs," *Center for Electronic Design and Technology (CEDT) Tech Report,* Indian Institute of Science, January 1999.

[12] J. Mysore and V. Bharghavan, "A New Multicasting–based Architecture for Internet Host Mobility," *Proceedings of ACM Mobicom,* 1997.

[13] D. Towsley, J. Kurose and S. Pingali, "A Comparison of Sender-Initiated and Receiver-Initiated Reliable Multicast Protocols," *IEEE JSAC,* April 1997.

[14] R.T. Valadas *et al*, "The Infrared Physical Layer of the IEEE 802.11 Standard for Wireless Local Area Networks," *IEEE Communications Magazine,* December 1998.

[15] W. Stallings, "Local and Metropolitan Area Networks," 5th Edition, Prentice Hall, 1997.

[16] G. Xylomenos and G.C. Polyzos, "IP Multicast for Mobile Hosts," *IEEE Communications Magazine,* January 1997.