A Comparative Study of Fairness in Wireless MAC Protocols

Xiao Zhi Lu

A Major Report
in
The Department
of
Computer Science

Presented in Partial Fulfillment of the Requirements
For the Degree of Master of Computer Science at
Concordia University
Montreal, Quebec, Canada

April 2003

0-612-77990-4

ABSTRACT

A Comparative Study of Fairness in Wireless MAC Protocols

Xiao Zhi Lu

Fairness in Medium Access Control protocols is a challenging problem because of the existence of the hidden terminal problem, partially connected network topology and lack of central administration. In this report, we present a comparative study of fairness in two MAC protocols, known as Distributed Fair Scheduling (DFS) and Estimation Based Fair Medium Access (EBFMA). We also compared these two protocols with the IEEE 802.11 Distributed Coordination Function (DCF). Our study shows that in the fully connected network, although EBFMA has the best fairness index, it has the worst throughput. In the fully connected network, the fairness index of IEEE 802.11 and that of DFS do not have significant differences while DFS has better throughput than IEEE. In a partially connected network, IEEE 802.11 is seen to have the best fairness index and throughput because of its binary exponential backoff algorithm while DFS and EBFMA have far worse fairness index and throughput.

To my parents

To my wife, Yan Hui Zhou

To my daughters, Roselyn Weimei Lu & Daisy Ai Lu & Jasmine Yue Lu.

ACKNOWLEDGMENTS

Part of this report was presented in OPNETWORK 2002, Washington, DC, U.S.A, August 2002.

I would like to express my sincere thanks to my advisor Lata Narayanan for her guidance, comments and encouragement. From her, I learned what is seriousness in research.

I would like to thank Mr. Yihui Tang for his advice, as well as Professor J.W. Atwood for his valuable suggestions.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Wireless computing is a rapidly emerging technology providing users with network connectivity without being always connected by cables. In recent years, there has been a tremendous growth in the wireless networking industry. With the increasing usage of mobile and wireless networks in both indoor and outdoor environments, the issue of providing fair channel access among multiple contending hosts over a scarce and shared wireless channel has come to the fore. Fairness is an important issue when accessing a shared wireless channel.

The Medium Access Control (MAC) protocol through which mobile stations can share a common broadcast channel is essential in a wireless network [4]. At the same time, to achieve fairness in the Medium Access Control protocol is a challenging problem [8,20] because of the existence of the hidden terminal problem, partially connected network topology and lack of central administration. In the past few years, several Medium Access Control protocols for wireless networks have been proposed [2,3,8,12,14,20]. In this report, we present a comparative study of fairness in two MAC protocols, known as Distributed Fair Scheduling (DFS) [8,20] and Estimation Based Fair Medium Access (EBFMA) [2]. We also compared these two protocols for fairness with the IEEE 802.11 Distributed Coordination Function (DCF) [10], which is based on the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) [4] mechanism with a rotating backoff window.

We start with reviewing some background on wireless networks. In the following sections, we describe the advantages of wireless networks, the physical layer technologies specified by standards, and the two main models of wireless networks. We conclude this chapter with a brief description of the contribution made by this report.

A. Wireless Network

A wireless network is comprised of devices with wireless adapters communicating with each other using radio waves. A wireless network has following characteristics:

- Lower setting up cost. It avoids the high cost of installing wired lines, and costs associated with frequent moves, additions or changes of devices.
- Flexibility. It does not have constraining wiring infrastructure, so it is easy for workstations to be added or relocated.
- Mobility. Users can access the network at any time, anywhere without being bound to a fixed location.
- Faster deployment. There are no site licenses to obtain, cables to run, or trenches to dig. With a wireless LAN one can be transmitting data in a fraction of the time needed to install a wired connection.
- Compatibility. It is fully compatible with existing wired and wireless networks, one can use it to build a complete wireless infrastructure, add to an existing wireless LAN or add a wireless extension to a wired LAN.

In plain words, the wireless networks have the clear advantage that they do not use wires. This obviously means that portable devices can be part of a LAN without being physically connected. In addition, installation of a wireless network causes less disruption than putting together a wired one. The access point (AP) needs a wired connection to the LAN and power. Most APs can connect up to 32 users to the LAN. By contrast, wiring an additional 32 wall points back to patch panels would undoubtedly be more expensive, more time consuming and cause more disruption than simply adding an AP.

However, wireless networks do have disadvantages. Performance is the main weakness of current wireless technology. The maximum advertised bandwidth is currently 11 Mbps, which just exceeds Ethernet at 10 Mbps. In practice, however, actual bandwidth is 4 Mbps for a single user and peaks at a combined bandwidth of 7 Mbps if several users are using the same AP simultaneously. The severity of these bandwidth limitations will depend on what the wireless LAN is used for and what kind of data is being transferred. If it is employed for Internet surfing, for example, 20 simultaneous users would get 350 Kbps each, which is more than enough for general Internet use. However, if intensive flows applications or video streaming are carried out by more than four simultaneous users, they may experience poor performance, including dropped video frames.

B. Physical Layer Technologies for Wireless Networks

Although wireless devices for interconnecting computers and their peripherals have been available for some years, it is only recently that the adoption of agreed standards by manufacturers has meant that the wireless LAN can be regarded as a generic system level component interchangeable with the wired LAN network interface card (NIC). Two standards in particular have brought this about: the IEEE standard 802.11 and the European Telecommunications Standards Institute (ETSI) spectrum allocation RES.2 [5].

The IEEE 802.11 standard specifies the media access control (MAC) protocol, which forms the lower half of layer 2 of the open systems interconnect (OSI) 7-layer network reference model, and also the physical layer (PHY) specification for layer 1. Within the physical layer specification of IEEE 802.11, there are standards covering the use of infrared optical communications and spread spectrum radio. The spread spectrum radio standard in turn covers both FH and DS variants. Frequency hopping (FH) spreads the spectrum by rapidly switching the carrier frequency. The more sophisticated direct sequence (DS) technique achieves the same effect by multiplying the message data with a pseudorandom bit sequence (PRBS). Both variants have the same overall characteristics, but DS typically will allow a higher over-air data rate than FH.

The following are technologies that manufacturers may choose from when they design a WLAN according to the IEEE 802.11 standard.

- Spread Spectrum Technology. Most wireless LAN systems use the spread spectrum technology. This technology is a wideband radio frequency technique developed by the military during the late 1940s as a mechanism to provide a reliable and secure communication method for the military under battlefield conditions. More bandwidth is consumed with this technology than with narrowband technology, but it produces a signal that in effect is louder and easier to detect. The receiver must know the parameters of the spread spectrum being broadcast. If a receiver is not tuned to the right frequency, a spread spectrum signal looks like background noise. There are two types of spread spectrum radio used in wireless LANs defined under the IEEE 802.11 standard which were mentioned before: frequency hopping and direct sequence.

a. Frequency-Hopping Spread Spectrum Technology. Frequency-hopping spread-spectrum (FHSS) uses a narrowband carrier that changes frequency in a pattern known to both transmitter and receiver. Properly synchronized, the net effect is to maintain a single logical channel. To an unintended receiver, FHSS appears to be short duration impulse noise.

b. Direct-Sequence Spread Spectrum Technology. Direct-sequence spread-spectrum (DSSS) generates a redundant bit pattern for each bit to be transmitted. This bit pattern is called a chipping code. The longer the chip, the greater the probability that the original data can be recovered; however, the more bandwidth required. Even if one or more bits in the chip are damaged during transmission, statistical techniques embedded in the radio can recover the original data without the need for retransmission. To an unintended receiver, DSSS appears as low power wideband noise and is ignored by most narrowband receivers.

- Infrared Technology. Infrared (IR) systems use very high frequencies, just below visible light in the electromagnetic spectrum, to carry data. Like light, IR cannot penetrate through solid objects; it is either directed (line of sight) or diffused (or reflective) technology. Inexpensive directed systems provide very limited range (3 ft) and occasionally are used in specific WLAN applications. High performance directed IR is impractical for mobile users and is therefore used only to implement fixed sub networks. Diffuse IR WLAN systems do not require line of sight, but cells are limited to individual rooms.

C. Infrastructure Wireless Networks

Wireless networks are commonly separated into two broad categories: Ad hoc wireless networks and infrastructure wireless networks.

An infrastructure wireless network refers to wireless stations connected to a wired network via access points or wireless hubs much like workstations being attached to a backbone network via a hub in a wired local area network. In other words, it is an extension of the wireline network with wireless in the last section of the network, and the access point acts as the interface between wireless and wireline networks. Figure 1 gives an example of the infrastructure wireless network model.

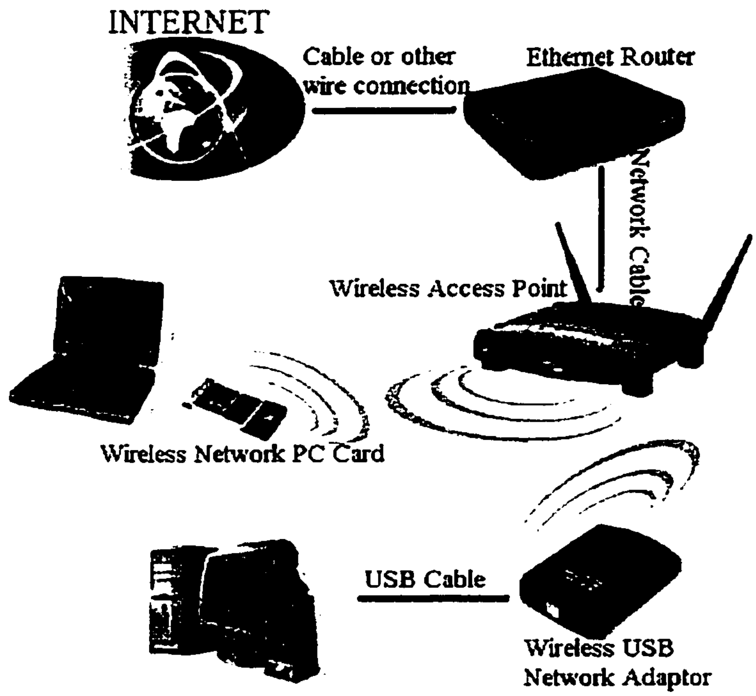

Figure 1. Example of Infrastructure Wireless Network Model

D. Ad Hoc Wireless Network

In areas in which there is little or no communication infrastructure or the existing infrastructure is expensive or inconvenient to use, wireless mobile users may still be able to communicate through the formation of an ad hoc network. An ad hoc network is a collection of wireless mobile hosts forming a temporary network without the aid of any established infrastructure or centralized administration. In such an environment, it may be necessary for one mobile host to enlist the aid of other hosts in forwarding a packet to its destination, due to the limited range of each mobile host's wireless transmissions. In such a network, each mobile node operates not only as a host but also as a router, forwarding packets to other mobile nodes in the network that may not be within direct wireless transmission range of each other. Each node participates in an ad hoc routing protocol that allows it to discover multi-hop paths through the network to any other node. The idea of ad hoc networking is sometimes also called infrastructureless networking [15], since the mobile nodes in the network dynamically establish routing among themselves to form their own network "on the fly". Some examples of the possible uses of ad hoc networking include students using laptop computers to participate in an interactive lecture, business associates sharing information during a meeting, soldiers relaying information for situational awareness on the battlefield, and emergency disaster relief personnel coordinating efforts after a hurricane or earthquake. Figure 2 gives an example of the ad hoc wireless network model.

Ad hoc wireless networks are attracting a lot of interest now mainly because of the following reasons:

- Ad hoc networks are set up on demand and they do not rely on wired base stations.
- Ad hoc networks are fault tolerant, a malfunction in one node can be easily overcome through network reconfiguration.
- Ad hoc networks offer unconstrained connectivity in the area, so that if two nodes are within hearing distance of each other, an instantaneous link between them is automatically formed.

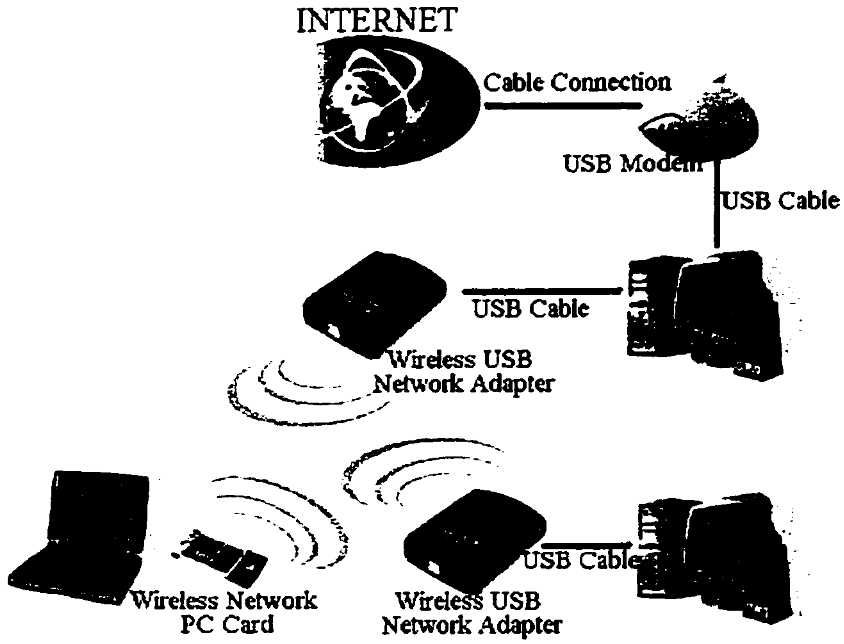

Figure 2. Example of Ad Hoc Wireless Network Model

E. Contribution

This report studies issue of fairness related to the wireless MAC protocols IEEE 802.11 [10], the DFS protocol [8,20] and the EBFMA protocol [2]. We investigate the fairness properties that can be achieved by the above wireless MAC protocols in shared channel wireless networks in general, and in ad hoc networks in particular. We adopt a "shared wireless channel" as a communication regime wherein all nodes communicate over the same logical channel using decentralized control, and there is no concept of a base station in the MAC layer. Shared wireless channels underlie both ad hoc networks and packet cellular networks, and most wireless multiple access protocols [3,8,12,16,18,20], including the basic IEEE 802.11 MAC standard, are designed with these channel assumptions.

First we review the existing literature on these protocols. Then we implement two protocols, DFS and EBFMA, by modifying existing IEEE 802.11 models in the OPNET Modeler 8.0 simulator. We perform simulations of these three protocols in order to understand and study their features. We perform experiments in fully connected as well as partially connected networks, and simulate random as well as fixed flows. In the fully connected network, although EBFMA has the best fairness index [11], it has the worst throughput. The fairness index of IEEE 802.11 and that of DFS do not have significant differences while DFS has better throughput than IEEE. All the simulations show that there is a tradeoff between throughput and fairness index. However, in the partially connected network, IEEE 802.11 has much better fairness index and throughput than both DFS and EBFMA. There is less likelihood of collision in the partially connected network than in the fully connected network. When the medium is idle, IEEE 802.11 will decrease the contention window immediately to the minimum size while DFS and EBFMA do not reduce it so dramatically. This may explain why the IEEE 802.11 protocol's throughput is better that of DFS and EBFMA in the partially connected network. The effect of the topology of the partially connected network on the performance of these algorithms needs to be studied further.

The remainder of this report is organized as follows. Chapter 2 describes the basic background details related to Medium Access Control protocols including the IEEE 802.11 standard, the fairness problem, and several fair queuing algorithms. Chapter 3 describes two MAC protocols, the DFS protocol presented in [20] and the basic EBFMA protocol presented in [2]. Chapter 4 describes the details of the implementing the DFS protocol and the EBFMA protocol, and details of the simulation and data collection. Chapter 5 gives our conclusions.

CHAPTER 2

MEDIUM ACCESS CONTROL & FAIRNESS PROBLEM

In this chapter, first we give a brief background on the medium access control problem and the IEEE 802.11 architecture. In section B, we describe the hidden terminal problem that arises in wireless networks, and the IEEE 802.11 solution to the problem. In the following section, we define the fairness problem, and describe various approaches proposed to resolve it. In the last section, we give a brief description of basic fair queuing algorithms used in wireline networks and the centralized Self-Clocked Fair Queueing algorithm that is inherited by DFS.

A Medium Access Control

The logical architecture of the 802.11 standard that applies to each station consists of a single MAC and one of several possible PHYs: frequency hopping spread spectrum, direct sequence spread spectrum, and infrared light. The goal of the MAC Layer is to provide access control functions, such as addressing, access coordination, frame check sequence generation and checking, and LLC (Logical Link Control) PDU (Protocol Data Unit) delimiting, for shared-medium PHYs in support of the LLC Layer. The MAC Layer performs the addressing and recognition of frames in support of the LLC. The 802.11 standard uses CSMA/CA (carrier sense multiple access with collision avoidance); whereas standard Ethernet uses CSMA/CD (carrier sense multiple access with collision detection).

At the MAC sublayer, IEEE 802.11 supports both contention-free access to the medium, the Point Coordination Function (PCF), which is under the control of a single point coordinator (PC); and contention-based access to the medium, the Distributed Coordination Function (DCF). As can be seen in Figure 3 [1], the PCF ultimately uses the contention-based DCF to provide access to the physical layer. It is the responsibility of the PC to ensure only one of the stations using the PCF transmits at a time.

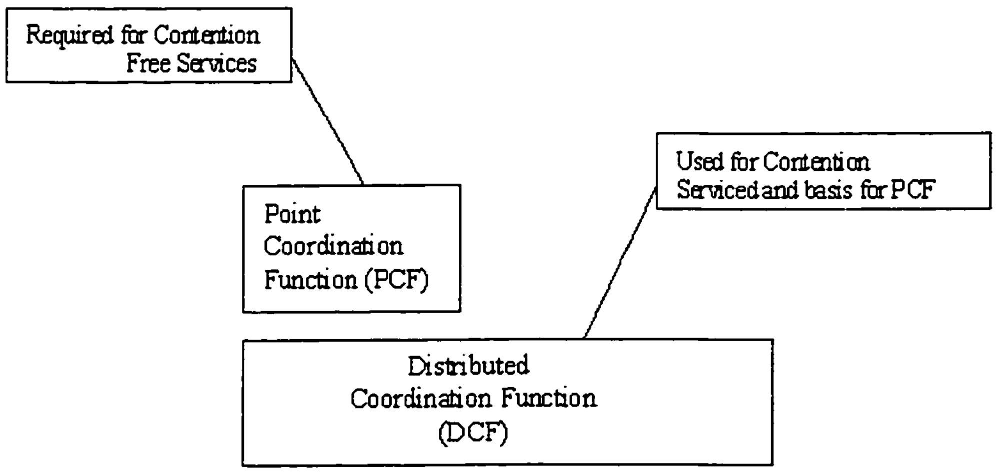

Figure 3. MAC Architecture

IEEE 802.11 also has provisions for a station to operate in a power-save mode, only “waking-up" at specified intervals to determine if there is traffic bound for it. Stations that need to transmit frames to a station that is in power-save mode buffer the frames until the destination station can receive them.

A.1 The Distributed Coordination Function

IEEE 802.11 prioritizes access to the medium by specifying a time interval between frames known as the inter-frame space (IFS). By definition, during an IFS the medium is idle. The different types of IFSs, along with the binary exponential backoff mechanism described below, are the core mechanism a station uses to determine whether it may transmit. This core mechanism is known as the basic access method.

There are four types of IFS: Short IFS (SIFS), PCF IFS (PIFS), DCF IFS (DIFS), and Extended IFS (EIFS). EIFS, which is the longest IFS in terms of time, is used when bit errors are introduced by the physical medium, which cannot be corrected by the radio receiver. Transmission after SIFS, the shortest IFS, is reserved for the PC to send any type of frame required or for other stations to begin transmission of an acknowledgment (ACK) frame, a clear to send (CTS) frame, to respond to polling by the PC, or to send a fragmented MAC protocol data unit (MPDU). Similarly, access after PIFS is reserved for stations to begin transmission of PCF traffic. After DIFS, in general, if a station determines that the medium is idle, it may transmit a pending frame. If the medium is not idle after DIFS, a backoff timer is set by selecting a random integer (i.e., a backoff value (BV)) from a uniform distribution over the interval [0, CW-1], where CW is the width (in slots) of the contention window range. This BV is the number of idle slots the station must wait until it is allowed to transmit. For every idle slot detected (after a DIFS), the timer is decremented by one. If the medium becomes busy prior to the timer expiring, the timer is frozen until the next DIFS, upon which the timer decrements again. Upon expiring, the station transmits its frame. If there is a collision, CW is doubled until it reaches a predefined maximum value, CW_max. Upon a successful transmission, CW is reset to the default minimum value of CW_min. The backoff mechanism is called binary exponential backoff. Figure 4 [1] shows the structure of the basic access method.

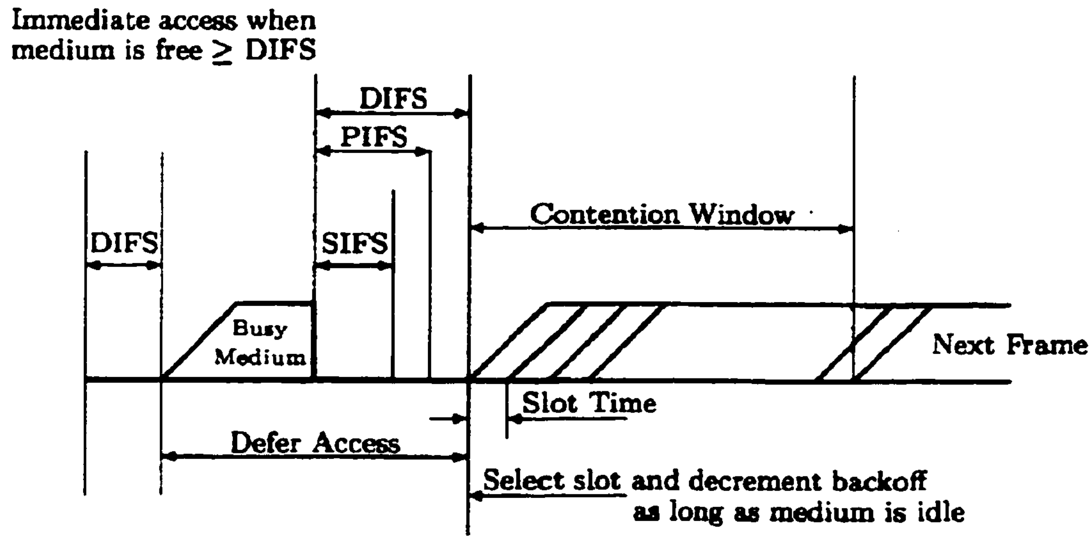

Figure 4. IEEE 802.11 Basic Access Method

A.2 Point Coordination Function

The PCF within the PC controls transfers during a Contention Free Period (CFP). Within IEEE 802.11, CFPs alternate with Contention Periods (CPs) (when the DCF controls transfers) as shown in Figure 5 [1]. The PC determines the rate at which CFPs are generated. At the beginning of a CFP, the PC transmits a beacon frame. That beacon signals the beginning of the CFP and includes a timestamp, beacon interval, and maximum duration information (CFPMaxDuration) for this CFP. All stations set their Network Allocation Vector (NAV) with the CFPMaxDuration. During the duration specified by CFPMaxDuration, stations may only transmit in response to a poll by the PC, or transmit ACKs in response to frames sent to them.

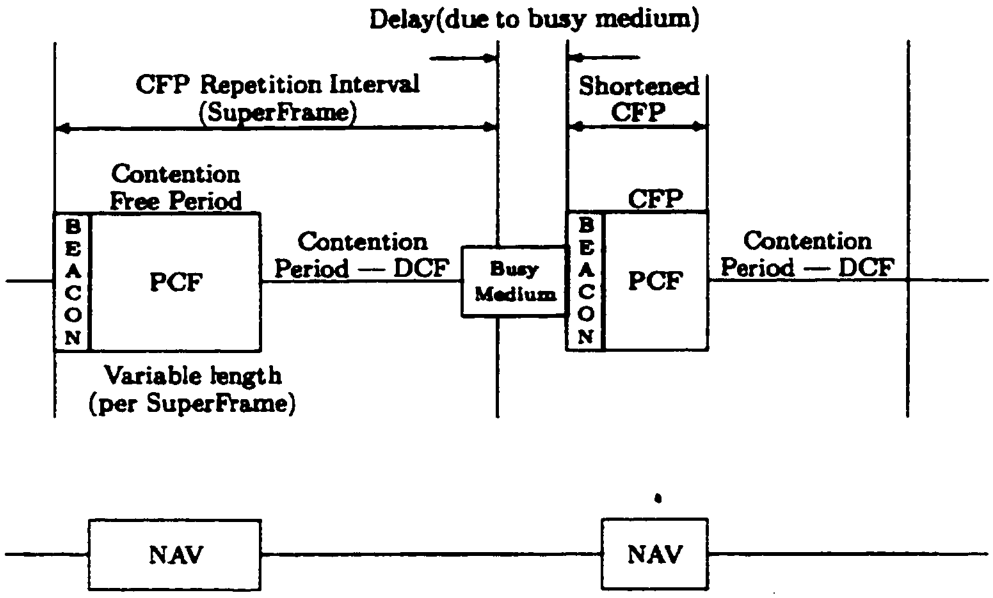

Figure 5. CFP/CP Alternation

This continues for CFPMaxDuration or until the PC explicitly declares the CFP terminated, whichever occurs first. As can be seen in Figure 5, the beacon interval is a nominal value, that is, it may be delayed due to a busy medium. In those cases, the CFP is shortened by the amount of the delay. During the CFP, the PC may send unicast or multicast frames and/or poll stations that have indicated that they would like the opportunity to transmit during the CFP.

B. Hidden Terminal Problem

In ad hoc radio networks, due to the limited transmission range of mobile stations, packets arriving from transmitters who may not know of each other may collide at a given receiver, rendering the data unintelligible. Consider a simple channel model where a transmitter has a fixed transmission range, and multiple transmissions in the neighborhood of a receiver will cause a collision at the receiver. A successful transmission precludes any station in the neighborhood of either the transmitter or the receiver from engaging in another simultaneous packet transmission/reception. In other words, transmission of packets involves contention over the joint neighborhoods of the sender and receiver, and the level of contention for the shared wireless channel in a geographical region is spatially dependent on the number of contending nodes in the region. This is fundamentally different from the wireline situation, wherein all flows perceive the same contention. Consider the example shown in Figure 6. A hidden station is one that is within the range of the intended destination but out of range of the sender. Station A is transmitting to station B. Station C cannot hear the transmission from A. During this transmission when C senses the channel, it falsely thinks that the channel is idle. If station C starts a transmission, it interferes with the data reception at B. In this case station C is a hidden station to A. Hence, hidden stations can cause collisions on data transmission. This is the so-called "hidden terminal" problem [3], which is known to degrade throughput significantly.

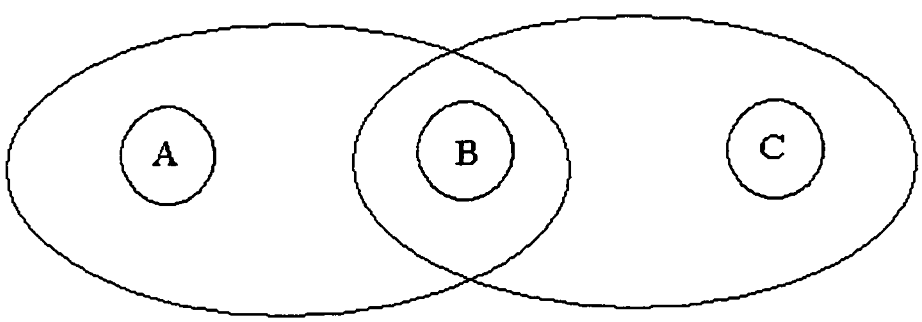

Figure 6. Hidden Terminal Problem

Several medium access control protocols have been devised to address this "hidden terminal" problem [2,3,4,10]. Among these protocols, IEEE 802.11 Distributed Foundation Wireless Medium Access Control (DFWMAC) is a proposed standard for wireless ad hoc and infrastructure LANs. DFWMAC is based on CSMA/CA and provides also the RTS/CTS access method. The RTS/CTS access method is used to deal with the hidden terminal problem by allowing stations to acquire the channel before they transmit the data packets. In other words collisions can occur only during transmission of short control packets, RTS and CTS, rather than during the transmission of potentially very long data packets. Thus, through the proper use of RTS/CTS, the performance of a wireless network would be improved depending on the operating environment. Ideally, the RTS/CTS access method/handshake can eliminate most interference.

If the RTS/CTS is chosen to be implemented on a particular station, it will refrain from sending a data frame until the station completes an RTS/CTS handshake with another station, such as an access point. A station initiates the process by sending an RTS frame; the access point receives the RTS and responds with a CTS frame. The station must receive a CTS frame before sending the data frame. The CTS also contains a timeout value/backoff value that keep other stations away from accessing the medium while the station initiating the RTS transmits its data.

In the case illustrated by Figure 6, if either Station A or Station C activates RTS/CTS, the collision will not happen. Before transmitting, Station A would send an RTS and receive a CTS from the station B. The timing value in the CTS, which Station C also receives, will cause Station C to hold off long enough for Station A to transmit the frame. Thus, the use of RTS/CTS reduces collisions and increases the performance of the network if hidden stations are present.

The increase in performance using RTS/CTS is the net result of introducing overhead (i.e., RTS/CTS frames) and reducing overhead (i.e., fewer retransmissions of large packets). In an environment where there is no hidden node, the use of RTS/CTS will only increase the amount of overhead while reducing throughput. A slight hidden node problem may also result in performance degradation if RTS/CTS is enabled. In this simple case, the additional RTS/CTS frames cost more in terms of overhead than the increased performance by reducing retransmissions.

The effects of the RTS/CTS mechanism are:

- Increases bandwidth efficiency by reduced collision probability since the ongoing transmission has been made known everywhere within the relevant range.
- Increases bandwidth efficiency since if collisions occur they do not occur with the long data packets but with the relative small control packets.
- Decreases bandwidth efficiency since it transmits two additional packets without any payload.
- Decreases bandwidth efficiency since it reserves geographical space for its transmission where or when it might not actually be needed.

Due to the above listed trade offs of the RTS/CTS mechanism, the standard allows its usage but does not demand it. So, RTS/CTS is an optional feature of the standard. Usage policy is set on a per station basis with the help of a manageable object RTS threshold that indicates the payload length under which the data frames should be sent without the RTS/CTS prefix. This parameter is not fixed in the standard and has to be set separately by each station. One can enable RTS/CTS by setting a specific packet size threshold (0 — 2347 bytes) in the user configuration interface. If the packet that the access point is transmitting is larger than the threshold, it will initiate the RTS/CTS function. If the packet size is equal to or less than the threshold, the access point will not kick off RTS/CTS. Most vendors recommend using a threshold of around 500. The use of 2347 bytes effectively disables RTS/CTS for the access point.

One of the best ways to determine if RTS/CTS should be activated is to monitor the wireless LAN for collisions. If a large number of collisions occurred and the users are relatively far apart and likely out of range, then RTS/CTS can be enabled on the applicable user wireless NICs. For the access point, after receiving an RTS frame from a user's radio NIC, the access point will always respond with a CTS frame.

Although the RTS/CTS access method can alleviate the effects induced by the presence of hidden terminals, DFWMAC still suffers from a fairness problem that is also induced mainly by the intrinsic multi hop nature of ad hoc networks.

C. Fairness Problem

While most current non-military ad hoc network test-beds are experimental in nature, possible future deployment scenarios include deeply networked conglomerations of embedded devices, emergency rescue operations, "zero conf" meeting setups, and rapidly reconfigurable metropolitan wireless networks. Migrating from experimental environments to commercial environments, ad hoc network designers will need to address critical new challenges, such as "service differentiation" among contending users for the dynamic and scarce channel resources [14]. In a pay-for-use model, the network must guarantee that minimum performance requirements of paying users will be met, at least in relative terms. Since link layer fairness mechanisms serve as the basis for achieving network layer quality of service (QoS) [14], wireless MAC protocols in commercial ad hoc networks must support some notion of "weighted fairness", wherein flows with larger weights receive correspondingly better service in accordance with a system wide fairness model.

The fairness problem was first pointed out by Bharghavan *et al.* [3]. This problem occurs mainly because of the hidden terminal problem as well as the backoff scheme used in the DFWMAC protocol. This phenomenon can be simply illustrated by the configuration in Figure 7.

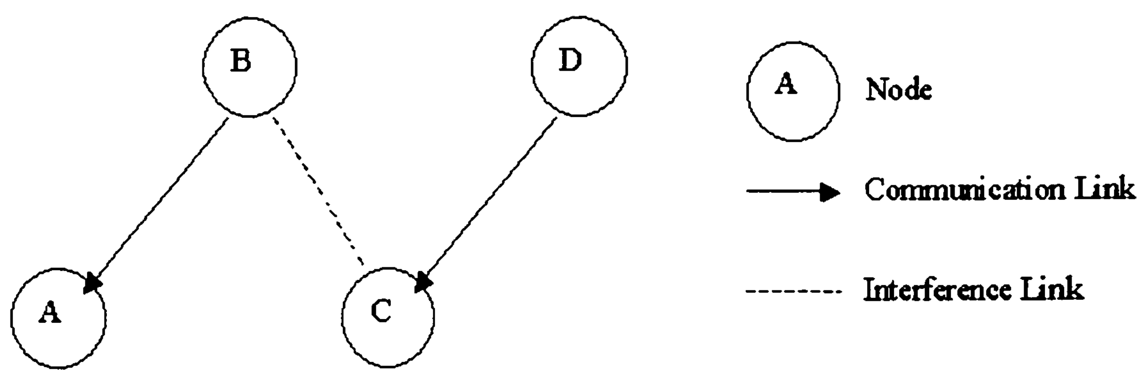

Figure 7. Fairness Problem

In Figure 7, two flows/links compete for the radio resources: flow 1, established between nodes A and B and flow 2 established between nodes C and D. Due to the nature of the RTS/CTS based protocol, in this configuration, node C overhearing the RTS of node B will not reply to RTSs of node D. Nodes D and B are hidden from each other and thus node B would not overhear node D's RTSs to node C. When the traffic in both flows is heavy, due to the binary exponential backoff used in DFWMAC, node D's contention window would double each time there is collision at C (inferred by D from the absence of CTS from C) until it reaches the maximum value (CW_Max) specified by the protocol. Meanwhile, node B's window would decrease since B would (eventually) receive ACKs for the data packets sent to node A until it reaches the minimum value (CW_Min) specified by the protocol. In effect thus node B would have an average contention window size that is much smaller than that of node D and thus would have statistically much higher chance of accessing the channel than node D, which is unfair since the two flows should share the link equally. Even if we define the notion of fairness by addressing the two issues above, designing mechanisms for achieving such fairness is a major challenge since there is no centralized control and no station is guaranteed to have accurate knowledge of the contention even in its own neighborhood [14]. Further, contention is really "per-flow" (i.e., a sender-receiver pair) rather than per-node, which makes its estimation harder at the transmitter before it decides to contend for the channel. Finally, contention resolution must be achieved without assuming any explicit coordination or handshakes among the contenders in order to preserve the robustness of multiple access protocols.

Usually, the problem of fairness in wireless ad hoc networks is addressed using a classic approach inherited from wired networks [9]. The common assumption is that nodes/flows have pre-assigned fair shares. The task becomes then to modify wired networks' fair queuing algorithms to address the nature of wireless networks. Wired networks have efficient means of allocating fair shares through admission control and additionally, the fair shares remain constant throughout the session duration due to the static nature of the nodes. In ad hoc networks, it is meaningless to assume statically pre-assigned fair shares, since on one hand, not only do the nodes move, but also the routers are mobile. On the other hand, contention is location dependent, such that in terms of absolute guarantees, fairness would at most mean "avoiding starvation". Thus applying rate proportional fair queuing algorithms is beyond the original goal of such an algorithm, namely, flow isolation/protection and bandwidth guarantees. In a shared channel wireless network, spatial reuse of the channel bandwidth may be obtained by simultaneously scheduling transmissions whose regions of contention are not in conflict. While spatial reuse is very useful for increasing the utilization of the wireless channel, it introduces a fundamental conflict between optimizing aggregate allocated bandwidth and achieving fairness, because allocating the channel to a flow with a large contention correspondingly reduces the channel reuse. Thus, there is a tradeoff between channel utilization and fairness [4,14,20]. In contrast, wireline and cellular networks do not face this problem because all flows perceive the same contention.

Luo *et al.* [13] have proposed a two-phase scheduling scheme to achieve fairness in ad hoc networks. The algorithm constructs a tree comprising all mobile stations, and then rearranges the tree so that it becomes conflict-free among the one-hop flows in each level of the tree. During the process of constructing the conflict-free tree, the stations in each level of the tree will propagate the tree knowledge so that the stations can perform a weighted fair queuing (WFQ) scheduling among different levels in the tree. The time required for constructing the tree can however be very long when there are many nodes involved in the network. When mobility is taken into account, say if the root station of the conflict-free tree moves out of its original location, the tree has to be reconstructed to maintain the global fairness among one-hop flows.

Another approach devised by Vaidya and Bahl [19], also called DFS, to address this problem inherits the virtual clock method used in wired networks to provide fair queuing [17]. The mobile station broadcasts its virtual clock to its neighboring stations, using piggy backing or in a broadcast channel, and updates its own virtual clock from other stations' broadcasts. Then the mobile station scales its contention window according to the updated virtual clock and its flow's fair share. This approach is good in wireless networks in which there are no hidden terminals. However in multihop wireless environments, there is a problem among different regions in the network as contention is not homogeneous. Two stations that can interfere with each other while being hidden from each other, may face very different competition (i.e., may hear each a different number of stations). Consequently, the stations will negotiate different virtual clocks from their neighborhood; the one that faces heavier competition will have a slower virtual clock. Thus two stations that can interfere with each other may use different virtual clocks obtained from their different regions, thus, they cannot schedule fairly.

In [2] a measurement-based algorithm called EBFMA is proposed to achieve fairness in ad hoc networks. The algorithm replaces the binary exponential backoff (BEB) algorithm by another backoff scheme where the contention windows are adjusted according to the stations' fair shares. Each station in the ad hoc network estimates the amount of traffic it generates against its fair share and the amount of traffic generated by other stations it can overhear against other stations' fair share, and based on this, the station adjusts its contention window size in order to equalize both ratios. This makes the probability of attempting to access the channel proportional to the station's own traffic weight. This algorithm is simple and incurs no additional computational overhead. However, the algorithm has a major drawback when operating in a dense network where all stations can hear each other's transmissions. That is, the contention window adjustment in [2,14] replaces the exponential increment of the BEB without paying attention to the traffic and stations density. While the exponential increment mechanism of the BEB of IEEE 802.11 is used to alleviate the frequent collisions when the density of the traffic and stations increases, the stations in the proposed algorithm become very aggressive under the same traffic conditions. Each station overhearing other's successful transmission will reduce its contention window towards CW_Min, and on average all contention windows will be much closer to CW_Min, leading thus to more collisions. The BEB in the same environment would lead to an average window increasingly closer to CW_Max to alleviate the collisions. In short, while the algorithm addresses very well the fairness problem in ad hoc networks by overcoming the bad properties of the BEB algorithm, it fails to conserve the good properties of the latter.

D. Fair Queuing

In wireline networks, fair queuing has long been a popular paradigm [7,17] for providing bounded delay access to a shared unidirectional channel, and hence for providing guaranteed quality of service. All fair queuing algorithms are based on the notion of approximating the stream model, in which packet flows are modeled as stream that traverse a shared pipe. Consider a set of flows $i \in F$ that share a channel. Let flow i have a weight r_i, where r_i is the number of bits of flow i served in a single "round" by the stream fair queuing server. Stream fair queuing guarantees that for an arbitrary time window $[t_1, t_2]$ during which any two flows i and j are backlogged (i.e., they have bits to transmit), $W_i(t_1, t_2)/r_i = W_j(t_1, t_2)/r_j$, where $W_i(t_1, t_2)$ is the service (in bits) received by flow i in the time window $[t1, t2]$. Essentially, stream fair queuing divides the channel capacity at any instant among backlogged flows in the proportion of their weights. As a direct consequence of this model, flows that do not have any bits to transmit at some time cannot be compensated at a later time. Since networks switch flows at the granularity of packets rather than bits, and since the switching is non preemptive (i.e., all bits in a packet are transmitted back-to-back), packet fair queuing algorithms must approximate the stream model. Thus, the goal of a packet fair queuing algorithm is to minimize

$$| W_i(t_1, t_2)/r_i - Wj(t_1, t_2)/r_j |$$

for any two backlogged flows i and j over an arbitrary time window $[t1, t2]$. This is achieved by assigning a "virtual time" start tag and finish tag to each packet, and serving the packet with the minimum finish tag, where the virtual time of the channel corresponds to the current round being served in the corresponding stream fair queuing model. Thus, the kth packet of flow i, p_i^k, that arrives at time $A(p_i^k)$, is assigned a start tag $S(p_i^k) = \max\{V(A(p_i^k)), F(p_i^{k-1})\}$, and a finish tag $F(p_i^k) = S(p_i^k) + L(p_i^k)/r_i$, where $L(p_i^k)$ is the size (in bits) of packet p_i^k and r_i is the weight of flow i. $V(t)$, the virtual time corresponding to time t, maintains the "round number" of the stream fair queuing model at time t. Let $\sum_{i\in B(t)} r_i$ be the bits transmitted in each round, where $B(t)$ is the set of flows that are backlogged at time t. Then, $dV/dt = C/\sum_{i\in B(t)} r_i$, where C is the channel capacity.

The DFS algorithm [20] was designed in an attempt to emulate Self-Clocked Fair Queuing (SCFQ) [6] in a distributed manner. The following paragraphs briefly describe the centralized SCFQ algorithm. A virtual clock is maintained by the central coordinator, and $v(t)$ denotes the virtual time at real time t. Let p_i^k denote the kth packet arriving on flow i. Let A_i^k denote the real time at which packet p_i^k arrives. Let L_i^k denote the size of packet p_i^k. A start tag S_i^k and a finish tag F_i^k are associated with each packet p_i^k, as described below. Let $F_i^0 = 0, \ \forall i$.

- On arrival of packet p_i^k, the packet is stamped with start tag S_i^k, calculated as

  $$S_i^k = \max\{v\,(A_i^k), F_i^{k-1}\}$$

- Also, F_i^k, the finish tag of p_i^k is calculated as

  $$F_i^k = S_i^k + \frac{L_i^k}{\phi_i}$$

- Initially, the virtual clock is set to 0, i.e., $v(0) = 0$. The virtual time is updated only when a new packet is transmitted. When a packet begins transmission on the output link, the virtual clock is set equal to the finish tag of that packet.

- Packets are transmitted on the link in the increasing order of their finish tags. Ties are broken arbitrarily.

As noted in Step 1 above, in the SCFQ algorithm (and, also in other algorithms, such as SFQ [7] etc.), the start and finish tags are calculated when a packet arrives in a flow. An alternative approach is to calculate the start tag when a packet reaches the front of its flow—that is, for a packet p_i^k in flow i, start and finish tags are calculated only after all packets that arrived in flow i before packet p_i^k have been serviced. If this approach were to be used, then calculation of the start tag above should be modified as follows:

- Let f_i^k denote the real time when packet p_i^k reaches the front of its flow. If p_i^k arrives on an empty flow, then $f_i^k = A_i^k$; else f_i^k will denote the real time when P_i^{k-1} finishes service. On arrival of packet p_i^k at the front of its flow, the packet is stamped with start tag S_i^k, calculated as $S_i^k = v(\, f_i^k\,)$.

The finish tag is calculated as before, as $F_i^k = S_i^k + \frac{L_i^k}{\varphi_i}$. It is a simple exercise to verify that, for the SCFQ algorithm, this new procedure and the earlier procedure result in the same start and finish tags for all packets. In our distributed implementation, however, we emulate the latter procedure.

CHAPTER 3

TWO MAC PROTOCOLS – DFS & EBFMA

In this chapter, we give an overview of the two MAC protocols — DFS and EBFMA — that we investigate in our simulation.

A. DFS Overview

The Distributed Fair Scheduling (DFS) [20] protocol is a fully distributed algorithm for fair scheduling in a wireless LAN. The DFS protocol is based on the IEEE 802.11 MAC and SCFQ [6]. It can allocate bandwidth in proportion to the weights of the flows sharing the channel. The authors of the DFS protocol claim the DFS protocol may also be applied to wired LANs, as well as be extended to multi-hop wireless networks. The essential idea is to choose a backoff interval that is proportional to the finish tag of the packet to be transmitted. The idea of the finish tag is borrowed from SCFQ [6]. The DFS protocol borrows on SCFQ's idea of transmitting the packet whose finish tag is smallest, as well as SCFQ's mechanism for updating the virtual time. The smallest finish tag is determined by a distributed approach using the backoff interval mechanism from the IEEE 802.11 standard.

The following is a brief introduction to DFS. Each node i maintains a local virtual clock, $v_i(t)$, where $v_i(0) = 0$. Now, p_i^k represents the kth packet arriving at the flow at node i on the LAN.

- Each transmitted packet is tagged with its finish tag.

- When at time t node i hears or transmits a packet with finish tag Z, node i sets its virtual clock v_i equal to *maximum*$(v_i(t),Z)$.
- Start and finish tags for a packet are not calculated when the packet arrives. Instead, the tags for a packet are calculated when the packet reaches the front of its flow. When packet p_i^k reaches the front of its flow at node i, the packet is stamped with start tag $S_i^k = v(f_i^k)$, where f_i^k denotes the real time when packet p_i^k reaches the front of the flow.
- The finish tag F_i^k for packet p_i^k is calculated as follows, where an appropriate choice of the *Scaling_Factor* allows us to choose a suitable scale for the virtual time.

$$F_i^k = S_i^k + Scaling\_Factor * \frac{L_i^k}{\phi_i} = v(f_i^k) + Scaling\_Factor * \frac{L_i^k}{\phi_i}$$

- The objective of the next step is to choose a backoff interval such that a packet with smaller finish tag will ideally be assigned a smaller backoff interval. This step is performed at time f_i^k. Specifically, node i picks a backoff interval B_i for packet p_i^k, as a function of F_i^k and the current virtual time $v_i(f_i^k)$, as follows: $B_i = \lfloor F_i^k - v(f_i^k) \rfloor$ slots.

Now, we can get:

$$B_i = \left| Scaling\_factor * \frac{L_i^k}{\phi_i} \right| \qquad (1)$$

Scaling_factor allows for the choice of a suitable scale for the backoff intervals. To reduce the possibility of collisions, the authors propose a randomization of B_i as follows,

$$B_i = \rho * B_i \tag{2}$$

where ρ is a random variable with mean 1, and is uniformly distributed in the defined interval. When this step is performed, a variable named *CollisionCounter* is reset to 0.

Unlike IEEE 802.11, DFS separates the backoff intervals used initially from those used after collision. When a collision occurs for node *i*, it chooses a new backoff interval as follows.

- Increment *CollisionCounter* by 1.
- Choose new B_i uniformly distributed in $\left[1, 2^{CollisionCounter-1} * CollisionWindow\right]$

where *CollisionWindow* is a constant parameter.

The authors of DFS refer to the scheme presented above for calculating the backoff interval as the Linear Mapping (or the Linear Scheme). The authors also present two other mappings known as the Exponential Mapping Scheme and the Square Root Mapping Scheme that can improve the throughput in the wireless networks compared to the Linear Mapping. We only consider the Linear Mapping Scheme here.

The DFS protocol allocates throughput in proportion to the weights of the flows. It accounts for variable packet sizes and variable weights.

B. EBFMA Overview

In the Estimation Based Fair Medium Access (EBFMA) [2] protocol, each station adjusts its contention window according to the estimated share it obtained for itself and other stations. For fair queuing, it borrows the idea from wireline networks [17] and defines the "fairness index" for ad hoc networks to quantify the fairness, so that the goal of achieving fairness becomes equivalent to minimizing the fairness index. Then it chooses a backoff scheme that is different from the IEEE 802.11 standard.

The following is a brief introduction to EBFMA. The authors of the EBFMA protocol introduce the following notations:

ϕ_i: A pre-defined fair share that station i should receive. Normally, it should be determined at admission control, i.e., when the node joins the ad hoc network, and can be readjusted for example when a node becomes a router.

W_i: The actual throughput achieved by station i.

L_i: Station i's offered load.

According to the authors of the EBFMA protocol, a fair MAC protocol should have the following properties. When a station's offered load to the channel is much lower than the channel capacity, each station's request for transmission should be met. This means that for any station i, $W_i = L_i$. When a station's offered load exceeds the channel capacity, each station i should be able to get its fair share of the channel, i.e., proportional to ϕ_i.

This means that for any pair of stations i and j, $\frac{Wi}{\phi_i} = \frac{Wj}{\phi_j}$. This is just for ideal situations.

In reality, we want to bound the value of $\left| \frac{Wi}{\phi_i} - \frac{Wj}{\phi_j} \right|$ by the smallest possible value.

Instead of working with absolute values, they defined the fairness index, FI, to be:

$$FI = \max\left\{\forall i, j : \max\left(\frac{Wi}{\phi_i}, \frac{Wj}{\phi_j}\right) / \min\left(\frac{Wi}{\phi_i}, \frac{Wj}{\phi_j}\right)\right\} \tag{3}$$

Therefore, the authors' goal becomes the design of a distributed MAC protocol that can minimize *FI* and thus achieve fairness for all the stations in an ad hoc network.

The following approach can be used to choose ϕ for any station i and in situations where the ad hoc network is open to everyone without admission control, which can happen in situations where all the stations are trusted and known not to misbehave. If each station is considered to be a greedy source and wants to get the same share as all other stations as a whole, then it can just set $\phi = 0.5$ regardless of the number of its neighbors. As for any station, say i, it requests the same share as all the others in its vicinity. These stations have a total share of $\phi_0 = 1 - \phi_i = 0.5$, which equals this station's share ϕ_i. This can be interpreted as a per-station fairness. If a station has two active links (or streams in the terminology of MACAW [3]), which can happen when a station acts as a router in an ad hoc network, it can set ϕ to satisfy:

$$\frac{\phi_i}{\phi_0} = \frac{\phi_i}{1-\phi_i} = \frac{2}{1} \Rightarrow \phi_i \approx 0.67$$

This shows simply that the station (router) wants to obtain two times as much share of bandwidth as other stations to function as a router properly. This can be interpreted as MACAW's per-stream fairness [3].

The authors propose two algorithms.

Algorithm 1. Fair Share Estimation

switch (received packet type) {

case RTS:

if (destID != localID) W_{eo} += T_{rts}

else {send CTS packet;

W_{eo} += (T_{rts} + T_{cts})}

case CTS:

if (destID != localID) W_{eo} += (T_{rts} + T_{cts})

else {send DATA packet;

W_{ei} += (T_{rts} + T_{cts} + T_{data})}

case DATA:

if (destID != localID)

W_{eo} += (T_{rts} + T_{cts} + T_{data})

else {send ACK packet;

W_{ei} += (T_{rts} + T_{cts} + T_{data} + T_{ack})}

case ACK:

if (destID != localID)

W_{eo} += (T_{rts} + T_{cts} + T_{data} + T_{ack})

else { W_{ei} += (T_{rts} + T_{cts} + T_{data} + T_{ack})}

}

Whenever sending an RTS packet, W_{ei} += T_{rts}.

Here W_{ei} is the estimated share of the estimating station itself. W_{eo} is the estimated share of other stations. T_{type} is the time to transmit a packet of type *type*. For example, T_{rts} is the time to transmit a "Request to send" packet.

Algorithm 1 shows how estimation works. The basic idea is that from the point of view of station *i*, it sees that it is sharing the channel with a group of contentious stations who are competing with it for channel access. Thus we have the notion of "me, and the others". Stations estimate dynamically what throughput they get and what throughput "others" get, and then adjust their contention windows according to the fairness index defined. In other words the contention window is adjusted in order to equalize the throughput obtained by the different stations. A station can estimate roughly how much bandwidth "others" obtain by looking at the packets in its vicinity. For example (the details can be seen in the algorithm) an RTS packet that station *i* sends leads to an increase of its obtained throughput since it used the channel. A received RTS means "others" are trying to obtain the channel and thus it increases "others" obtained throughput, etc.

With this estimation, the EBFMA changed the binary exponential backoff scheme used in IEEE 802.11 DCF. The *fairness index* is defined to be: $FI_e = (\frac{W_{ei}}{\phi_i}) \,/\, (\frac{W_{eo}}{\phi_o})$ and the adjustment of the contention window is shown in Algorithm 2.

Algorithm 2. Contention Window Adjustment

Switch (FI_e) {

case >C:

$CW_{new} = min\ (CW_{new} * 2, CW\_MAX)$

case ($1/C$, C):

$CW_{new} = CW_{old}$

case <$1/C$:

$CW_{new} = max(\ CW_{old} / 2, CW\_MIN)$

}

The variables CW_{new} and CW_{old} are contention window sizes. In this algorithm, C is a constant used to adjust the ability of adapting the algorithm. The smaller the value of C, the more aggressively the contention window size is adjusted and vice versa. However, the choice of C is rather limited. For example, if we choose C = 2, stations would not change their contention windows when the estimated FI is between (0.5, 2) and probability of collision may be high when the number of competing stations is large and load to the channel is high. On the other hand, if C is too close to 1, say 1.01, stations may be busy adjusting their contention windows all the time and the algorithm becomes unstable. The calculation shows that if a station estimates that it has got more share than it should get, it will double its contention window size until it reaches a maximum value (CW_MAX) so that its neighbors can have more chances to recover earlier from the backoff procedure and win access to the channel and vice versa. If a station estimates that it has got only its fair share, it will hold onto its current contention window size.

CHAPTER 4

IMPLEMENTATION, SIMULATION & ANALYSIS

A. Simulation Tool

We use OPNET Modeler [21] as our simulation tool. OPNET Modeler is the industry's leading network technology development environment, which can be used to design and study communication networks, devices, protocols, and applications with unmatched flexibility. Modeler is used by the world's most prestigious technology organizations to accelerate the R&D process. This report is partially supported by OPNET's University Program [22].

B. Implementation

We implemented two MAC protocols: DFS and EBFMA, by simply modifying the existing Process Model in OPNET Modeler.

Because both protocols propose backoff schemes different from that of IEEE 802.11, it is easy to implement these two protocols by modifying the "backoff" part of the existing Process Model "wlan_mac". From Figure 8, one can get some details about the process model "wlan_mac".

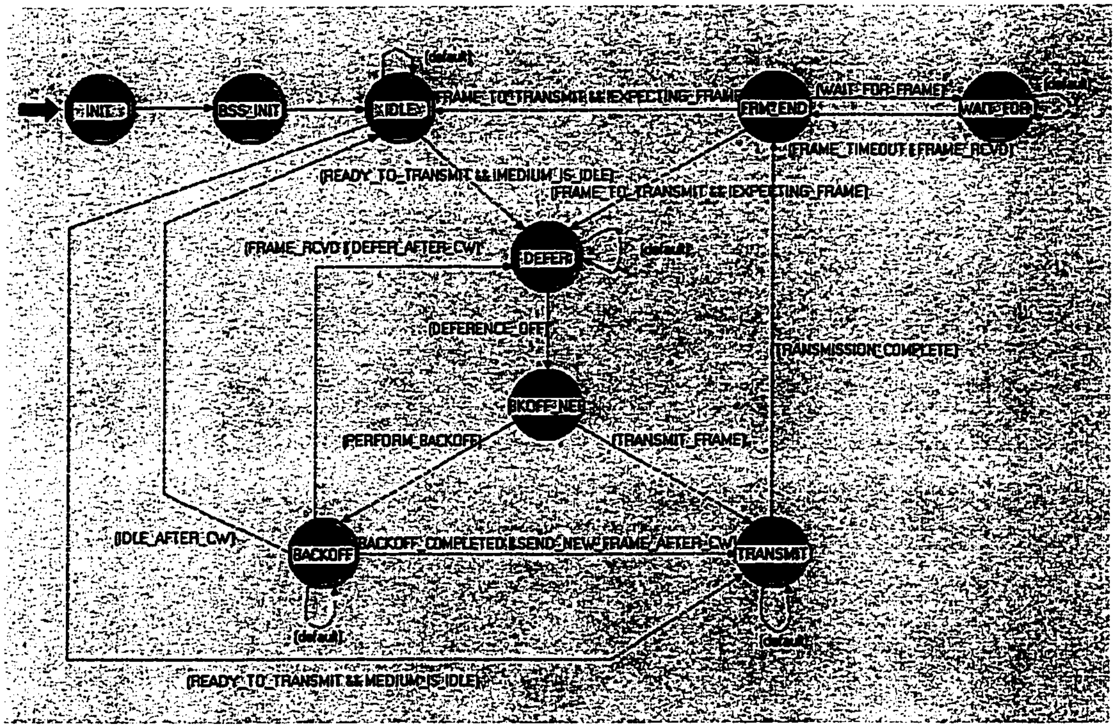

Figure 8. The WLAN MAC Process Model

For the implementation of the DFS protocol, the following is an explanation of some parameters [8]:

- *Scaling_factor* is 0.02. The choice of the *scaling_factor* governs the trade-off between aggregate throughput and fairness.
- *CollisionWindow* is 4 slots.
- There are *n* nodes with *n/2* identical flows of weight *2/n*. Since IEEE 802.11 does not account for weights, equal weights are considered in DFS for its comparison with IEEE 802.11. For the number *n*, it can be found in the Process Model as a variable with the name *bss_stn_count*.

- The random variable ρ in Equation (2) is uniformly distributed in the interval [0.9, 1.1].

Without declaring any variables in the *state variable block* in the Process model, the DFS protocol is implemented with a simple modification to the IEEE 802.11 standard at state "BKOFF_NEEDED" in the Process Model "wlan_mac".

For the implementation of EBFMA, the following is an explanation of some parameters:

- *received packet type* can be found in the *function block* of the Process Model as a variable with the name *rcvd_frame_type*.
- *destID* can be found with the name *destaddress*.
- *localID* can be found with the name *myaddress*.

All parameters in Algorithm 1 and Algorithm 2 can be found in *state variable block* or *function block* in the Process Model except FI_e, W_{ei} and W_{eo}. We declared these three variables in the *state variable block*.

After implementing Algorithm 1 in the Process Model, EBFMA is implemented by simply modifying the IEEE 802.11 standard at state "BKOFF_NEEDED" in the Process Model "wlan_mac".

C. Simulation Environment Setting

We set up two network models in our report. One is an Independent BSS WLAN. A BSS (Basic Service Set) is a set of stations that communicate with one another. When all the stations in the BSS can communicate directly with each other and there is no connection to a wired network, the BSS is called an independent BSS (IBSS). We also call an

Independent BSS WLAN a fully connected network since every station can communicate directly with any other station. The other network model is a partially connected network. In this model, some stations are out of other nodes' transmitting range, so that some nodes cannot communicate with each other directly.

To each network model, we apply the three protocols separately. We study the three protocols performance in the same network model, in the same scenario and with the same simulation parameters. This means we set up the same Network Model, the same Node Model and the same scenario for the simulation of the DFS protocol, the EBFMA protocol and the IEEE 802.11 standard.

Details of the *Traffic Generation Parameters* of nodes in our network models are given below:

- *Start Time* (seconds): constant (0.0).
- *On State Time* (seconds): constant (10.0).
- *Off State* Time (seconds): constant (0.1).
- *Stop Time* (seconds): Never.
- *Packet Generation Arguments*:
    1. *Interarrival Time* (seconds): constant (0.1).
    2. *Packet Size* (bytes): constant(1024).
    3. *Segmentation Size* (bytes): No Segmentation.

Here *Start Time* specifies the distribution name and arguments to be used for generating random start time across different nodes. *On State Time* specifies the distribution name and arguments to be used for generating random outcomes for time spent in the "ON" state, packets are generated only in the "ON" state. *Off State Time* specifies the distribution name and arguments to be used for generating random outcomes for time spent in the "OFF" state. No packets are generated in the "OFF" state. *Stop Time* specifies time (in seconds) at which traffic generation procedures stop from the node. *Interarrival Time* specifies the distribution name and arguments to be used for generating random outcomes for the time between successive packet generations in the "ON" state. *Segmentation Size* determines the size of segments that need to be created before sending a packet out. If set to "No Segmentation", then the packet sent is of the size specified in the "Packet Size" attribute.

We list below the *Wireless Lan Parameters* used in our network models:

- *Rts Threshold* (bytes): None
- *Fragmentation Threshold* (bytes): None
- *Data Rate* (bps): 1 Mbps
- *Physical Characteristics*: Frequency Hopping
- *Short Retry Limit* (slots): 7
- *Long Retry Limit* (slots): 4
- *Access Point Functionality*: Disabled
- *Channel Settings*: Bandwidth (Khz):10; Min Frequency (Mhz): 30
- *Buffer Size* (bits): 265000
- *Max Receive Lifetime* (secs): 0.5

Here *Rts Threshold* is to decide whether or not to transmit an RTS frame for a particular data frame. Where the value is set to "None", RTS frames are never used regardless of the size of the data frame. *Fragmentation Threshold* means that data received from a higher layer greater than this threshold has to be fragmented. *Data rate* can be chosen from 1Mbps, 2Mbps, 5.5 Mbps and 11 Mbps. Based on the flag *Physical Characteristics*, the station sets the appropriate values to the following protocol attributes which are specific to the physical layer characteristics. *Short Retry Limit* is the maximum number of transmission attempts of a frame the size of which is less than or equal to Rts threshold (provided that Rts is enabled). *Long Retry Limit* is the maximum number of transmission attempts of a frame the size of which is greater than Rts threshold (provided that Rts is enabled). *Access Point Functionality* enables or disables the access point feature in the station. *Channel Settings* specifies the bandwidth for 4 channels. Each channel is associated with a data rate of (1,2,5.5 and 11 Mbps). *Buffer Size* specifies the maximum length of the higher layers data arrival buffer. Once the buffer limit is reached the data arriving from higher layers will be discarded until some packets are removed from the buffer. *Max Receive Lifetime* is the maximum time after the initial reception of the fragmented MSDU after which further attempts to reassemble the MSDU will be terminated.

D. Simulations, Data Collection and Analysis

In our report, we analyze the simulation results for the throughput and *Fairness Index* of each protocol.

In this report, we evaluate the *Fairness Index* [10,20] as follows.

$$\text{Fairness Index} = \frac{\left(\sum_f T_f / \phi_f\right)^2}{numberofflows * \sum_f \left(T_f / \phi_f\right)^2} \tag{4}$$

T_f denotes throughput of flow f, and ϕ_f denotes the weight of flow f.

This Fairness Index has the following desirable properties:

- It is dimensionless. The units used to measure the throughput (bits/sec, cells/sec, frames/sec) do not affect its value.
- It is a normalized measure that ranges between 0% and 100%. The maximum fairness is 100% and the minimum 0%. This makes it intuitive to interpret and present.
- If all T_f's are equal, the allocation is fair and the fairness index is one.

A station's throughput, measured in bits/second is the amount of traffic destined for the station that is received by it in one time period. The aggregate throughput is the sum of the throughput of each station in one time period. The weight of each flow is considered equal in this report.

D.1 Comparison of Random Flow and Fixed Flow

First, we study the effects of random flows and fixed flows in the fully connected network. We set up two kinds of flows. One is a fixed flow that is set up from node i to node $i+1$. The other one is a random flow: a flow can be set up from one node to any other node and is set up randomly. In this experiment, in the fully connected network, we set up one scenario that contains 10 nodes. The simulation time is set to 10 seconds. The offered station load is set to 80k bits per second. The transmitting range is set to 300 meters. In the partially connected network, we set up one scenario that contains 10 nodes. The simulation time is set to 10 seconds. The offered station load is set to 80k bits per second. The transmitting range is set to 60 meters. Figure 9 illustrates the layout of the fully connected network. Figure 10 illustrates the layout of the partially connected network.

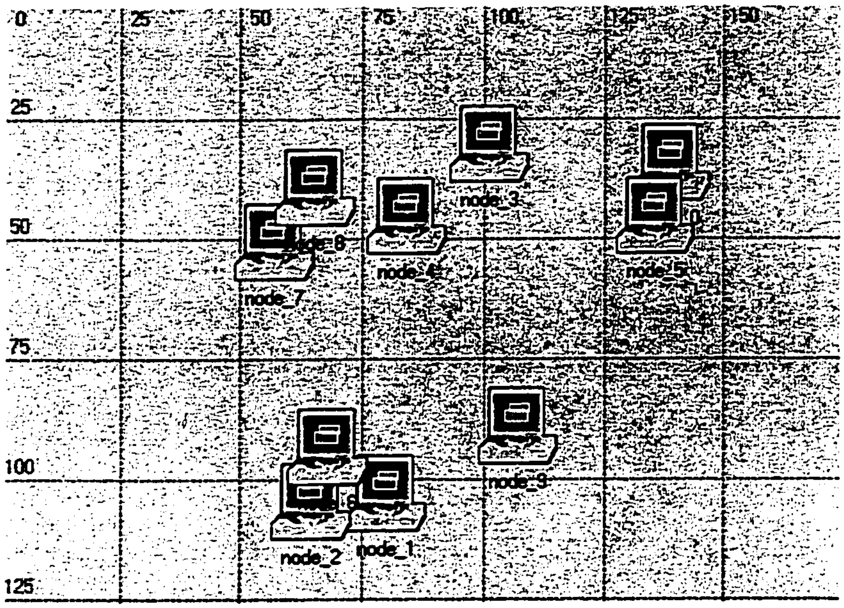

Figure 9. A Fully Connected Network: Transmitting Range 300 m, 10 nodes, 10 seconds, 80kb/s

Table 1. Comparison of Fairness Index (Fully Connected Network: 10 nodes, 10 seconds, 80kb/s)

	DFS	EBFMA	IEEE
Fixed Flows	0.98127	0.99132	0.92461
Random Flows	0.98928	0.98763	0.98687

Table 2. Comparison of Aggregate Throughput (Fully Connected Network: 10 nodes, 10 seconds, 80kb/s)

	DFS	EBFMA	IEEE
Fixed Flows	656999	578354	695502
Random Flows	687310	593103	721714

Table 1 shows the Fairness Index in this fully connected network scenario for fixed flows and random flows. Table 2 shows the Aggregate Throughput in this fully connected network scenario for fixed flows and random flows.

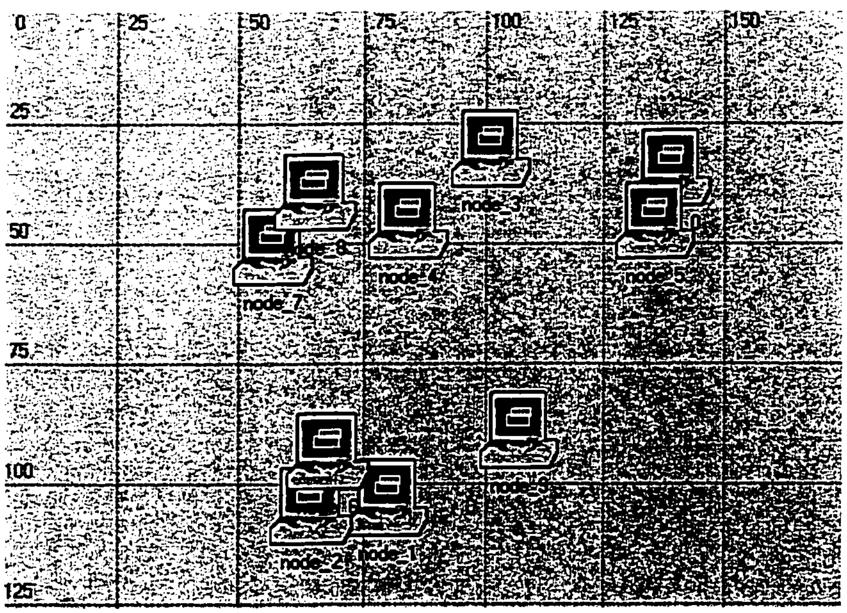

Figure 10. A Partially Connected Network: Transmitting Range 60 m, 10 nodes, 10 seconds, 80kb/s

Table 3. Comparison of Fairness Index (Partially Connected Network: 10 nodes, 10 seconds, 80kb/s)

	DFS	EBFMA	IEEE
Fixed Flows	0.83166	0.63122	0.91261
Random Flows	0.83953	0.62186	0.94144

Table 4. Comparison of Aggregate Throughput (Partially Connected Network: 10 nodes, 10 seconds, 80kb/s)

	DFS	EBFMA	IEEE
Fixed Flows	45688	39912	90712
Random Flows	46696	40960	100762

Table 3 shows the Fairness Index in this partially connected network scenario for fixed flows and random flows. Table 4 shows the Aggregate Throughput in this partially connected network scenario for fixed flows and random flows.

From Table 1 and Table 2, we can see that fixed flows and random flows do not have a significant impact on the fairness index and throughput in the fully connected network. From Table 3 and Table 4, we can also see that in a partially connected network, fixed flows and random flows do not have a significant impact on the fairness index and throughput.

D.2 Comparison of Performance in Fully Connected Network and Partially Connected Network

In this experiment:

- In the fully connected network, we set up one scenario that contains 10 nodes. The simulation time is set to 10 seconds. The offered station load is set to 80k bits per second. The transmitting range is set to 300 meters. Flows are set randomly. Figure 9 illustrates the network layout of this scenario.
- In the partially connected network, we set up one scenario that contains 10 nodes. The simulation time is set to 10 seconds. The offered station load is set to 80k bits per second. It is the same scenario as the above except that transmitting range is set to 60 meters; this causes the network to be only partially connected. Flows are set randomly. Figure 10 illustrates the network layout of this scenario.

Table 5 shows the Fairness Index of the protocols in the fully connected network and in the partially connected network. Table 6 shows Aggregate Throughput of the protocols in the fully connected network and in the partially connected network.

Table 5. Comparison of Fairness Index (10 nodes, 10 seconds, 80kb/s)

	DFS	EBFMA	IEEE
Fully Connected Network	0.98928	0.98763	0.98687
Partially Connected Network	0.83953	0.62186	0.94144

Table 6. Comparison of Aggregate Throughput (10 nodes, 10 seconds, 80kb/s)

	DFS	EBFMA	IEEE
Fully Connected Network	687310	593103	721714
Partially Connected Network	46696	40960	100762

From Table 5 and Table 6, we know that in the partially connected network, the aggregate throughput of all these three protocols is worse than in the fully connected network as expected. It is interesting that the fairness index of all three protocols is also worse in a partially connected network than in a fully connected network. In the partially connected network, IEEE 802.11 has the best fairness index and best throughput, and DFS has better fairness index and throughput than EBFMA in the partially connected network. We will give a deeper study of the performance of the protocols in the fully connected network in the next section. For performance of the protocols in the partially connected network, we set up another partially connected network scenario that contains 20 nodes. The simulation time is set to 10 seconds. The offered station load is set to 80k bits per second, random flows are set up; the transmitting range is 60 meters. Figure 11 illustrates the network layout of this scenario.

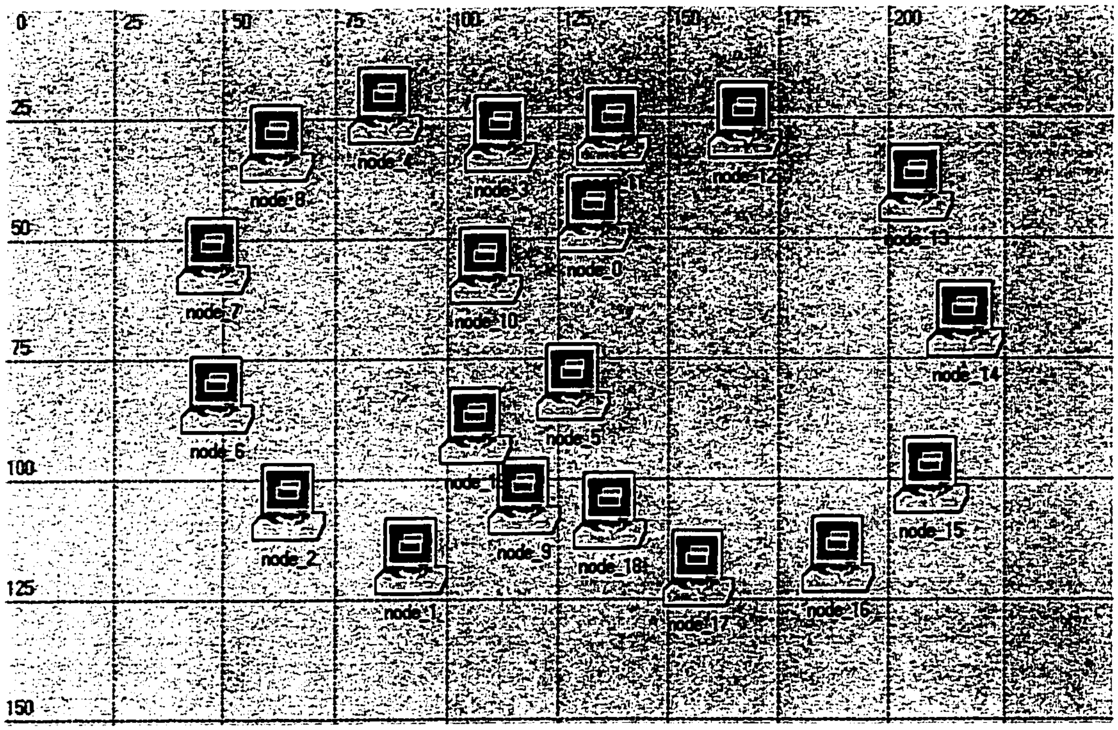

Figure 11. A Partially Connected Network: Transmitting Range 60 m, 20 nodes, 10 seconds, 80kb/s

Table 7 shows the fairness index and aggregate throughput of the protocols in the partially connected network with 20 nodes, where transmitting range is 60 m, simulation time is 10 seconds and station load is 80kb/s.

Table 7. Fairness Index & Aggregate Throughput (Partially Connected Network: 20 nodes, 10 seconds, 80kb/s)

	DFS	EBFMA	IEEE
Fairness Index	0.46448	0.56877	0.73981
Aggregated Throughput	39,319	37,682	84,377

The information we get from Table 7 enhances the conclusion that in a partially connected network, IEEE 802.11 has the best fairness index and best throughput. This may be because in IEEE 802.11, the contention window size reduces much faster than it increases. DFS has better throughput than EBFMA in the 20-node scenarios as well; however its fairness index is worse than EBFMA in this case.

D.3 Performance in Fully Connected Network

In this section, we study the performance of the protocols in fully connected networks.

In the first set of experiments, we set up several scenarios in the fully connected network. The simulation time is set to 10 seconds and the offered station load is set to 80k bits per second. In the condition, 19 scenarios were set up. In each scenario, the network contains 10, 12, 13, 16, 19, 22, 25, 28, 31, 34, 37, 40, 43, 46, 49, 52, 56, 60 and 64 stations in sequence. Flows are fixed. Figure 12 illustrates one scenario of the network with 64 stations.

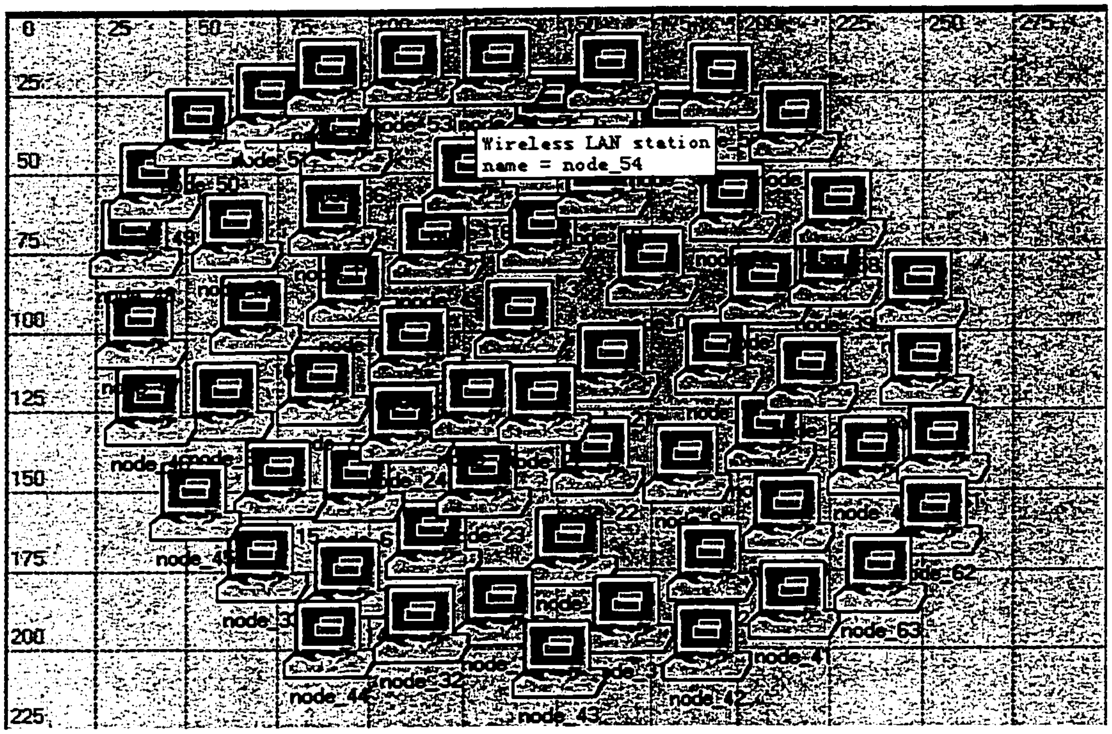

Figure 12. A Fully Connected Network: Transmitting Range 300 m, 64 nodes, 10 seconds, 80kb/s

Figure 13 shows the variation in fairness index with the number of nodes in a fully connected network with these scenarios. With the increase in the number of nodes, the fairness index appears to decrease. The main reason may be traffic generating parameters. In our report, we study the comparison of fairness index between different protocols, so we ignore trend of the fairness index decreasing with increasing numbers of nodes. Figure 14 shows the variation in throughput/weight with the number of nodes in the same scenarios.

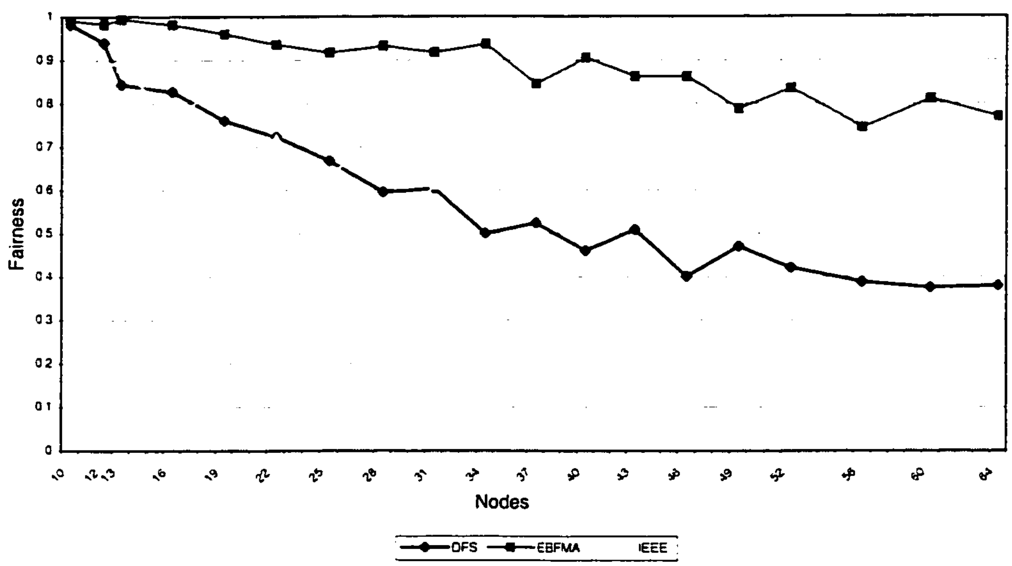

Figure 13. Comparison of Fairness Index (10 seconds, 80kb/s)

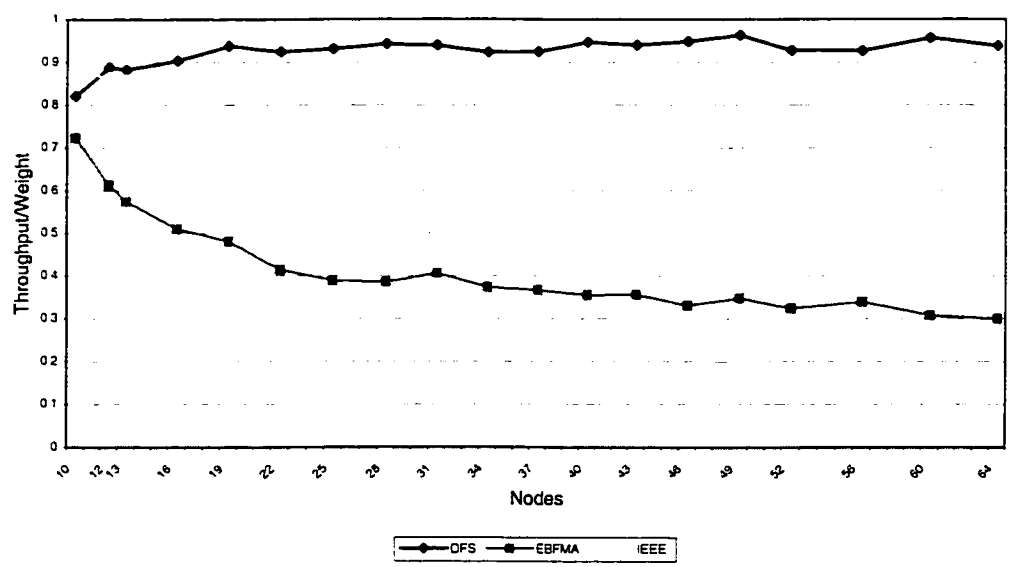

Figure 14. Comparison of Throughput/Weight (10 seconds, 80kb/s)

From Figure 13 and Figure 14, we can know although EBFMA has the best fairness index, it has the worst throughput. The fairness index of IEEE 802.11 is a little bit better than DFS when the number of nodes is at least 28, but for a smaller number of nodes, there is no significant difference in their fairness. However, DFS has the best throughput.

The poor throughput performance of EBFMA in the fully connected network is because of its tendency to reduce the contention window size for all nodes in this situation, which leads to more collisions.

In the second set of experiments, we set up scenarios that contain 10 stations in the fully connected network. In this network, 8 scenarios were run separately. The simulation time is set to 4 seconds, 6 seconds, 10 seconds, 15 seconds, 20 seconds, 30 seconds, 1 minute and 2 minutes. In this situation the offered station load is set to 80k bits per second. Flows are fixed. Transmitting range is 300 meters. Figure 15 illustrates the network layout of these scenarios.

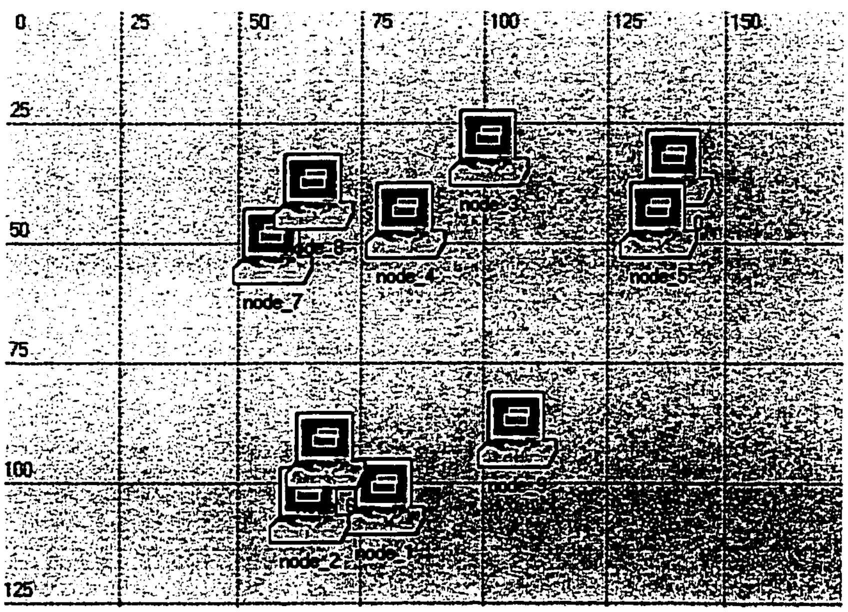

Figure 15. A Fully Connected Network: Transmitting Range 300 m, 10 nodes, 80kb/s

Figure 16 shows the variation in fairness index with the time in the above scenarios. Figure 17 shows the variation in throughput/weight with the time in the above scenarios.

From Figure 16 & Figure 17 we verify again that EBFMA has the worst throughput but the best fairness index. In these scenarios, the IEEE 802.11 has the best throughput while the time is short. As the time becomes longer, the DFS has the best throughput. After 10 seconds, the fairness index of DFS is better than IEEE's and is a little bit lower than EBFMA's.

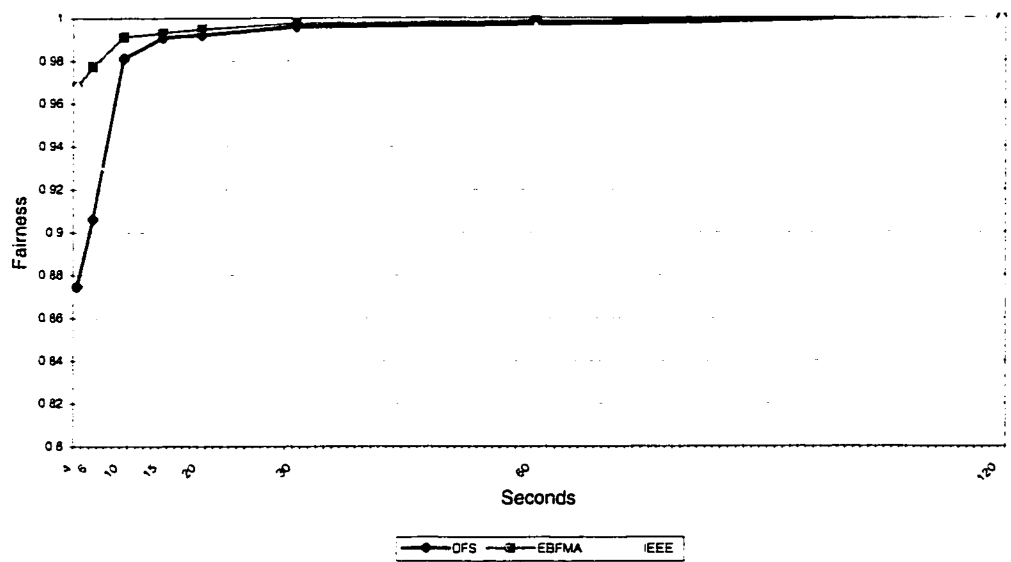

Figure 16. Comparison of Fairness Index (10 nodes, 80kb/s)

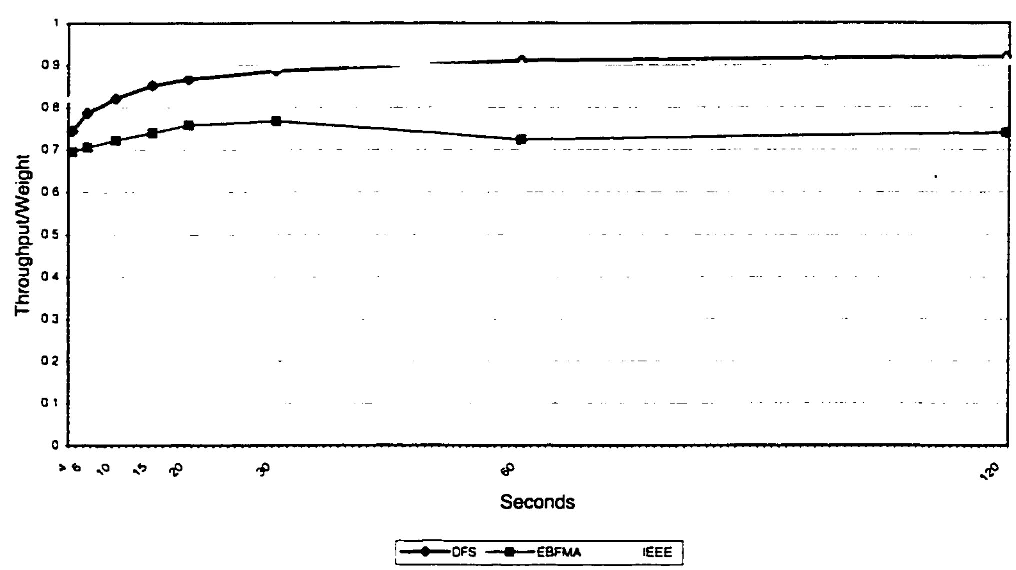

Figure 17. Comparison of Throughput/Weight (10 nodes, 80kb/s)

In the third set of experiments, we set up scenarios containing 20 stations in the fully connected network. In this network, 8 scenarios were run separately. The simulation time is set to 4 seconds, 6 seconds, 10 seconds, 15 seconds, 20 seconds, 30 seconds, 1 minute and 2 minutes. In these scenarios the offered station load is set to 80k bits per second, and the transmitting range is 300 meters. The traffic flows are fixed. Figure 18 illustrates the network layout of these scenarios.

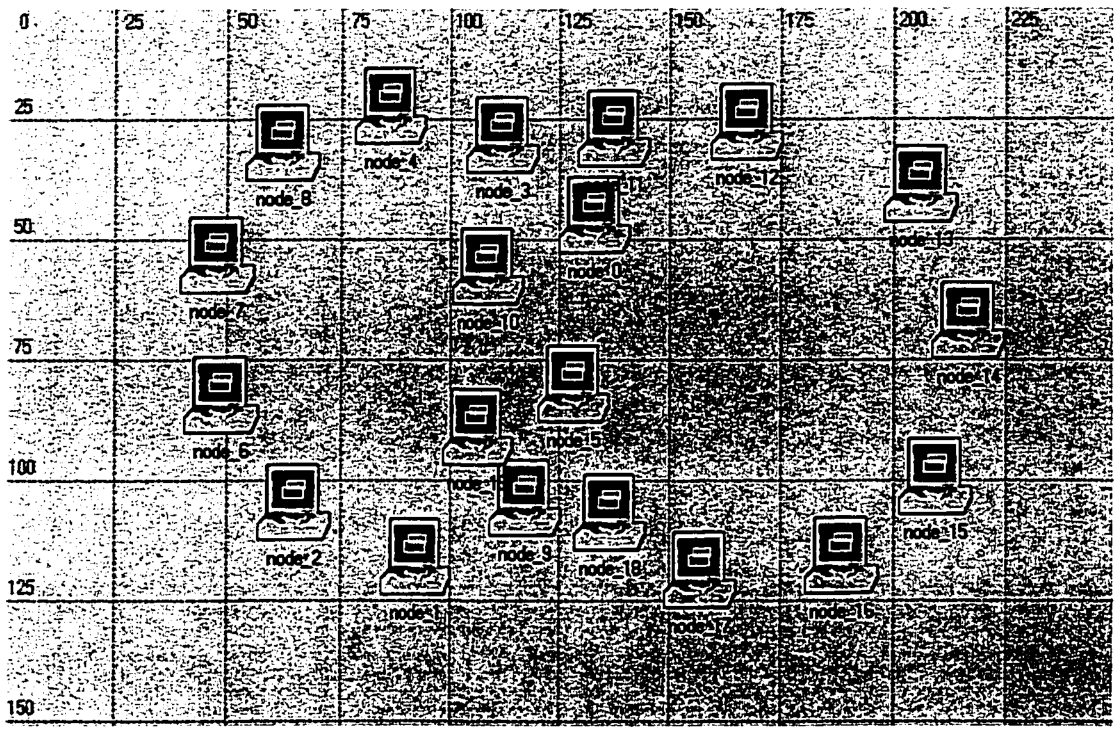

Figure 18. A Fully Connected Network: Transmitting Range 300 m, 20 nodes, 80kb/s

Figure 19 shows the variation in fairness index with the time in the above scenarios, while Figure 20 shows the variation in throughput/weight with the time.

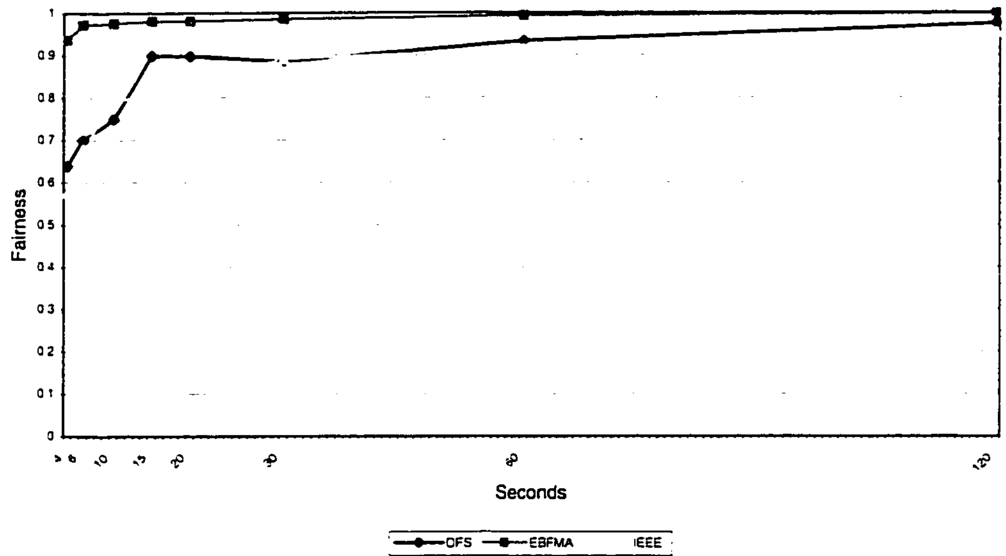

Figure 19. Comparison of Fairness Index (20 nodes, 80kb/s)

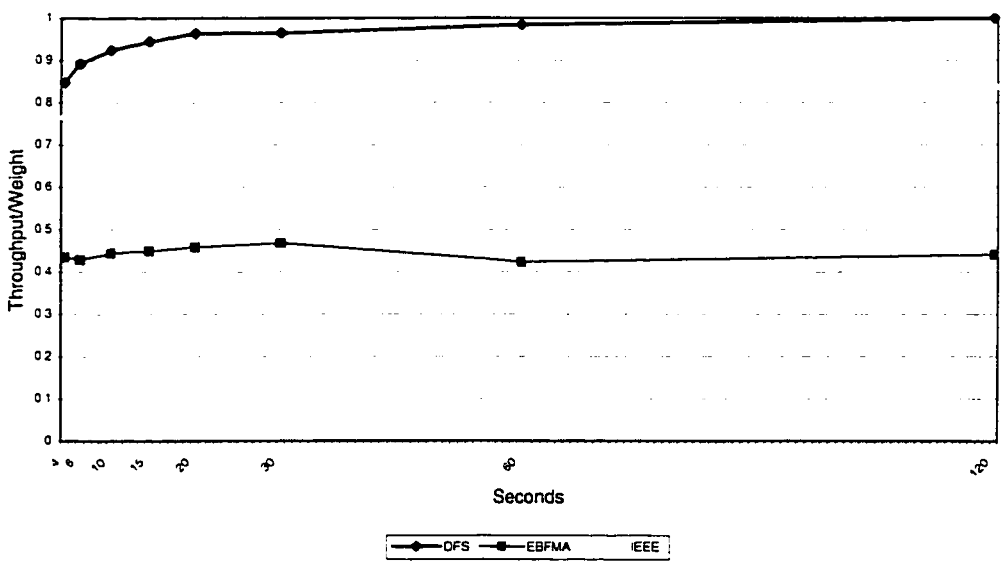

Figure 20. Comparison of Throughput/Weight (20 nodes, 80kb/s)

From Figure 19 & Figure 20 we verify that EBFMA has the worst throughput but the best fairness index. DFS has the best throughput. With increasing simulation time, the fairness index of DFS becomes better than that of IEEE.

CHAPTER 5

CONCLUSION

This report investigates the fairness problem and introduces the wireless MAC protocols IEEE 802.11, the DFS protocol and the EBFMA protocol. The DFS and EBFMA protocols are implemented by modifying the existing IEEE 802.11 model in the OPNET Modeler 8.0 simulator. The analysis of the simulation results shows that in the fully connected network, although EBFMA has the best fairness index, it has the worst throughput. In the fully connected network, the fairness index of IEEE 802.11 and that of DFS do not have a significant difference while DFS has better throughput than IEEE. The analysis of the results also shows that in the partially connected network, IEEE 802.11 has the best fairness index and throughput while DFS & EBFMA have far worse fairness index and throughput. More experiments are needed to study the effect of the topology of the partially connected network on the fairness index and throughput. Future work should attempt to find a protocol that has good throughput as well as fairness index in both partially and fully connected networks.

REFERENCES

[1] R. O. Baldwin, "Improving the Real-time Performance of a Wireless Local Area Network," Ph.D. Disertation, Bradley Department of Electrical and Computer Engineering, 1999.

[2] B. Bensaou, Y. Wang, and C. C. Ko, "Fair Medium Access in 802.11 based Wireless Ad-Hoc Networks," MOBIHOC'00.

[3] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: A Media Access Protocol for Wireless LAN's," SIGCOM'94.

[4] B. P. Crow, I. Widijaja, J. G. Kim, and P. T. Sakai, "IEEE 802.11 Wireless Local Area Networks," IEEE Communications Magazine. September 1997.

[5] European Telecommunications Standards Institute, Spectrum Allocation RES.2, 1998.

[6] S. J. Golestani, "A self-clocked fair queueing scheme for broadband applications," IEEE INFOCOM, 1994.

[7] P. Goyal, H. Vin, and H. Cheng, "Start-time fair queuing: a scheduling algorithm for integrated services packet switching networks," SIGCOMM'96, Stanford, CA.

[8] S. Gupta, "Study of distributed fair scheduling in a wireless Local Area Networks," Master of Science thesis, Texas A&M University, August 2000.

[9] X. L. Huang, and B. Bensaou, "On Max-Min Fairness and Scheduling in Wireless Ad-Hoc Networks: Analytical Framework and Implementation," MOBIHOC'01.

[10] IEEE, "IEEE std 802.11 - wireless LAN medium access control (MAC) and physical layer (PHY) specifications," 1997.

[11] R. Jain, G. Babic, B. Nagendra, and C. Lam, "Fairness, call establishment latency and other performance metrics," Tech. Rep. ATM Forum/96-1173, ATM Forum Document, August 1996.

[12] S. Lu, V. Bharghavan, and R. Srikant, "Fair Scheduling in Wireless Packet Networks," SIGCOMM'97.

[13] H. Luo, S. Lu, and V. Bharghavan, "A new model for packet scheduling in multihop wireless networks," MOBICOM'00.

[14] T. Nandagopal, T. Kim, X. Gao, and V. Bharghavan, "Achieving MAC Layer Fairness in Wireless Packet Networks," MOBICOM'00.

[15] National Science Foundation, "Research priorities in wireless and mobile communications and networking," Report of a workshop held March 24 - 26, 1997, Airlie House, Virginia. Available at http://www.cise.nsf.gov/anir/ww.html.

[16] T. Ozugur, M. Naghshineh, P. kermani, C. M.Olsen, B. Rezvani, and J. A. Copeland, "Analysis and Simulation of a Fair Queuing Algorithm," SIGCOMM'89.

[17] A. K. Parekh, and R. G. Gallager, "A generalized processor sharing approach to control in integrated services networks: the single node case," IEEE/ACM Transactions on Networking, vol. 1, pp. 344-357, 1993.

[18] P. Ramanathan, and P. Agrawal, "Adapting Packet fair Queuing Algorithms to Wireless Networks," MOBICOM'98.

[19] N. H. Vaidya, and P. Bahl, "Fair scheduling in broadcast environments," Tech. Rep. MSR-TR-99-61, Microsoft Research, Dec. 1999.

[20] N. H. Vaidya, P. Bahl, and S. Gupta, "Distributed fair Scheduling in a Wireless LAN", MOBICOM'00.

[21] www.opnet.com.

[22] www.opnet.com/services/university/univ_use_opnet_list.html.