2009

An efficient and fair reliable multicast protocol for 802.11-based wireless LANs

Varun Srinivas
Iowa State University

An efficient and fair reliable multicast protocol for 802.11-based wireless LANs

by

Varun Srinivas

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Computer Science

Program of Study Committee:
Lu Ruan, Major Professor
Soma Chaudhuri
Wensheng Zhang

Iowa State University

Ames, Iowa

2009

DEDICATION

I would like to dedicate this thesis to mom, dad and my entire family for all their love and support. I would also like to thank my friends for their guidance and encouragement without which I wouldn't have been able to successfully complete this work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

The satisfaction that accompanies the successful completion of this thesis would be incomplete without the mention of the people who made this possible; whose constant guidance and encouragement were my driving force during the course of this work.

I consider myself privileged to express gratitude and respect to all those who guided me through the course of this thesis. I am extremely grateful to my major professor, Dr. Lu Ruan for her effective guidance and mentoring. Her constant support and encouragement right from the naescent stages of this work through its evolution has played a major part in the successful completion of this work. I would like to thank her for helping mould my ideas into a concrete piece of research work.

I would also like to thank my committee members, Dr. Soma Chauduri and Dr. Wensheng Zhang for all their guidance and support. I would also take this opportunity to thank the Department of Computer Science at Iowa State University for providing me a congenial environment to carry out my research in. I would also like to express my sincere gratitude to Aditya Dhananjay for kindling interest in me to carry out research in the area of Wireless Networks.

ABSTRACT

Many applications are inherently multicast in nature. Such applications can benefit tremendously from reliable multicast support at the MAC layer since addressing reliability at the MAC level is much less expensive than handling errors at the upper layers.

However, the IEEE 802.11 MAC layer does not support reliable multicast. This void in the MAC layer is a limiting factor in the efficacy of multicast applications. In this work, we propose a *Slot Reservation based Reliable Multicast protocol* that adds a novel reliability component to the existing multicast protocol in the 802.11 MAC. Our protocol builds on the existing *DCF* support in the IEEE 802.11 MAC to seamlessly incorporate an efficient reliable multicast mechanism. Intelligent assignment of transmission slots, minimal control packet overhead and an efficient retransmission strategy form the basis of our protocol. We evaluate the performance of our protocol through extensive simulations. Our simulation results show that our protocol outperforms another reliable multicast protocol, *Batch Mode Multicast MAC* in terms of delivered throughput in various scenarios.

We enhance our protocol to add a fairness component in the presence of parallel unicast and multicast flows and provide unicast friendly multicast operation. We then evaluate the performance of our *Slot Reservation Based Reliable Multicast Protocol with Fairness* through extensive simulations and see that the scheme ensures fairness among parallel unicast and multicast flows.

CHAPTER 1. INTRODUCTION

1.1 Overview of the Problem

Multicast is an efficient technique to disperse data to a group of recipients. By sending data to all the recipients simultaneously, multicast leads to significant savings in the usage of network resources and the time needed to disperse the data to all the recipients. A number of applications such as video conferencing, shared whiteboards, ground/air transportation networks, and military communication and control are inherently multicast in nature. Several popular routing protocols such as Dynamic Source Routing (DSR) [22][23] and Ad Hoc On Demand Distance Vector Routing (AODV) [24] rely on broadcast, which is a special case of multicast where the group of recipients includes all nodes the sender can communicate with. Works such as [20][21] describe the benefits of using multicast in several existing applications and how numerous applications in the future will benefit from a well defined multicast infrastructure.

IEEE 802.11-based wireless LANs are widely deployed in homes, offices, university campuses, and public areas. When stations in a wireless LAN are interested in receiving multicast data, they can benefit greatly from reliable multicast support at the MAC layer. Ensuring reliability at the MAC layer can significantly reduce the time and bandwidth spent in error recovery compared to handling errors in the upper layers. As a result, better end-to-end throughput and delay guarantees can be achieved. However, as described in the next chapter, the existing multicast technique in IEEE 802.11 [25] is unreliable. In addition, multicast frames are sent at the base rate of $1Mbps$ to increase the robustness of the communication. This means that we are not fully utilizing the bandwidth offered by the *802.11* MAC. Introducing reliability allows multicast frames to be sent at higher rates akin to unicast frames. In purview of the aforementioned benefits of reliable multicast at the MAC layer, this work aims at enhancing the *IEEE 802.11* protocol to include reliable multicast support.

1.2 Contribution of this work

In this work, we propose a *Slot Reservation based Reliable Multicast Protocol* that adds a novel reliability component to the existing multicast protocol in the 802.11 MAC. Our protocol builds on the existing *DCF* support in the IEEE 802.11 MAC to seamlessly incorporate an efficient reliable multicast mechanism. Intelligent assignment of transmission slots, minimal control packet overhead and an efficient retransmission strategy form the basis of our protocol. We also address the fairness issue in parallel unicast-multicast transmissions and provide a simple and elegant solution to tailor the level of fairness and throughputs obtained from the unicast and multicast flows.

The proposed efficient and fair reliable multicast protocol has the following features.

1. The reliability of multicast communication is achieved with RTS-CTS-DATA-ACK exchange. Using RTS and CTS control frames to capture the channel before sending multicast data is more efficient than recovering from an unsuccessful multicast because data frames are generally much longer than control frames and hence retransmitting data is more costly.

2. Efficient utilization of network bandwidth is achieved with a slot reservation based scheduling algorithm that schedules the transmission of CTS and ACK frames from different recipients to avoid collisions at the access point (AP).

3. The scheme also achieves fairness in terms of parallel operation of unicast and multicast transmissions by preventing multicast transmissions from starving unicast communications. By introducing multicast-free time periods, the scheme ensures that unicast transmissions receive a fair share of the bandwidth.

4. We propose several possible strategies for introducing multicast-free time periods which may be adopted based on the required level of fairness [17].

We simulate our protocol using the *ns-2* [36] simulator and provide comparative results with another efficient reliable multicast protocol, the *Batch Mode Multicast MAC* protocol [31], and show that our protocol outperforms it. We then add the fairness component to our simulation and show how varying levels of fairness can be achieved with parallel unicast and multicast transmissions.

1.3 Outline of this work

The rest of this thesis is organized as follows. We start with an overview of the *IEEE 802.11* [25] protocol, its architecture and its advantages in chapter 2. In chapter 3 we review relevant literature in the area. In chapter 4 we describe the *Slot Reservation Based Reliable Multicast Protocol* [39] and provide comparative simulation results. In chapter 5 we present the *Slot Reservation Based Reliable Multicast Protocol with Fairness* by providing enhancements to achieve *fairness* in the basic protocol described in chapter 4. We evaluate our fairness scheme under various scenarios and metrics. We end the thesis by providing conclusions from our work and outlining future work in chapter 6.

CHAPTER 2. OVERVIEW OF IEEE 802.11 PROTOCOL

2.1 Introduction

A wireless LAN (WLAN) as defined by [1] is 'a data transmission system designed to provide location-independent network access between computing devices by using radio waves rather than a cable infrastructure'. A wireless network is deployed typically as the final link between the wired network and mobile clients allowing these systems wireless mobile access to network resources [1]. The IEEE 802.11 was developed by by the IEEE LAN/MAN Standards Committee (IEEE 802) [25] as a set of standards for wireless local area networking, and currently is the de-facto standard in the area. 802.11 based wireless networks are extensively deployed in the corporate environment, educational institutions and homes making them virtually pervasive and ubiquitous.
802.11 based wireless networks operate in two major modes.

- *Infrastructure Mode:* Infrastructure mode 802.11 based wireless networks are characterized by the presence of *Access Points* (*APs*), which act as bridges between the wireless network and the wired backbone. Access Points themselves are connected using a wired backbone [12]. Wireless clients communicate with each other through these Access Points. Thus, communication between wireless station within an *AP's* realm takes place via the *AP*. This effectively doubles the bandwidth usage compared to the case where the stations directly communicate with each other. However, this is the most widely used category of WLANs, and this work is based on the infrastructure mode of operation.

- *Ad-Hoc Mode:* Ad-Hoc mode is characterized by stations directly communicating with one another. Networks can be set up and torn down without the need of any backbone or infrastructure. These networks are typically active for short periods of time, and torn down when they are no longer needed.

There are a number of benefits in using wireless LANs [2] as listed below.

- The most important benefit is increased mobility. The end user is no longer wired and hence makes mobile communication truly possible. In addition, as described above, the infrastructure mode bridges wired and wireless components of the network, thus enabling seamless integration of the two.

- Another important benefit is low cost and ease of deployment of wireless LANs [7][6]. Wireless network interface cards and access points are inexpensive devices and hence the cost of deployment and replacement is low. In addition, the placement of access points can be easily changed as required since they are small, handy devices. Deployment is extremely simple compared to its wired counterpart since physical obstacles have no effect on the placement of these devices.

- Wireless LANs are extremely useful in cost-effective network setup for hard-to-wire locations since the high cost of laying cables can be avoided.

- Wireless stations or APs can be added or removed without any disruption to the remainder of the system. Building scalable systems becomes possible because of this.

- Wireless LANs operate in the unlicensed frequency bands. This considerably reduces the cost of network operation since licensing is avoided.

2.2 The IEEE 802.11 Protocol Architecture

The *IEEE 802.11* protocol architecture is shown in Figure 2.1 [4]. The lowest layer is the physical layer which defines the operating frequency bands, the supported data rates and the details of radio transmission. *IEEE 802.11* comes in various flavors based on physical layer criterion of the operating frequency band and the modulation techniques used.

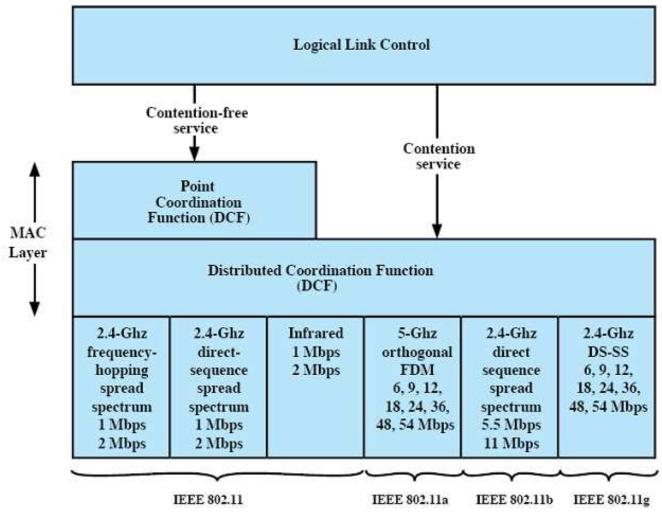

Figure 2.1 The IEEE 802.11 Protocol Architecture

We now outline the various flavors of the *IEEE 802.11* standard and some of their important features [3]. *802.11b* is the most popular and widespread of the various *IEEE 802.11* standards. It operates in the unlicensed $2.4GHz$ frequency band. It uses *DSSS* (Direct-sequence spread spectrum) modulation at the physical layer. It is capable of delivering a throughput of up to $11Mbps$; however the observed throughput is considerably lesser and is typically about $6Mbps$ since it faces interference from microwave ovens, cordless phones and other such devices. *802.11a* was an improvement on *802.11b*. It operates at a higher frequency (5 GHz) and avoids wireless interference. It is more vulnerable to signal loss through walls and other obstacles. Its operating range is smaller compared to *802.11b*. Theoretically it supports data rates of up to 54 Mbps. 802.11a equipment tends to be more expensive than 802.11b. *802.11g* was designed to be interoperable with *802.11b* while maintaining the high data rate achieved by *802.11a*. It provides *802.11a*'s higher bit-rate of up to $54Mbps$ in the $2.4Ghz$ band. The coverage area is better than *802.11a*. *802.11n* is the latest standard and can potentially deliver up to $600Mbps$, which is 50 times greater than *802.11b*, and 10 times greater than *802.11a* or *802.11g*. It is based on *MIMO* (Multiple Input Multiple Output) which comprises the use of multiple antennas at both the transmitter and receiver to improve communication performance.

Although the standard is expected to be finalized by December 2009, *802.11n* based cards are already in production.

Above the physical layer is the Medium Access Control *(MAC)* layer where in the heart of operation of *IEEE 802.11* lies. The MAC layer arbitrates access to the shared medium. The *IEEE 802.11* MAC is based on *CSMA/CA*, (Carrier Sense Multiple Access with Collision Avoidance) [26]. The basic idea is to avoid collision by not transmitting if the medium is busy thus ensuring that the transmitting wireless stations do not interfere with each other. The MAC layer has two sublayers, the *DCF* and the *PCF* and are described in section 2.3. The Logical Link Control Layer sits on top of the MAC and provides interface to the higher layers and performs basic Link level functions such as error control.

2.3 The IEEE 802.11 DCF and PCF

The *IEEE 802.11* standard [25] defines two medium access control (MAC) protocols, namely the Distributed Coordination Function (DCF) and the Point Coordination Function (PCF). These are described below.

2.3.1 The Distributed Co-ordination Function (*DCF*)

The DCF is the most popular mode of MAC operation in 802.11 and this work is based on DCF. The DCF uses a Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) [26] based scheme for its operation and is augmented with a RTS-CTS mechanism for collision free frame transfer.

2.3.1.1 Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA)

In this section we describe the CSMA/CA protocol used in *IEEE 802.11.* Any station wishing to transmit senses the channel for a period of time called the *DCF InterFrame Spacing* (DIFS). If the medium is idle for DIFS, the station transmits its frame. If the medium is busy, the station continues to listen until the medium is idle for DIFS and then backs off a random number of slots of time chosen within the size of its contention window $[0, CW]$ where each slot is equivalent to one slot time (typically 20 μs). If the channel becomes busy before the backoff timer expires, the timer is frozen. The station continues to listen to the medium until the medium is idle for a period of DIFS and the backoff timer is restarted. The stations transmits its frame once the backoff timer expires.

After transmitting the data frame, the transmitting station expects an ACK from the receiver within a fixed period called the ACK timeout. If the station receives an ACK within the ACK timeout period, the transmission is deemed complete and successful. Else, the transmission is unsuccessful and the station attempts to retransmit the frame. Every time a retransmission attempt is made, the size of the contention window is doubled up to a maximum of CW_{max}, which is the maximum possible size of the contention window. This operation is called the *Binary Exponential Backoff* scheme.

2.3.1.2 The Hidden Terminal Problem

The CSMA/CA protocol described in section 2.3.1.1 is known to suffer from the *Hidden Terminal* problem. Consider three stations A, B, and C where A and C are within B's transmission range but A and C are outside each other's transmission range. Suppose node A wants to transmit a frame to node B while B's neighbor, C, is transmitting. Node A will find the medium idle and transmit the frame, causing collisions at B between its frames and frames from C.

The RTS-CTS mechanism was proposed to circumvent this problem and is described in the next section.

2.3.1.3 The RTS (Request to Send) - CTS (Clear to Send) Operation

If the channel is found to be free for a period of *DIFS*, the station sends out an RTS frame containing the receiver's MAC address and the time duration it would require for the transmission. Any station in the transmission range of the sender will see the RTS frame. When the receiver receives the RTS, it checks to see if it can accommodate the transmission request. If so, it responds by sending a CTS frame to the sender. Any station in the transmission range of the receiver will see the CTS frame. When a station sees an RTS or CTS, it realizes that some station is going to occupy the medium for the duration of time specified in the frame and therefore defers its transmission. The RTS and CTS work together to ensure that the areas in the transmission range of the sender and the receiver are clear of any parallel transmission that might overlap with the transmission they are guarding. Upon receipt of the CTS, the sender transmits the data frame. Upon successful reception of the data frame, the receiver sends an ACK. Upon an ACK timeout, the sender retransmits the corresponding data frame until it hits the retry limit.

If the intended receiver of an RTS sees that it will not be able to accommodate the requested transmission, it does not send a CTS back to the sender. Upon non receipt of a CTS from the sender (based on a timeout), the sender realizes that the intended receiver is unable to process the request at this time and performs *Binary Exponential Backoff* before contending for the medium again.

2.3.2 The Point Co-ordination Function (*PCF*)

The point co-ordination function is designed to provide contention free frame transfer service for time bound transmissions. In *PCF*, a point coordinator in the access point controls transmission of frames from stations. It controls medium access, by determining which station is allowed to access the medium at any point of time. The point co-ordinator can enter into *contention free* periods and control transmissions when required by gaining control of the medium. The *contention* period is simply the DCF operation. The point coordinator senses the medium at the beginning of each contention-free period and if the medium is deemed to be free for a specified period of time, the *PIFS* (*PCF* Inter Frame Spacing), the point co-ordinator sends out a *beacon* frame with the duration of the *contention free* period and all stations defer their attempt to grab the channel till the expiration of the *contention free* period. The point co-ordinator typically implements a round robin scheduling scheme.

2.4 Multicast support in IEEE 802.11

The point of interest for this work arises from the fact that the *RTS-CTS-ACK* exchange and the *Binary Exponential Backoff* algorithm is defined only for *unicast* transmission, i.e. transmission to a *single* receiver. The semantics for *broadcast* (transmission to all stations) and *multicast* (transmission to a group of stations) are completely different. Hereafter, we treat *broadcast* as a special case of *multicast* where the multicast group includes all stations in the purview of an Access Point's service group. To make a multicast transmission, the sender (the AP) senses the medium for a period of *DIFS*. If the medium is found to be free for this period of time, it transmits the multicast frame. The *RTS-CTS* mechanism is not used. Thus, the scheme doesn't check if the receivers are busy, or if they have interfering transmissions going on in parallel. In addition, the destination stations do not respond with ACKs after they receive the multicast frame. Thus, the sender does not know whether the intended receivers received the multicast frame. This means that reliability is not ensured for multicast transmission.

As described in the previous section, for unicast transmissions the size of the contention window is doubled every time there is an unsuccessful attempt of an RTS transmission. The size of the contention window is doubled up to a maximum value of CW_{max} after which the frame is eventually dropped. However, no such scheme exists in case of multicast transmissions. If the medium is free for a period of DIFS, the station transmits its frame and the transmission is deemed complete. If the medium is not free for a period of DIFS, the transmitting station waits until the medium is free for a period of DIFS and backs off for a fixed period of time (typically CW_{min}) and there is no increase in the size of the contention window since there is no concept of ACK in multicast. The backoff timer is frozen when the channel becomes busy and restarted after the channel is deemed free for a period of DIFS similar to unicast transmissions. This unfairness in terms of the sizes of the backoff windows between unicast and multicast transmissions shows a marked effect when there are simultaneous unicast and multicast transmissions [37], which will be dealt in detail in chapter 5 where we incorporate fairness into the scheme we propose.

Another drawback of the native *DCF* multicast algorithm is that multicast frames are transmitted at the base rate of *1 Mbps* to increase the robustness of the communication, even though *802.11b* can support data rates of up to *11 Mbps* and *802.11a/g* up to *56Mbps*. This means that *rate adaptation* [34][13][19] where in senders dynamically adapt their transmission rate based on channel conditions is void.

To address the aforementioned problems of DCF multicast, we propose a slot reservation based reliable multicast protocol in chapter 4 and enhance it to incorporate fairness as described in chapter 5.

CHAPTER 3. LITERATURE REVIEW

In this chapter we review some of the relevant literature in the area. The schemes for reliable multicast in *IEEE 802.11* can be broadly classified into two categories based on whether or not the scheme uses an additional signaling interface. We review the basic ideas, advantages and drawbacks of schemes presented in [27],[28], [29], [30] and [31] in the single interface category and [33] which uses an additional signaling interface. We then review [37] which introduces fairness when there are parallel unicast and multicast flows where in native unreliable DCF multicast protocol is used. At the time of writing this thesis we have come across no prior work that provides a reliable multicast solution which is also fair to parallel unicast flows.

3.1 Schemes for Reliable Multicast

3.1.1 The Broadcast Medium Window *(BMW)* protocol [28]

3.1.1.1 Main Idea

This is a protocol designed to support reliable multicasting in wireless Ad-hoc networks. We review this paper since it does some fundamental work in introducing reliability in wireless MAC multicast and can be extended to infrastructure networks. Each node maintains a NEIGHBOR LIST of all its neighbors. An entry is purged off it if a node hasn't been heard from for a specified time. Each node also maintains a SEND BUFFER that stores frames that were already sent but haven't been acknowledged by all stations. A frame is purged from the SEND BUFFER after all neighbors have received it. Each node also maintains RECEIVER BUFFER where in it maintains the sequence number of each received frame. A transmitting node sends the range of frame sequence numbers in that transmission. Each destination node checks its RECEIVER BUFFER to determine if there are any frames missing in the range. If so, the destination node replies with the missing sequence number in the CTS response (the start sequence number of the least numbered unreceived frame). When a sender has to transmit, it goes into the collision avoidance phase similar to DCF. It then sends RTS to one of its neighbors, specifying what sequence numbers were already transmitted and what the current sequence number is. Upon receiving the RTS, the intended neighbor examines its RECEIVER BUFFER and specifies the frames it needs in its CTS. All other neighbors hearing the RTS will wait for this CTS-DATA-ACK sequence to finish. After the reception of the CTS, the source transmits data, and neighbors back off until the ACK has been transmitted. Upon receiving the DATA, the destination node updates its RECEIVER BUFFER and replies with an ACK. In the meantime neighboring nodes that received the DATA will also update their RECEIVER BUFFER. If the DATA sent to a receiver was obtained from the SENT buffer, transmission is continued until the current data is sent; collision avoidance is omitted in this case. The source node then buffers the current packet and chooses the next neighbor in its NEIGHBOR LIST until all neighbors have received the current frame. This is the basic mode of operation of the protocol. This formed one of the earlier significant works in the area.

3.1.1.2 Advantages

- The protocol ensures completely reliable multicast.

3.1.1.3 Drawbacks

- The protocol involves a large number of contention phases. There can be a maximum of n contention phases for a multicast group size of n stations. This makes the protocol inefficient and unsuitable for delay intolerant applications.

- Modification to existing frames is needed.

3.1.2 MAC Layer Broadcast Support [29] [30]

3.1.2.1 Main Idea

In [29] after executing the collision avoidance phase, the source sends RTS to all neighbors and waits for WAIT-FOR-CTS time for a CTS. Neighbors of source send CTSs if they are not in the YIELD state and wait for WAIT-FOR-DATA time for data. If source receives a CTS it sends its data frame. Else,on expiry of the WAIT-FOR-CTS timer, it back off and goes back to contend for the medium. Nodes that are not involved in the broadcast exchange, upon receiving CTS, set their state to YIELD and wait for the broadcast operation to finish.

[30] enhances the operation of [29] to improve reliability. In addition to the steps described for [29], neighbors of the sender send NAK if WAIT-FOR-DATA timer expires and data has not been received. If source receives a NAK before the WAIT-FOR-NAK period, it goes back to retransmit its frame. Else, the broadcast is considered complete.

3.1.2.2 Advantages

- These protocols are very simple extensions to *IEEE 802.11* multicast/broadcast.

3.1.2.3 Disadvantages

- The protocols do not ensure completely reliable multicast.

3.1.3 The Leader based protocol [27]

3.1.3.1 Main Idea

This protocol takes a leader based approach to solving the reliable multicast problem. The leader is in charge of sending CTSs and ACKs on behalf of the group. In this approach, a leader is elected for a multicast group, and only the leader sends a CTS to the sender. Other stations remain silent if they see that the transmission is feasible from their standpoint, else send a NCTS. If no NCTS was received the sender goes ahead and sends data. A similar scheme works for ACK and NACK, where only the leader sends an ACK if data was successfully received. Other stations only send NACKs if they had problems in receiving the data frame.

The protocol abstractly works in terms of slots.

- Slot 1: The Access Point sends multicast RTS.
- Slot 2: The leader sends a CTS if it is ready to receive data. Other stations in the group remain silent if they are ready to receive data. Else they send NCTS (not clear to send).
- Slot 3: If CTS was heard in slot 2, the Access Point starts multicast data operation. Else it executes the backoff scheme and starts from slot 1.
- Slot 4: After the Access Point has transmitted data, the leader sends an ACK if it received data correctly. Else it sends a NACK. Other stations remain silent if they received data correctly. Else they send NACK. The basic idea here is that if at least one station sends a NCTS or a NACK, it either collides with the CTS/ACK sent by the leader if it sent one, or the NCTS/NACK reaches the sender. In either case, the transmission is considered unsuccessful. If the leader didn't sent a CTS/ACK or the NCTS/NACKs sent by multiple stations collide and do not reach the sender, the transmission is considered unsuccessful in which case it retries after a timeout.
- Slot 5: If a ACK was heard in slot 4, the transmission is considered complete. Else, the access point retransmits the multicast RTS in slot 1.

3.1.3.2 Issues with Leader Selection

This scheme has several drawbacks based on leader selection. First, a new leader needs to be chosen every time the current leader leaves the network. Second, an intelligent leader selection algorithm is needed to choose an appropriate leader. For example, choosing a leader which is very close to the sender compared to another node could lead to a case where when the leader sends its CTS and the other node sends its NCTS at the same time, the signal from the leader reaches the sender at a higher strength, hence suppressing the NCTS from the other node. Leader selection scheme should also be based on the current load distribution. Selecting a station whose neighboring traffic or interference is lesser compared to other multicast member stations as a leader means that there is a lesser chance of a NCTS or a NACK successfully reaching the Access Point compared to the CTS or ACK from the access point. Such a selection becomes more difficult in the presence of mobility and variable traffic and network conditions.

3.1.3.3 Advantages

- Reduced number of contention phases; a single contention phase in the best case.
- Control frame overhead is minimal.

3.1.3.4 Drawbacks

- As evident from section 3.1.3.2, leader selection/election algorithm is critical .
- The protocol cannot ensure 100% reliability in the following cases.
  - Criterion such as capture effect [38] introduce unreliability.
  - The protocol doesn't work when the RTS reaches certain stations and not others. There is no way of ensuring that everyone received the RTS.
  - The protocol fails when a NCTS or a NACK is lost and the leader sends a CTS or an ACK. The failure of the non-leader is suppressed and the multicast is deemed successful.
- Use of new control frames namely NCTS and NACK.

3.1.4 Delayed Feedback Based *(DFB)* and Probabilistic Feedback based *(PFB)* protocols [27]

3.1.4.1 Main Idea

In the *DFB* protocol, random timers are used to avoid CTS/ACK collisions. The Access Point sends a multicast RTS and waits for a CTS timeout period to receive all CTSs, else backs off and retransmits. On hearing the RTS, stations start a countdown of a random number of backoff slots and decrease timer by 1 in each slot. If a station hears a CTS before its timer expires, it freezes its counter, called CTS suppression. Else it sends a CTS at the end of the timer expiration. If the Access Point doesn't hear a CTS by the expiration of the CTS timeout, it backs off and tries to send the RTS again. Else it starts multicast transmission. If the stations receive data in error, they send a NAK after contending for the channel to avoid collisions among NACKS. In the *PFB* protocol, instead of sending the CTS after a countdown period, the group members send out a CTS in the slot immediately following the RTS with a certain probability based on the number of stations. The stations can send a NCTS with a probability *1* if they are not ready to receive data. If a NCTS was not heard in the first slot, the Access Point waits for either a CTS in one of the following slots or its CTS timer to expire. The rest of the protocol operation is similar to *DFB*.

3.1.4.2 Advantages

- Simple to implement since they are variations of DCF operation.

3.1.4.3 Drawbacks

- Time taken for a RTS-CTS exchange can be considerably large since CTS collisions are possible.
- Since they are NAK based, link level buffering requirements are high at both the Access Point and the receivers for retransmission and sequencing purposes respectively.
- Choosing ideal wait times or probabilities is not trivial.
- 100% reliablility cannot be ensured.

3.1.5 The Batch mode multicast MAC *(BMMM)* protocol [31]

3.1.5.1 Main Idea

Perhaps the most interesting work in this area, and the approach our work is based on is the Batch Mode Multicast MAC (BMMM) proposed in [31]. In BMMM, in order to send a multicast frame, the sender sends RTSs to each station individually and waits for CTSs from each of them. Upon receipt of CTSs from all intended recipients, the sender goes ahead and sends the data frame. Then, it sends a special frame called *RAK*, Request for ACK, to each of the stations serially, and each station responds to the *RAK* with an ACK. Upon receipt of ACKs from all intended recipients, the transmission is deemed complete. If there were stations who did not send ACKs, the sender again contends for the medium and repeats the above procedure, although this time, the recipient set is the subset of stations whose ACKs were not received. BMMM is a simple and rather efficient scheme to achieve reliable multicast in *IEEE 802.11*. We design our slot reservation based reliable multicast protocol based on BMMM, but takes a slightly different approach to improve the efficiency of multicast.

3.1.5.2 Advantages

- The scheme can ensure reliable multicast.
- The scheme involves a single contention phase.

3.1.5.3 Drawbacks

- Excessive control frame overhead. n pairs of RTS-CTS and RAK-NAK for a multicast group of size n.
- Use of a new control frame, *NAK*.

3.1.6 The 802.11 MX (A busy tone based protocol)[33]

3.1.6.1 Main Idea

The protocol in [33] requires each node to have an additional busy tone interface. Busy tones are used to signal NCTSs or NACKs instead of sending packets. The advantage here is that, even if multiple stations signal NCTSs or NACKs simultaneously, it is alright since it is only a tone. The protocol functions as follows. The sender executes the contention phase and then transmits an RTS. It then listens on its signaling channel (busy tone) to see if any station is transmitting an NCTS tone. If not, it goes ahead with data transmission and then senses its signaling interface to check if any station has set a NACK tone. If not, the transmission is deemed successful. Else the sender goes into contention and retransmits multicast data repeating the procedure described above.

3.1.6.2 Advantages

- The scheme ensures reliable multicast.

3.1.6.3 Drawbacks

- Use of an additional signaling interface is inconsistent with the *IEEE 802.11* standard and is difficult to implement in existing deployments.
- Extensive modification of the MAC protocol.

In table 3.1 we summarize some of the major drawbacks of existing schemes for reliable multicast.

3.2 Schemes for Fairness

There is a single significant work that addresses the issue of fairness among simultaneous unicast and multicast flows using native *IEEE 802.11* multicast. We review the work here. At the time of writing this thesis, there were no schemes that addressed unicast-multicast fairness in the presence of reliable multicast operation.

3.2.1 Unicast-Friendly Multicast in IEEE 802.11 Wireless LANs [37]

3.2.1.1 Main Idea

The main idea here is to achieve fairness between unicast and multicast flows by using the concept of Unicast-Friendly Multicast (UFM). In this scheme, the contention window size for multicast flows is dynamically adjusted with an aim to limit the bandwidth share of a multicast flow equal to that of a unicast flow. The scheme adjusts contention window size

Scheme	**Drawbacks**
Broadcast Medium Window Protocol [28]	• Large number of contention phases. n contention phases for a multicast group of size n. • Inefficient; not suitable for delay intolerant networks. • Modification to existing frames needed.
Batch Mode Multicast MAC Protocol [31]	• Use of new control frame, the RAK. • One RTS frame per CTS and one RAK frame per ACK. Excessive overhead of control frames.
MAC Layer Broadcast Support Protocols [29][30]	• These protocols do not ensure 100% reliable delivery.
Leader Based Protocol [27]	• Leader election is inherently difficult. • Use of new control frames namely NCTS and NACK. • The protocol doesn't ensure 100% reliable delivery.
Busy Tone Based Protocols [33]	• Use of an additional signaling interface is difficult to implement in existing deployments and requires hardware. • Extensive modification of the MAC protocol.
Probabilistic and Delay Feedback based protocols [27]	• Time taken for a RTS-CTS exchange can be considerably large since CTS collisions are possible. • Since they are NAK based, link level buffering requirements are high at both the Access Point and the receivers for retransmission and sequencing purposes respectively. • Choosing ideal wait times or probabilities is not trivial. • Cannot ensure 100% reliable delivery.

Table 3.1 Reliable Multicast schemes and their drawbacks

for multicast based on the number of competing stations. Two versions of the scheme are presented. In the first version, each multicast node infers packet collision probability of its multicast frames based on the estimate of the number of other competing stations. Assuming the knowledge of average number of packet collisions until successful transmission, it virtually performs binary exponential backoff like a unicast station until it reaches the inferred average backoff stage called *virtual backoff*. This is based on the assumption that the average number of packet collisions until successful transmission is known apriori. At each backoff stage, it picks up a random number from its contention window corresponding to that of a unicast flow. At the final stage, the size of the contention window is equal to the recursive sum of all the selected backoff times, each of which is multiplied by its collision probability. The station then performs a backoff within this interval. In the second version, each station maintains a table of multicast contention window sizes for different number of competing stations such that on adopting the specified contention window size, the the bandwidth share of the multicast flow becomes equal to that of a competing unicast flow.

3.2.1.2 Advantages

- The scheme ensures fairness among unicast and multicast flows.

3.2.1.3 Drawbacks

- The scheme doesn't work with reliable multicast schemes as described in section 5.2.

After reviewing several existing schemes, we now enlist some of the desirable features that a reliable multicast MAC protocol which also ensures fairness should possess. Our protocol incorporates all the features mentioned below.

- The protocol should support complete reliability in terms of delivery.
- The number of contention phases should be minimized.
- As far as possible, new control frames should not be introduced.

- The protocol should be designed for a single interface to enable seamless integration with the existing standard.
- The time spent in control frame exchange as compared to data transmission must be minimized.
- The protocol should ensure unicast-multicast fairness in the presence of reliable multicast.
- The comparative throughputs of unicast and multicast flows must be adjustable to provide required degree of fairness. In other words, the protocol should be able to provide user requested degree of fairness.

CHAPTER 4. THE SLOT RESERVATION BASED RELIABLE MULTICAST SCHEME

4.1 Introduction

We propose an efficient reliable multicast protocol with the following features.

1. The reliability of multicast communication is achieved with RTS-CTS-DATA-ACK exchange. Using RTS and CTS control frames to capture the channel before sending multicast data is more efficient than recovering from an unsuccessful multicast because data frames are generally much longer than control frames and hence retransmitting data is more costly.

2. Efficient utilization of network bandwidth is achieved with a slot reservation based scheduling algorithm that schedules the transmission of CTS and ACK frames from different recipients to avoid collisions at the access point (AP).

4.2 The Basic Idea

The proposed SRB protocol uses the *RTS-CTS-DATA-ACK* exchange to ensure reliable multicast. To send a multicast frame, the AP first sends an RTS frame to the multicast group address. A station in the multicast group responds with an CTS if it can accommodate the transmission request. After the AP receives the CTSs from all multicast group members, it transmits the multicast data frame. A station in the multicast group responds with an ACK if it successfully receives the data frame. Clearly, the stations in the multicast group should not transmit their CTSs or ACKs simultaneously, or else collision will occur at the AP. Thus, there needs to be a mechanism to coordinate the transmissions of the CTSs and ACKs from different stations to avoid collision at the AP. Our solution is to schedule the transmissions from different stations in a non overlapping fashion. This concept of *scheduling* lies at the heart of our protocol.

4.3 The AID and $MAID$

Before going into describing the actual solution, we introduce the concepts of *Association ID* (AID) and *Multicast AID* (MAID) which help us establish the *schedule* of transmissions of the multicast receivers.

Upon successful association of a station with an AP, the station receives from the AP, among several other parameters, a parameter called the *Association ID* (AID) as a part of AP's *Association Response* frame. An AID is a number between 1 and 2007 [35]. It is unique within the set of stations associated with the AP. It is primarily used in the *Powersave* mode [25] to deliver frames buffered at the AP while the station is in a low power (sleeping) state. We will use the AID concept to arrive at a serialized schedule for broadcast communication. We impose the following two constraints on the issue of AIDs.

1. Before issuing an AID to a station, the issued AID set is examined to see if there are unused AIDs resulting from the disassociation of stations that existed before. If such AIDs are found, the smallest such AID is issued.

2. The AIDs shall be issued in increasing order starting from AID 1.

To derive a serialized schedule for multicast communication, we make use of *Multicast AIDs* (MAIDs). If a station subscribes itself to a multicast group, the AP issues a *Multicast AID* (MAID) which uniquely identifies the station within its multicast group. The rules for issuing a MAID remain the same as described for AID. Why we require a MAID when a station can be uniquely identified by its AID, is simply for efficiency. Further details are based on the operation of the protocol itself, and shall be provided in section 4.5.

4.4 Proposed Solution

Consider an AP and a set of n stations associated with the AP such that these n stations make up a multicast group G. Whenever an AP wants to send a multicast frame to G, it first executes the contention phase exactly as in *DCF*. Once it gains access to the medium, it sends out an RTS for multicast. The receiver address field of the *RTS* frame contains the multicast group address of G. The time duration in the RTS frame is the time required to transmit n CTSs, the data frame, and n ACKs. A station, on seeing that it belongs to the multicast group G, transmits a CTS if it can accommodate the transmission request. The CTS is transmitted in a time slot determined by a simple rule. A station with *MAID* i transmits in the i^{th} time slot. A CTS is always transmitted at the base rate of *1 Mbps*. Since the CTS frame size is fixed, the time required for a CTS transmission is fixed, denoted by T_{CTS}. Hence, a station with *MAID* i transmits starting at time $(i-1) * T_{CTS}$ from the instance of reception of the RTS. After the AP sends out the RTS, it waits for nT_{CTS} and then transmits the data frame. Once the data frame has been received, each station transmits its ACK at time starting $(i-1)*T_{ACK}$ from the instance of reception of the data frame, where T_{ACK} is the time required for the transmission of an ACK. In case of broadcast, stations will use their AIDs instead of MAIDs to determine the transmission time of the CTS and ACK frames. This forms the first and compulsory phase of our protocol.

It is well possible that not all stations respond with CTSs and/or ACKs. These control frames might also be lost. Hence we need some way of retransmission to such stations to ensure reliability, and on top of it, we need to take efficiency into consideration when performing retransmissions. A straightforward extension to our scheme for retransmissions is to retransmit the RTS just like before. However, the sender now expects CTSs and ACKs only from those stations whose transmissions were unsuccessful in the previous attempt. Repeating this scheme until all stations have received the data frames successfully would ensure reliability. Although simple to implement, this scheme is highly inefficient. The inefficiency arises from the fact that only the CTS and ACK time slots for those stations which require a retransmission are really useful. Those slots corresponding to stations whose transmissions were successful in a previous phase go waste. Hence, we propose a modification of the protocol presented above in case of retransmissions.

For each retransmission phase, we establish an order of transmissions among stations participating in that particular retransmission phase using a modification to the RTS multicast frame sent out at the beginning of each retransmission phase. The RTS frame is appended with a bitmap with n bits, where n is the number of stations in the multicast group. There is a one-to-one mapping from bits in the bitmap to the *MAID*s of stations in the multicast group, i.e., *bitmap[i]* corresponds to station with *MAID i*. The bits corresponding to stations participating in the current retransmission phase are set. In other words, $bitmap[i] = 1$ iff station with *MAID* i is a participant in this retransmission phase. Looking into this bitmap, the stations can determine their transmission slots for CTS/ACK as follows. The first bit position which is set corresponds to the station that has to transmit in the first slot (recall the one-to-one mapping between bit positions and *MAID*s). The second bit position that is set corresponds to the station that should occupy the second slot and so on. Specifically, a station with *MAID* i should occupy slot j if $bitmap[i]$ is the j^{th} bit that is set. Thus, with relatively small increase in the size of the RTS, in any retransmission phase, we effectively schedule only those stations who are participating in the current phase, and thus avoid the inefficiency described earlier. The retransmission phase is repeated until all stations have successfully received the data frame, or we have reached a specified *retry* limit. These two phases make up our protocol. In the next section, we provide an algorithmic description of the two phases.

4.5 The Algorithm

4.5.1 The Initial Transmission Phase

1: AP sends RTS reserving time for n CTS slots, data, and n ACK slots where n is the number of stations in the multicast group.

2: Station S_i ($i = 1$ to n) transmits CTS in the i^{th} slot if feasible.

3: At the end of the n CTS slots, the AP sends *DATA*.

4: Station S_i ($i = 1$ to n) transmits ACK in the i^{th} slot if feasible.

5: If AP received n CTSs and n ACKs, *END*. Else enter *Retransmission Phase*.

4.5.2 The Retransmission Phase

1: Construct the modified RTS frame with the bits corresponding to stations whose transmissions were unsuccessful in the previous phase set. The RTS frame reserves time for n' CTS slots, data, and n' ACK slots where n' is the number of stations participating in this phase.

2: Station S_i ($i = 1$ to n) transmits CTS in the j^{th} slot if *MAID* i is the j^{th} bit set in the bitmap where $1 \leq j \leq n'$.

3: At the end of the n' CTS slots, the AP sends *DATA*.

4: Station S_i ($i = 1$ to n) transmits ACK in the j^{th} slot if *MAID* i is the j^{th} bit set in the bitmap where $1 \leq j \leq n'$.

The retransmission phase is executed until all stations have received the data frame successfully, or a *retry* limit is reached.

We are now in a position to answer the question we stated about the manner in which we issue *MAID*s. We issue *MAID*s in a serial fashion starting from 1, since the *MAID*s have a one-to-one mapping to the transmission slots of stations. Issuing continuous *MAID*s ensure that our scheme is efficient. We also check if there are *MAID*s freed in the set of *MAID* starting from 1 to the maximum issued *MAID* till the current time since it is possible that a station previously joined a multicast group leaves the group, and hence its *MAID* becomes unused.

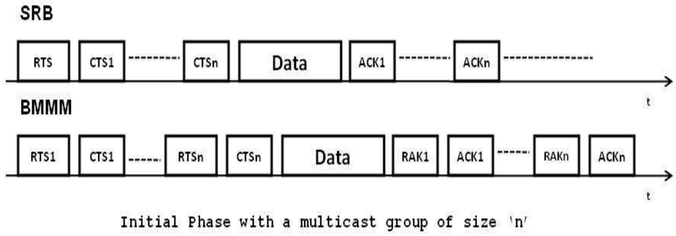

Figure 4.1 Comparison of the initial transmission phase.

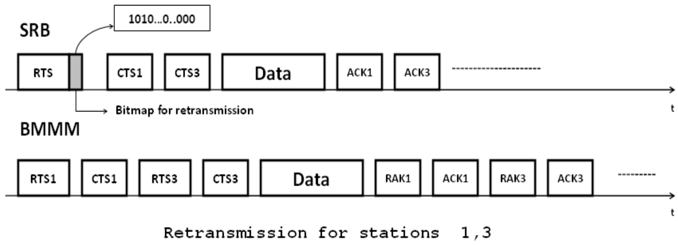

Figure 4.2 Comparison of the retransmission phase.

By utilizing all such *MAID*s before issuing new ones, we minimize the time slots wasted due to the occurrence of such events.

4.6 Comparison with BMMM

Fig. 4.1 represents a timeline comparison of the initial transmission phase of the *SRB* and the *BMMM* schemes. The timelines represent combined activity of the AP and n multicast receiver stations. In the case of *BMMM* the timeline begins with transmissions of RTS-CTS pairs for each of the n stations in the multicast group. Assuming that all stations sent CTSs, data is then transmitted. This is followed by a phase of RAK-ACK exchanges. In case of the SRB scheme, the timeline begins with a single RTS transmission followed by CTS replies from all stations in the multicast group. Then there is a data transmission phase where the AP sends its data. This is followed by a phase of ACK transmissions from all stations in the multicast group.

We now compare the transmission time of BMMM and SRB in the initial transmission phase. We have

$T_{BMMM} = (T_{RTS} + T_{CTS}) * n + T_{Data} + (T_{RAK} + T_{ACK}) * n$

$T_{SRB} = T_{RTS} + T_{CTS} * n + T_{Data} + T_{ACK} * n$

Therefore, $T_{BMMM} = T_{SRB} + ((n-1) * T_{RTS} + n * T_{RAK})$. That is, SRB achieves a saving of $(n-1) * T_{RTS} + n * T_{RAK}$ in the initial transmission phase.

Fig. 4.2 represents a timeline comparison of the retransmission phase of the *SRB* and the *BMMM* schemes, assuming station 1 and station 3 did not successfully receive the data in the initial transmission phase. For BMMM, there are two RTS-CTS and two RAK-NAK exchanges for stations 1 and 3. In the case of SRB, a modified RTS with the bitmap of station *MAID*s is sent, where the first bit and the third bit of the bitmap are set. This is followed by CTS transmissions from station 1 and station 3. Following this, data is transmitted by the AP. Then, station 1 and station 3 send their ACKs.

For a retransmission phase with k participating stations ($k \leq n$), we have $T_{BMMM} = T_{SRB} + ((k-1) * T_{RTS} + k * T_{RAK})$ assuming the time to transmit an RTS with the bitmap is about the same as T_{RTS}. Hence, SRB achieves a saving of $(k-1) * T_{RTS} + k * T_{RAK}$ in the retransmission phase.

Compared with BMMM, SRB is absent of multiple RTS-CTS and RAK-ACK frame exchanges. The former is replaced by a singe RTS followed by a CTS sequence while the latter is replaced by a series of ACK responses alone.

4.7 Simulation Scenarios and Results

We simulated our SRB protocol using the *ns-2* simulator [36][14]. We modeled a *802.11b* network which is capable of delivering up to *11Mbps* as our basic network topology with a single AP and 25 stations associated with it. The AP was set up to generate Constant Bit Rate (CBR) traffic with data packets with varying rates and sizes as required for specific experiment scenarios. We then compared the performance of our *SRB* scheme with the *BMMM* protocol under the influence of various controlling factors. The results from the experiments

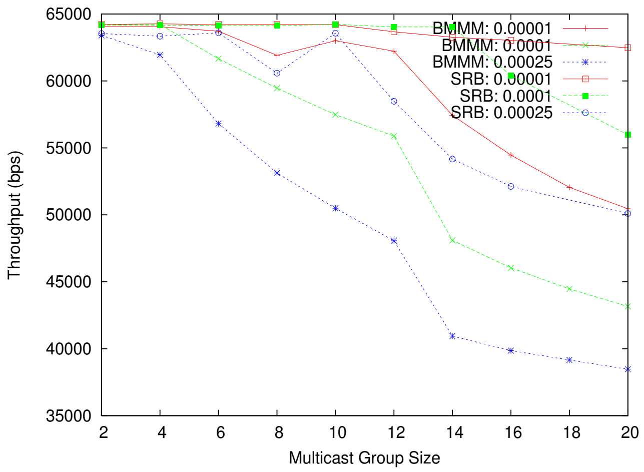

Figure 4.3 BMMM and SRB throughput vs. multicast group size for varying bit error rates.

are presented below.

4.7.1 BMMM and SRB throughput vs. multicast group size for varying bit error rates

Fig.4.3 represents a graph of throughput from the *BMMM* and the SRB protocols for varying Bit Error Rates *(BER)*. We fixed the traffic generation rate at 512 packets per second. The length of each packet is fixed at 1024 bytes. We plot the throughput under various *BER*s for multicast groups ranging from size 2 to 20. From the graph, we see that as the *BER* increases, the throughput of *BMMM* with respect to a given multicast group size decreases. For example, for a *BER* of 0.00001 the observed throughput for a multicast group of 12 stations is $63Kbps$ and reduced to $56Kbps$ and $48Kbps$ as the error rate is increased to 0.0001 and 0.00025 respectively. This occurs since a higher *BER* means more packets in error and hence more retransmissions. We also observe that for a given *BER* the throughput drops with increasing number of stations. This is expected since the number of control frames transmitted and hence the transmission time per data frame increases with increasing number of stations. As a consequence, the time spent in backoff periods also increase. As a result, a newly generated packet will have to wait for a longer period of time before it can be transmitted.

In case of the *SRB* protocol, a similar phenomenon to what was observed with *BMMM* is seen. The throughput decreases with increasing bit error rates and increasing multicast group sizes. But the performance under increasing bit error rates for a given multicast group size is much better compared to *BMMM*. For a group size of 10 stations, the throughput in case of the SRB scheme is approximately $64Kbps$, $64Kbps$ and $63.5Kbps$ for error rates of 0.00001, 0.0001 and 0.00025 respectively, while for the same scenario, the throughputs from the BMMM protocol are $64Kbps$, $59Kbps$ and $51Kbps$ respectively. We see an improvement of about 8.5% for *BER* 0.0001 and 22.5% for *BER* 0.00025 respectively. For a given bit error rate, the throughput is considerably greater in case of *SRB*. For example, for a multicast group of 14 stations and a bit error rate of 0.00025 the throughput from *SRB* is $55kbps$ compared to $40Kbps$ obtained by *BMMM*. The drop in throughput with increasing number of stations is more marked in *BMMM* in contrast to *SRB*. For a *BER* of 0.00025 throughput of *BMMM* drops from about $65Kbps$ to $38Kbps$ as the number of stations increase from 2 to 20. In comparison *SRB* drops from about $65Kbps$ to about $52Kbps$. As the error rates and the associated retransmissions increase *SRB* continues to perform increasingly better than *BMMM* since the control packet overhead is lesser in the *SRB* protocol.

4.7.2 BMMM and SRB throughput vs. multicast group size for varying packet generation rates

Fig.4.4 is a graph of throughputs of *SRB* and *BMMM* under various *CBR* traffic packet generation rates namely 256, 512 and 1024 packets per second, for various multicast group sizes. The *BER* is fixed at 0.00025. Since *802.11b* can support data rates up to *11Mbps* for its data frames, the observed throughput increases with increased packet generation rate for a fixed multicast group size. For example, for a multicast group of 10 stations, the throughput from *SRB* is about $30Kbps$ for a generation rate of 256 packets per second while it increases to $55Kbps$ for 512 packets per second. Also, we notice that the throughput drops with increasing number of stations for the same reasons as described for Fig.4.3.

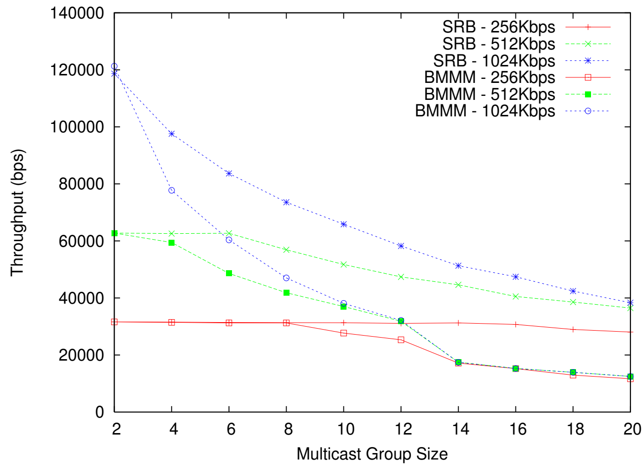

Figure 4.4 BMMM and SRB throughput vs. multicast group size for varying packet generation rates.

The throughput obtained by *SRB* for all packet generation rates is considerably higher compared to *BMMM*. For instance, for a packet generation rate of 1024 packets per second and a multicast group of 14 stations *BMMM* provides a throughput of $20Kbps$ while *SRB* provides a throughput of about $60Kbps$ which amounts to a 200% improvement. We also notice that the gap between curves for *BMMM* and *SRB* for a given packet generation rate grows bigger with increasing multicast group sizes. This again, is due to the difference in control packet overhead between the two protocols. The overhead becomes increasingly striking with increasing multicast group sizes.

4.7.3 BMMM and SRB throughput vs. multicast group size for varying packet sizes

Fig.4.5 shows the performance of the two protocols for varying packet sizes for a fixed *BER* and packet generation rate. The *BER* is fixed at 0.00025 and the packet generation rate is fixed at 512 packets per second. At a low packet size of 256 bytes, the performance of the two schemes is almost the same. This is due to the fact that the number of control bytes transmitted per data byte is so high that most bandwidth is consumed in transmitting control

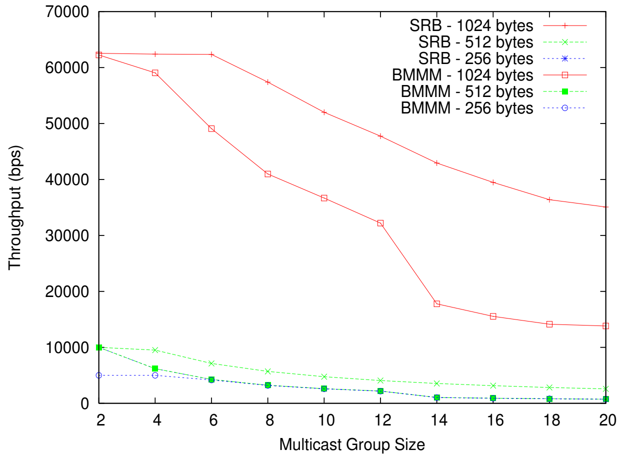

Figure 4.5 BMMM and SRB throughput vs. multicast group size for varying packet sizes.

packets. The operative earnings of *SRB* over *BMMM* in terms of control bytes saved is masked by an extremely high control packet overhead. As the packet size increases, the throughputs of both schemes increase considerably. However, *SRB* grows at a much faster rate compared to *BMMM* since it consumes fewer control bytes. This coupled with the improvement over *BMMM* with increasing multicast group size greatly improves the performance of *SRB* over *BMMM* as evident from Fig.4.5. For instance, for a multicast group of 14 stations the throughput of *SRB* increases from about $5Kbps$ to $42Kbps$ for increase in packet size from 512 bytes to 1024 bytes. The corresponding improvement in *BMMM* is from approximately $1Kbps$ to $20Kbps$.

4.7.4 BMMM and SRB throughput vs. number of cross flows

Fig.4.6 is a comparison of throughputs of *BMMM* and *SRB* in the presence of cross traffic. Cross traffic refers to flows that occur simultaneously with the multicast transmission. We have simulated cross flows by having nodes outside the multicast group communicate with nodes inside the group. The graph plots the total throughput of all flows in the presence of *0* to *4* cross flows. The *BER* is fixed at 0.00001 and the traffic generation rate for each of the flows is 512 packets per second. We see that both with *BMMM* and *SRB* there is an increase in the

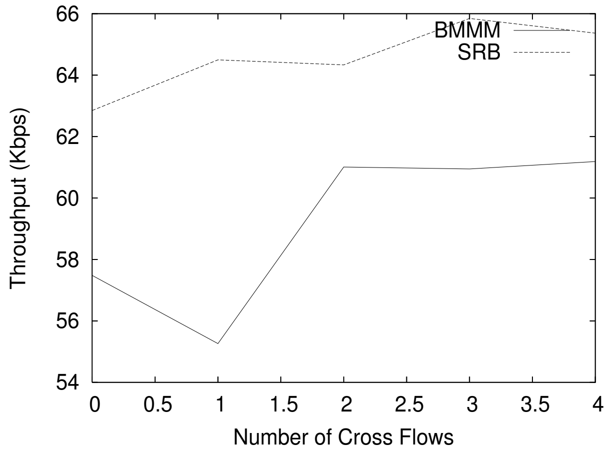

Figure 4.6 BMMM and SRB throughput vs. number of cross flows.

throughput with increasing flows as expected. However, the throughput of *SRB* is consistently better. This is because of the fact that the bandwidth spent in control frame transmission is lesser in case of *SRB*. As a consequence, more bandwidth is dedicated to data frame transfer. From the above illustrations, we have seen that the *SRB* protocol outperforms *BMMM* in presence of increased bit error rates, packet transmission rates and cross traffic, and the improvement is more marked as the size of the multicast group increases. Thus, we believe that the *SRB* protocol is extremely *scalable* since variations of all the factors mentioned above are part of any real world network.

4.8 Functionality with RTS/CTS disabled

It is not uncommon for network operators to completely turn off the *RTS-CTS* mechanism. This is done in order to avoid the control packet exchange overhead incurred. In this case, a sender senses the channel for a period of *DIFS*, and if the channel is idle, it transmits the data and waits for an ACK from the receiver. If the channel is busy, it backs off. Our scheme functions efficiently in such a scenario as well. The *MAID*s in this case, are used to consolidate ACKs alone. As before, stations transmit ACKs in the slots corresponding to their *MAID*s. It is clear that our scheme incurs less overhead than *BMMM* in this case as well due to the absence of the *RAK* frame transmissions.

4.9 Advantages of the scheme

Outlined below are the advantages of using the *Slot Reservation based Reliable Multicast* protocol.

- As in BMMM, the number of contention phases is reduced to 1.
- The number of control frames is further reduced since we use a single RTS to co-ordinate n CTSs and n ACKs.
- The scheme completely eliminates possible collisions among control frames.
- The scheme doesn't require introduction of new control frames unlike other protocols as described in chapter 3.

CHAPTER 5. THE SLOT RESERVATION BASED RELIABLE MULTICAST ALGORITHM WITH FAIRNESS

5.1 Introduction

Firstly, we describe the problem of fairness between co-existing unicast and multicast flows with the multicast flows operating on the native unreliable *IEEE 802.11* protocol. We recollect that the backoff period for a multicast flow is fixed, and is typically CW_{min} while unicast flows use *Binary Exponential Backoff*. Consider a situation where in the medium is currently busy. A competing unicast flow will now sense the medium to be busy and backs off. Once its backoff timer expires the station transmits its frame. If the frame is not delivered successfully, the unicast station doubles its contention window size up to a maximum of CW_{max}. However the backoff period for the multicast station is CW_{min} time slots constantly. Now, when the medium becomes free, the multicasting station will have to wait for a smaller period of time before it can transmit in most cases. However, the unicast station backs off for a longer period of time and by the time its backoff is complete, the medium would have been captured by the multicasting station. The unicast station again doubles its backoff interval since its transmission will be interrupted by the multicast flow and the problem grows worse with each such backoff. In the mostly improbable case where both unicast and multicast flows sense at the same time that the medium is free *i.e.* they both count down their backoff slots to *0* at the same time, the multicast flow sends out its data frame immediately while the unicast flow sends out its *RTS* at the same time. The RTS and the data frame collide and the RTS is lost. The unicast station now goes into a RTS timeout and has to retransmit its RTS after it counts down its doubled contention window. Since multicast is unreliable, the collision is ignored with respect to the multicast frame. Thus, in this case as well, multicast transmissions overwhelm parallel unicast flows.

In chapter 3 we provided an overview of [37] which solves the fairness problem in the case of native *IEEE 802.11* multicast operation. However, for reasons elucidated in section 5.2, we show that such contention window based schemes do not work in the case of reliable multicast. We then propose a novel solution to solve the fairness issue with reliable multicast in section 5.3.

5.2 Unsuitability of Contention Window Based Schemes for Reliable Multicasting

We now describe why Contention Window Based schemes [37] where in the size of the contention window of multicast flows is varied to achieve fairness doesn't work when a reliable multicasting scheme such as [39][31] is used. The time taken for a reliable multicast transmission is considerably larger compared to a unicast transmission. Consider the situation where a multicast transmission gains access to the medium. Suppose that a station with a unicast transmission now tries to gain access to the medium. The station now sees that the medium is busy and backs off. Once the multicast transmission is complete and the medium is idle, both the unicast and multicast station sense the medium to be free. Once the multicast station senses that the medium is free, it goes ahead and sends its RTS. The unicast station however, has to complete backing off for its remaining slots before it can send out its RTS and in the meanwhile, the multicast station occupies the medium again. Now, since the time taken for a reliable multicast is considerably large, in most cases, before the unicast can finish counting down to the 0^{th} slot, an RTS timeout is triggered and the station attempts to retransmit the RTS. In the process, it also doubles the size of its contention window. As a result, the chances of a unicast transmission capturing the medium decreases rapidly with time. After trying to transmit the RTS for certain number of times, the packet is eventually discarded.

In such a situation, clearly, increasing the contention window size of the multicast station has little effect on fairness since very rarely does a multicast station gets to execute the backoff phase. Even if a multicast station gets to execute its backoff, there is a good chance that it

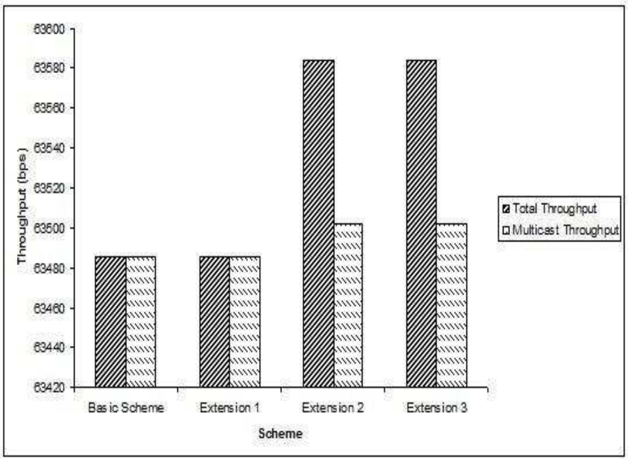

Figure 5.1 Throughput comparison with various contention window based schemes for fairness

will eventually get to transmit its frame, since the duration of a unicast flow is considerably smaller compared to a multicast and there is a very small chance of the RTS being timed out. Decreasing the contention window size of unicast transmission also doesn't work since the main issue affecting fairness is the time required for multicast transmissions and the related timeouts in unicast stations. Although, there is some benefit to be gained from reducing the backoff period for unicast transmissions, the effect is masked by the time spent in waiting for the medium to become idle after a multicast transmission. The wait time, and not the contention window size is the determining factor in affecting fairness. Simulation results in figure 5.1 confirm this observation. The simulation scenario consisted of a single multicast flow with multicast group size of 12 stations and 3 parallel unicast flows. *1024* byte packets were generated at the rate of *512* packets per second.

Basic Scheme in figure 5.1 refers to the scenario where the native *SRB* scheme is adopted for multicast and normal DCF for unicast flows. In *Extension 1*, we modify *SRB* where in the unicast stations do not double their contention window sizes if they are backing off due to an ongoing multicast transmission. In other words, they back off with the same size of the contention window and use the normal exponential backoff if they are backing off due to an ongoing unicast transmission. However, despite this modification we see that the total throughput and the multicast throughput remains the same confirming the observation we made before. The multicast flow quickly grabs the channel before the unicast flows can come out of their backoffs. In *Extension 2*, we set the contention window size for the multicast flow fixed at CW_{max} time slots. In this scenario as well, we see that the total throughput increases very slightly from about 63486*bps* to 63584 bps. In The unicast throughput in this case is about 90*bps* and is still negligibly low. In *Extension 3*, we combine *Extension 1* and *Extension 2*. In this scenario as well, we see very little improvement in the throughput as demonstrated in figure 5.1 and remains the same as in *Extension2*. We see that these schemes are still extremely unfair to unicast flows. The above observations confirm that contention window based schemes do not work well when reliable multicast strategies are used in coalition with DCF unicast transmissions.

In the next section, we provide our *unicast friendly reliable multicast* extension to the *Slot Reservation* based scheme described in chapter 4.

5.3 The Delay Based Method for Fairness in the Slot Reservation Based Reliable Multicast Scheme

Having seen in section 5.2 how and why contention window based schemes which help ensure fairness in case of native *DCF* multicast fail when a reliable multicast scheme is used, we now set out to design a scheme which ensures fairness with such a scheme. We extend the *Slot Reservation* based algorithm from chapter 4 to introduce a fairness component.

Figure 5.1 shows how contention window based schemes add no fairness component when used with reliable multicast schemes. We delved into the details of operation of parallel unicast and multicast flows in section 5.2 to see why such schemes do not work as expected. In doing so, we noted a particularly important criterion. The reason that contention window based schemes do not work are twofold.

- The multicasting station rarely ever has to perform backoff. Hence increasing contention window size of the multicasting station has no effect.

- As a consequence of the multicasting station rarely ever performing backoff, the effect of reducing or limiting the contention window sizes of unicast stations is annulled.

Close inspection of the above points suggests that the main reason that compromises fairness is the fact that a multicast station almost always gets access to the medium every time it has a frame to transmit. In other words, due to various factors described in section 5.2 as long as a multicast station has a frame to send, it beats any other waiting unicast station in staking claim to the medium and capturing it. The *wait time* between successive multicasts is virtually nil and hence unicast stations are never able to capture the medium. If we made the medium *multicast free* for a period of time *i.e.* a period of time where the multicast station is inactive, then unicast stations could contend for the medium during that period, thus giving them an opportunity to transmit their frames. This is the basic idea that we use to introduce fairness.

This idea serves as a basic framework for introducing fairness into parallel unicast and reliable multicast transmissions. We only present the idea as a proof of concept to show that the scheme helps achieve fairness. Schemes can build on this backbone to achieve various degrees of fairness as measured by the fairness index [17][37]. Schemes can also be based on various factors like total operating load [5][11][8][9] and desired bandwidth distribution between unicast and multicast transmissions [10][15][18].

We provide outlines of various possible strategies to introduce wait periods below.

1. Schemes can be based on adjusting the wait time between successive multicast transmissions based on desired *fairness index*[17][37] and bandwidth distribution strategies [15][16]. The frequency of occurrence of these wait periods is kept constant.

2. Based on the above mentioned factors, the number of multicast transmissions after which the wait period is introduced can be varied. The duration of a wait period itself is maintained a constant.

3. (1) and (2) can be combined for fine grained control by varying both the frequency of occurrence and the duration of the wait periods.

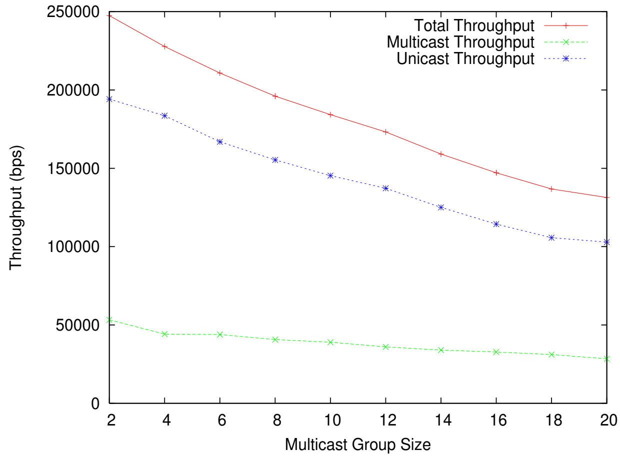

Figure 5.2 Total, Multicast and Unicast Throughputs vs multicast group size for wait period CW_{min}

We demonstrate the efficacy of the idea by using fixed values of wait periods and introducing wait periods between each pair of successive successful multicast transmissions. This might not be the best strategy in terms of achieving optimal throughput and desired fairness levels, but our intention is to demonstrate the effectiveness of the basic approach. Making informed decisions for the above forms our future work as described in chapter 6.

5.4 Simulation Results and Evaluation

We simulated our fairness scheme using our *ns-2* [36] setup. We now present observations from our simulation experiments. We introduced different wait periods between successive multicasts to generate various scenarios for our simulation. We set the packet generation rates for both unicast and multicast transmissions at $512kbps$. The packet size was fixed at 1024 bytes. Our simulation scenario consisted of one multicast transmission and three unicast flows in parallel.

In figure 5.2 we plot the Total, Multicast and Unicast Throughputs for various multicast group sizes for a wait period of CW_{min}. We obtain total throughput ranging from about $250Kbps$ to $160Kbps$ as the mutlicast group size increases from *2* to *20*. The decrease in

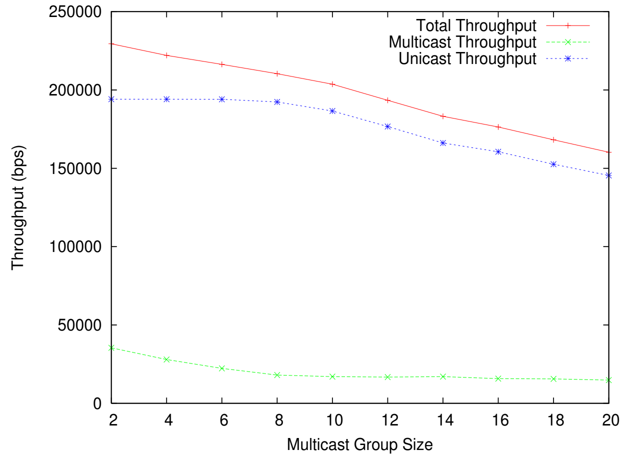

Figure 5.3 Total, Multicast and Unicast Throughputs vs multicast group size for wait period CW_{max}

throughput can be attributed to increased time spent in control frame exchange with increased multicast group size as described in section 4.7. We see that a large share of bandwidth is consumed by unicast flows. The total unicast bandwidth varies from about $190Kbps$ to $100Kbps$ with increase in multicast group size from *2* to *20* while the multicast bandwidth varies from about $50Kbps$ to $30Kbps$.

In figure 5.3 we plot the Total, Multicast and Unicast Throughputs for various multicast group sizes for a wait period of CW_{max}. We obtain total throughput ranging from about $230Kbps$ to $160Kbps$ as the mutlicast group size increases from *2* to *20*. We again see that a large share of bandwidth is consumed by unicast flows. The total unicast bandwidth varies from about $190Kbps$ to $150Kbps$ with increase in multicast group size from *2* to *20* while the multicast bandwidth varies from about $40Kbps$ to $12Kbps$.

In figure 5.4 we plot the Total, Multicast and Unicast Throughputs for various multicast group sizes for a wait period of $CW_{min} + (CW_{max} - CW_{Min}) * 0.5$. We obtain total throughput ranging from about $235Kbps$ to $130Kbps$ as the mutlicast group size increases from *2* to *20*.

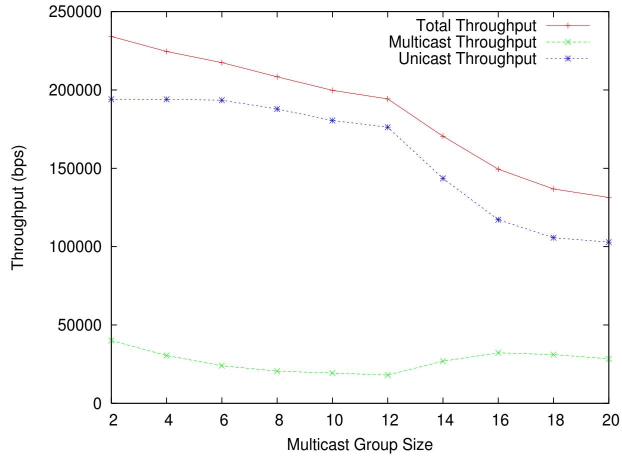

Figure 5.4 Total, Multicast and Unicast Throughputs vs multicast group size for wait period $CW_{min} + (CW_{max} - CW_{Min}) * 0.5$

We again see that a large share of bandwidth is consumed by unicast flows. The total unicast bandwidth varies from about $195Kbps$ to $100Kbps$ with increase in multicast group size from *2* to *20* while the multicast bandwidth varies from about $40Kbps$ to $20Kbps$.

In figure 5.5 we plot total throughput against multicast group size for various wait periods of $CW_{min}, CW_{min} + (CW_{max} - CW_{min}) * 0.25, CW_{min} + (CW_{max} - CW_{min}) * 0.5, CW_{min} + (CW_{max} - CW_{min}) * 0.75$ and CW_{max}. We see that the curve gets steeper in terms of the drop in total throughput with increasing multicast group sizes as the wait period decreases. For example, for a wait period of CW_{min} the throughput drops from about $210Kbps$ to about $170Kbps$ as the multicast group size increases from *6* to *12* while the corresponding drop for a wait period of CW_{max} is from about $220Kbps$ to $195Kbps$. This can be explained as follows. As the multicast group size increases, the time taken for a successful multicast increases. This means, the multicasting station occupies the medium for increasing periods of time as the multicast group size increases. This in turn means that lesser opportunity is available for unicast transmissions. Unicast transmissions get an opportunity in the wait period. Hence, larger the wait period, higher is the opportunity for unicast stations to transmit

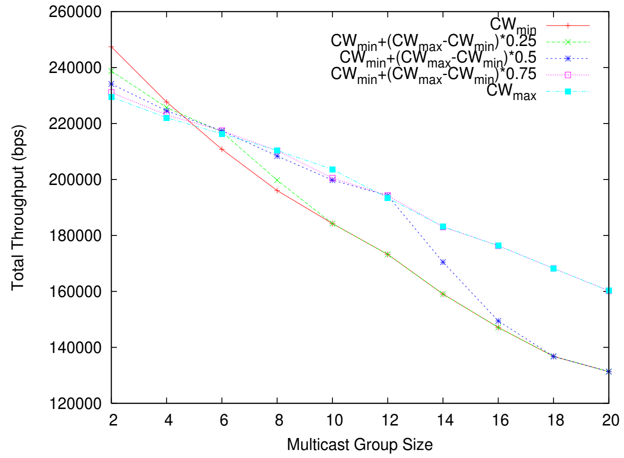

Figure 5.5 Total Throughput vs multicast group size for various wait periods

and hence higher throughput. As seen in figure 5.5 this phenomenon becomes more marked as the multicast group size increases as expected.

In figure 5.6 we plot multicast throughput against multicast group size for various wait periods. We see that the multicast throughput decreases with increase in size of the multicast group for reasons described in section 4.7. We also see that the multicast throughput decreases with increasing wait periods since the wait periods indicate multicast inactivity. For example, for a wait period of CW_{min} the multicast throughput for a multicast group size of *6* is about $45Kbps$ while it decreases to $22Kbps$ for CW_{max}.

In figure 5.7 we plot unicast throughput against multicast group size for various wait periods. We see that the unicast throughput decreases with increase in size of the multicast since the time taken for a successful multicast increases with increase in multicast group size implying lesser opportunity for unicast transmissions. For example for a wait period of CW_{max}, the unicast throughput with a multicast group of size *2* is about $195Kbps$ while it reduces to $145Kbps$ as the size increases to *20*. Also, for a given multicast group size, the unicast throughput increases with increase in wait period as expected. For example, for a multicast

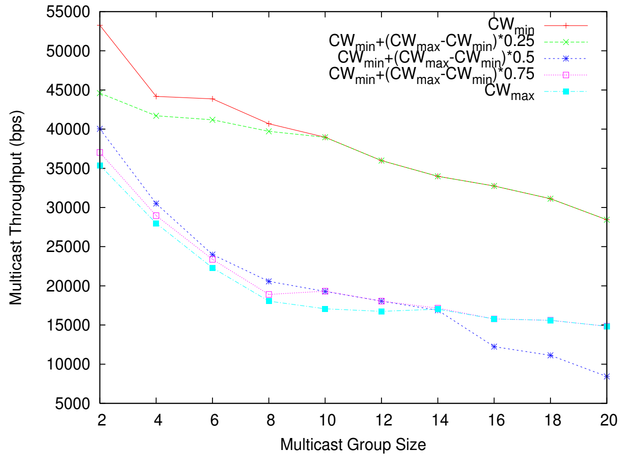

Figure 5.6 Multicast Throughput vs multicast group size for various wait periods

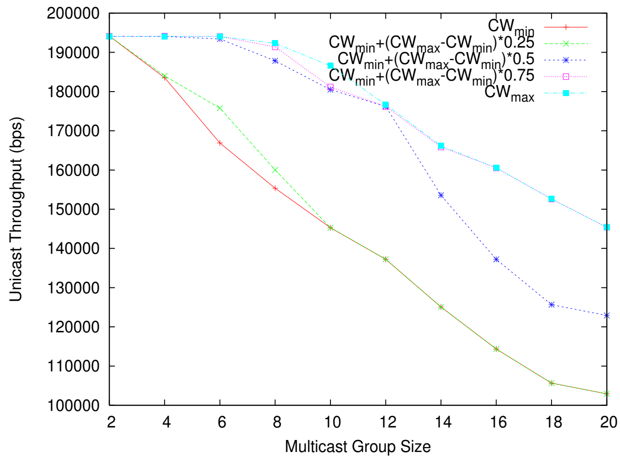

Figure 5.7 Unicast Throughput vs multicast group size for various wait periods

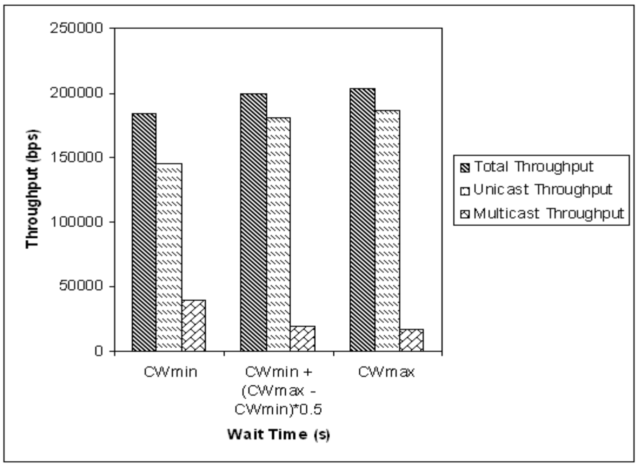

Figure 5.8 Throughput breakup for different wait periods

group of size *12*, for a wait period of $CW_{min}+(CW_{max}-CW_{min})*0.75$, the unicast throughput is $180Kbps$ while for a wait period of $CW_{min}+(CW_{max}-CW_{min})*0.25$, the unicast throughput is $135Kbps$.

In figure 5.8 we plot the break up in throughputs between the total, unicast and multicast throughputs for a fixed mulitcast group size of *12* stations for various wait periods. As expected, unicast throughput increases with increasing wait periods. Also, as a consequence, the time for which the multicasting stations occupies the channel is reduced and hence the multicast throughput decreases. However, the total throughput still increases since the increase in unicast throughput is greater than the decrease in multicast throughput, and the difference grows with increasing wait periods since the *turn around time* for a multicast transmission is much greater than that for a unicast.

In figure 5.9,we plot total throughput against different multicast group sizes for selected wait periods of CW_{min},$CW_{min} + (CW_{max} - CW_{min}) * 0.5$ and CW_{max} and the basic scheme. In doing so, we demonstrate how the SRB scheme with fairness outperforms the basic SRB scheme in terms of total achieved throughput. For example, the total throughput in the basic

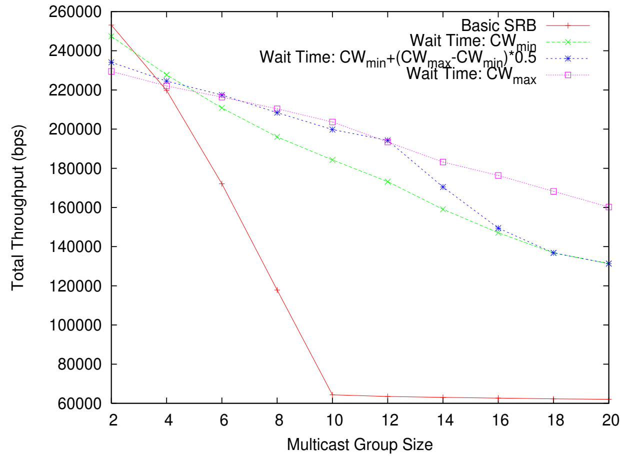

Figure 5.9 Comparison in total throughput between basic SRB and SRB with fairness

scheme falls from $220Kbps$ to about $65Kbps$ as the multicast group size increases from 4 to 10 while the corresponding decrease in throughput is from about $220Kbps$ to $210Kbps$ when the wait period is CW_{max}. As described before, this fall in throughput can be attributed to unfairness caused by multicast flows overwhelming the channel thus reducing the unicast throughput and the total throughput considerably. With the introduction of the wait period, unicast transmissions take place successfully thus increasing the total observed throughput as described before.

CHAPTER 6. CONCLUSIONS AND FUTURE WORK

Multicasting at the MAC layer has the potential of greatly improving current protocols and services and can also form the basis for optimized and efficient schemes in the future. However, multicast operation in the *IEEE 802.11* protocol is inherently inconsistent with its unicast mode of operation in terms of reliability and fairness. Multicast operation in the *IEEE 802.11* does not include RTS-CTS-ACK operation and binary exponential backoff unlike unicast transmissions. When used in coalition with unicast flows, multicast flows are unfair to unicast flows and overwhelm the network preventing unicast transmissions from taking place.

The *IEEE 802.11* MAC does not support reliable multicast and is unfair to unicast transmissions when operating in parallel. As a result, multicast applications with receivers in an 802.11-based LAN cannot deliver data reliably to the multicast receivers unless error recovery is implemented by the upper layers. Ensuring reliability at the MAC layer can greatly reduce the time and bandwidth spent in error recovery compared to handling errors in the upper layers. Therefore, it is desirable to enhance the *802.11 MAC* to support reliable multicast. In this work, we provided a simple, elegant, and efficient protocol to ensure reliability in 802.11 multicast. The protocol uses RTS-CTS-DATA-ACK exchange with a slot reservation based scheduling mechanism to ensure reliable multicast data delivery. We have compared our protocol with an existing reliable multicast protocol, namely *BMMM* through extensive simulations. The results show that our scheme achieves considerably higher multicast throughput compared to *BMMM.*

We then addressed the problem of fairness when unicast and multicast transmission occur in parallel in the *IEEE 802.11* MAC. We established that the relatively large duration of time taken by a multicast transmission compared to its unicast counterpart and the related effect on the backoff and waiting time before it grabs the medium was the cause of the problem. We showed that a multicast transmission waits a much smaller period of time before staking claim to the medium compared to a unicast transmission. By introducing a variable delay between successive multicasts, we showed that the unicast throughput and hence the overall throughput is considerably increased. The achieved throughput (unicast and multicast) can be controlled by appropriately choosing the wait time between multicasts.

As future work, we would like to enhance the fairness scheme and provide a more concrete basis for deciding the wait time between multicasts. We would like to use the concept of *load* [5] [8] [11] to decide the frequency and duration of wait time between multicasts and vary it dynamically depending on existing load and network conditions.

Bibliography

[1] *IEEE 802.11 Wireless LANs, Technical paper*

http : //www.3com.com/other/pdfs/infra/corpinfo/en$_U$S/50307201.pdf

[2] *Introduction to Wireless LAN and IEEE 802.11*

http : //www.tutorial − reports.com/wireless/wlanwifi/introduction$_w$ifi.php

[3] *Daily Wireless*

http : //www.dailywireless.com/features/80211n − wireless − standard − routers − 032807/

[4] William Stallings
Wireless Communications and Networks, Prentice Hall 2005

[5] Aditya Dhananjay and Lu Ruan.
PigWin: Meaningful Load Estimation in IEEE 802.11 Based Wireless LANs. Proceedings of ICC 2008.

[6] Anthony J. Nicholson, Yathin Chawathe, Mike Y. Chen, Brian D. Noble and David Wetherall.
Improved Access Point Selection, In proceedings of MobiSys 2006.

[7] Kompella, R. R., Ramabhadran, S., Ramani, I., and Snoeren, A. C. 2005.
Cooperative packet scheduling via pipelining in 802.11 wireless networks. In Proceeding of

the 2005 ACM SIGCOMM Workshop on Experimental Approaches To Wireless Network Design and Analysis (Philadelphia, Pennsylvania, USA, August 22 - 22, 2005). ACM Press, New York, NY, 35-40

[8] Shiann-Tsong Sheu and Chih-Chiang Wu.
Dynamic Load Balance Algorithm (DLBA) for IEEE 802.11 Wireless LAN. Tamkang Journal of Science and Engineering, vol. 2, No. 1 pp. 45-52 (1999)

[9] I. Papanikos and M. Logothetis. *A Study on Dynamic Load Balance for IEEE 802.11b Wireless LAN.* In Proceedings of COMCON, 2001.

[10] Broustis, I., Papagiannaki, K., Krishnamurthy, S. V., Faloutsos, M., and Mhatre, V. 2007. MDG: *Measurement-Driven Guidelines for 802.11 WLAN Design.* In Proceedings of the 13th Annual ACM international Conference on Mobile Computing and Networking (Montral, Qubec, Canada, September 09 - 14, 2007). MobiCom '07. ACM, New York, NY, 254-265

[11] Gang Wu and Tzi-cker Chiueh.
Passive and Accurate Traffic Load Estimation for Infrastructure-Mode Wireless LAN. MSWIM07, October 2226, 2007, Chania, Crete Island, Greece.

[12] Yigal Bejerano and Randeep S. Bhatia.
MiFi: a framework for fairness and QoS assurance for current IEEE 802.11 networks with multiple access points, Networking, IEEE/ACM Transactions on , vol.14, no.4, pp. 849-862, Aug. 2006

[13] Mathieu Lacage, Mohammad Hossein Manshaei, and Thierry Turletti.
IEEE 802.11 Rate Adaptation: A Practical Approach. MSWiM04, October 4-6, 2004, Venezia, Italy.

[14] Yue Wang.
A Tutorial of 802.11 Implementation in ns-2. MobiTec Lab, CUHK

[15] Balachandran, A., Bahl, P., and Voelker, G. M.
Hot-spot Congestion Relief and Service Guarantees in Public-area Wireless Networks. SIGCOMM Computer Communication Review, Volume 32, Issue 1 (Jan. 2002)

[16] T-C. Tsai and C-F. Lien.
IEEE 802.11 Hot Spot Load Balance and QoS-maintained Seamless Roaming. In Proceedings of National Computer Symposium (NCS), 2003.

[17] Jain, R., Chiu, D.M., and Hawe, W.
A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Systems DEC Research Report TR-301 (1984)

[18] Arnaud Legout, Jorg Nonnenmarcher and Ernst W. Biersack
Bandwidth Allocation Policies for Unicast and Multicast Flows IEEE INFOCOM March 1999.

[19] Starsky Wong, Songwu Lu, Hao Yang and Vaduvur Bharghavan.
Robust rate adaptation for 802.11 wireless networks. In proceedings of MobiCom 2006.

[20] Upkar Varshney
Multicast Over Wireless Networks Communications of the ACM, December 2002/Vol.45, No. 12.

[21] Miller, C. K. 1998
Multicast Networking and Applications. Addison-Wesley Longman Publishing Co., Inc.

[22] D. Waitzman, C. Partridge et. al.
Distance Vector Multicast Routing Protocol http://tools.ietf.org/html/rfc1075, IETF RFC, November 1988

[23] 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.

[24] 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-ietfmanet- aodv-06.tex, IETF Internet Draft, Jul. 2000.

[25] *The IEEE 802.11 Standard*
http://standards.ieee.org/getieee802/down-load/802.11-2007.pdf.

[26] Anurag Kumar, D. Manjunath, Joy Kuri
Wireless Networking, Morgan Kaufmann, 2008.

[27] Kuri, J., Kasera, S.K.
Reliable multicast in multi-access wireless LANs Proc. IEEE INFOCOMM '99, page(s): 760-767 vol.2

[28] K. Tang and M. Gerla
MAC Reliable Broadcast in Ad Hoc Networks, Proc. IEEE MILCOM2001, pp. 1008- 1013, Oct. 2001.

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

[30] 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.

[31] Sun, M., Huang, L., Arora, A., and Lai, T. 2002
Reliable MAC Layer Multicast in IEEE 802.11 Wireless Networks Proc. International Conference on Parallel Processing (Icpp'02) (August 18 - 21, 2002). ICPP. IEEE Computer Society, Washington, DC, 527.

[32] Chakeres, I., Koundinya, C., and Aggarwal, P. 2008
Fast, efficient, and robust multicast in wireless mesh networks, Proc. 5th ACM Symposium

on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks, PE-WASUN '08

[33] S. Gupta, V. Shankar, and S. Lalwani
Reliable Multicast MAC Protocol for Wireless LANs In Proc. IEEE ICC03, May 2003.

[34] M. Lacage, M. H. Manshaei, and T. Turletti.
IEEE 802.11 Rate Adaptation: A Practical Approach ACM MSWiM, 2004

[35] Matthew Gast
802.11 Wireless Networks: The Definitive Guide, O'Reilly, 2005.

[36] www.isi.edu/nsnam/ns/
The Network Simulator - ns-2

[37] Nakjung Choi, Jiho Ryu et. al
Unicast-friendly multicast in IEEE 802.11 wireless LANs Consumer Communications and Networking Conference, 2006. CCNC 2006

[38] Jeongkeun Lee, Jeongkeun Lee et. al.
An Experimental Study on the Capture Effect in 802.11a Networks WiNTECH07, September 10, 2007

[39] Varun Srinivas, Lu Ruan
An Efficient Reliable Multicast Protocol for 802.11-based Wireless LANs IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks, 2009