Simulation Modelling of QoS Enhancements in IEEE 802.11 Networks and the Effects of Channel Modelling

A dissertation submitted to the Department of Electrical Engineering,
Faculty of Engineering and the Built Environment at the University of Cape Town
in fulfilment of the requirements for the degree of

Master of Science

in

Electrical Engineering

— Tinotenda Leslie Mundangepfupfu —

Supervisor

Professor Pieter S Kritzinger
University of Cape Town, South Africa

Co-supervisor

Neco Ventura
University of Cape Town, South Africa

UNIVERSITY OF CAPE TOWN
IYUNIVESITHI YASEKAPA • UNIVERSITEIT VAN KAAPSTAD

I know the meaning of plagiarism and declare that all the work in the document, save for that which is properly acknowledged, is my own.

ABSTRACT

The IEEE 802.11 Medium Access Control (MAC) Protocol is a Wireless Local Area Networking (WLAN) technology. It implements the Distributed Coordinated Function (DCF) and the Enhanced Distributed Channel Access (EDCA). EDCA adds Quality-of-Service (QoS) enhancements, to manage access amongst stations (STAs) sharing the same transmission channel. It is important that the medium access control mechanism performs optimally and impedes QoS as little as possible.

The wireless channel modulation scheme used with these protocols is Orthogonal Frequency Division Multiplexing (ODFM) modulation. It is important to understand how the channel conditions resulting from this modulation scheme, as well as external factors, such as noise, interference and fading, affect network performance.

In this work the impact of the channel effects in modelling EDCA and DCF protocols of the IEEE 802.11 Medium Access Control (MAC) protocol on overall network performance is investigated. This work also looks at the QoS enhancements in these networks, and how the protocols handle QoS-sensitive applications. This is achieved by developing a simulator that abstracts the EDCA and DCF protocols and the OFDM modulation scheme with fading and noise and is used to conduct experiments that measure network performance operating under realistic workload conditions. A multimedia workload model is developed, which constitutes of Voice over IP (VoIP), video, P2P and HTTP traffic.

The experiments conducted measure network performance using six performance metrics. There are four system performance metrics: Normalised Aggregate Throughput, Bit Error Rate, Collision Probability, and Channel Efficiency and two user experience metrics: Access Delay and Number of Dropped Packets.

The experiments conducted are divided into two sets: scalability experiments and comparison experiments. The scalability experiments measure network performance for an increasing number of STAs, and the comparison experiments measure the network performance for an increasing Signal-To-Noise Ration (SNR), comparing performance for each traffic type and overall performance, of both 802.11 protocols.

The experimental results verify the simulator with respect to a hardware test bed and show that the network performance is largely dependent on wireless channel conditions, and the EDCA protocol is better suited to QoS-sensitive applications.

Future work includes evaluating the performance of the 802.11n protocol and to see how it manages QoS-sensitive applications.

ACKNOWLEDGEMENTS

A special thank you to the following people for their invaluable input and willingness to sacrifice some of their time to help develop this project:

- Professor Pieter Kritzinger for his guidance and supervision and also whose continuous motivation kept the project going.
- The Communications Research Group (CRG) for all the resources and support they gave and for making me part of their team.
- The Business Systems Group (BSG) HR team for always being available to deal with my Department of Home Affairs permits.

I would like to thank Rumbidzai Marufu for her encouragement and for providing a second eye on the dissertation.

A big thank you goes to my family my mother, Doreen, and my brother, Tionei, for all their support and encouragement during my stint at university. I could not have gone through with it without you.

Finally, I would like to thank God Almighty, for the special gift of life and for always guiding and walking with me every day of my life.

Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

Wireless Local Area Networks (WLANs) have become ubiquitous in personal and private enterprise networking and Internet infrastructure, mainly due to the advantages of wireless networking, such as rapid and easy deployment, and low maintenance costs. In recent years, much attention has been paid to the efficient design of such networks [4] in an attempt to utilise the scarce wireless resource. Working Group 11 was formed as part of Project 802 of the Institute of Electrical and Electronic Engineers (IEEE) to recommend an international standard for WLANs. The most recent version of the IEEE 802.11 [48] provides a detailed description for both Medium Access Control (MAC) and Physical (PHY) layers of this technology.

Two IEEE 802.11 WLAN protocols, the Distributed Coordination Function (DCF) and the Enhance Distributed Channel Access, which add Quality-of-Service enhancements, play an important role in gauging network performance and hence, much work has been done to model this protocol for network performance analysis under different traffic conditions. Typically, performance is measured in terms of throughput in the network, operating under specific topological and traffic configurations.

Three approaches have been used to investigate the impact of the DCF on network performance, namely Analytic Modelling, Simulation, and Hardware Experimentation.

Analytic models are most preferred, since they require fewer resources in terms of execution time, memory management and equipment. However, this type of model typically makes relatively more and generalising assumptions than the other approaches, such as using fixed probabilities and ideal channel conditions [4], in order to reduce model complexity. Although they have proven to produce reasonably accurate results, the analytic intractability of a problem often makes it impossible to develop an acceptable model.

A simulation study is typically a more complicated exercise than analytic modelling, mainly because it is resource intensive and time consuming, and is often used to validate the an analytic model. Simulator complexity depends on the abstraction level of the system model. Simulation environments, such as OMNeT++ [46] and NS-2 [45], were developed to aid the development of simulation models.

Hardware experimentation is rarely done, since it requires relatively higher cost, more time, and greater effort in terms of deployment and maintenance. A hardware experiment, executed on a hardware testbed, is therefore

typically a small-scale version of the actual system and is deployed in a controlled environment. Even so, such experiments can produce more accurate results, when done properly, since they capture all of the expected salient features of the system being modelled. Examples of wireless network testbeds are the MIT Roofnet [5], the UCSB Meshnet Testbed [40], the Meraka Wireless Grid [30], and the DNA Testbed [67].

1.1 PROBLEM STATEMENT

Wireless is currently a limited Megabit-capacity technology more prone to channel errors compared to Gigabit-capacity wired networks. It is important that careful consideration be made when deciding on medium management protocols for wireless networks. The performance degradation induced by the protocol must not reduce the network performance to the extent that communication over the network is impossible or impractical.

Over the years there has been a continual increase in the number of wireless networks deployed, and many organisations prefer these networks. This means that such networks should be able to handle QoS-sensitive applications, such as voice calls. However, the original wireless networks were designed for data only and not to handle real-time traffic well. This raises the need to understand how these networks handle applications such as real-time voice. There is also a need to understand how the realistic channel conditions affect the performance of the network.

In this dissertation, a discrete event-driven simulator, modelling the 802.11 Wireless Local Area Network EDCA and DCF standards [1], OFDM modulation with Rayleigh fading and AWGN, and a suitable multimedia workload model, is developed from the ground up. The performance of EDCA and DCF is investigated using this simulation model, operating under different channel model configurations and a realistic workload. This performance is measured in terms of the following:

- Normalised aggregate throughput, which is the rate at which the transmission channel is utilised.
- Bit error rate, which is the rate at which transmission errors occur in the network.
- Collision probability, which is the probability that a packet collides with at least one other in the network on transmission.
- Channel efficiency, which is measured as 1 (collision probability).
- Access delay, which is the time from which a packet becomes available for transmission to the time it is successfully transmitted.
- Number of dropped packets, which refers to the number of packets dropped due to transmission errors or collisions.

Our investigations include

1. an assessment of the network performance under different channel conditions (from low to high bit energy-to-noise ratios), and
2. overall network performance with an increasing number of Stations (STAs) and

3. a comparison of the EDCA protocol to DCF network performance to understand the protocol that is better suited for QoS-sensitive applications.

The investigations offer insight to the following questions:

1. To what extent does realistic channel behaviour influence network performance results?
2. What are the relative advantages and disadvantages of the simplifying assumptions in modelling a part of the IEEE 802.11 standard?
3. How do the EDCA and DCF protocols performance scale with an increasing number of STAs?
4. Which protocol is more suited to a particular situation?

1.2 RESEARCH CONTRIBUTIONS

Five major contributions are made in this project:

1. A *Performance Framework and its associated Models* for a fully-connected Point-to-Multipoint (PMP) wireless 802.11 wireless network abstracting the standards EDCA and DCF MAC protocols.
2. A modular *Channel Model* consisting of the OFDM modulation scheme, Rayleigh multi-path fading, and AWGN.
3. A *Multimedia Workload Model*, which generates VoIP, Video, P2P, and HTTP traffic.
4. A Discrete-Event Simulator implementation of the performance model proposed by the first three contributions. It comprises of a Machine Model, a Channel Model, and a Workload Model.
5. Two sets of experiments, one that compare EDCA to DCF in terms of scalability, and another that compares the protocols under various channel conditions. The experiments aim to show how channel conditions affect network performance, how the protocols perform with a increasing number of STAs, and what protocol is best suited for QoS-sensitive applications.

The performance framework and its associated models are a high-level abstraction of the system to be modelled that fully describes the network components and how they operate. A System Performance Model (SPM) is developed regarding a Scheduling Framework as a component and operation guideline (see Sections 4.1.2 and 4.2.2). The SPM is then further detailed to present the Network-of-Queues model which, in turn, can more easily be translated into the executable model of choice a simulation model in this case. Two performance frameworks are delivered in this work, one for EDCA and one for DCF.

As mentioned above, the discrete-event simulator consists of three components; a Machine Model, a Channel Model, and a Workload Model.

The machine model is an abstraction of the EDCA and DCF of the 802.11 MAC Standard [48]. It contains standard-specific components that are used by the STAs to access the wireless medium. A STA is any device that

contains an IEEE 802.11-conformant medium access control (MAC) and physical layer (PHY) interface to the wireless medium [48], and the transmission medium is the medium used to implement the transfer of protocol data units (PDUs) between peer PHY entities of a wireless local area network [48]. All the traffic from the STAs is directed towards an Access Point (AP), which is any entity that has STA functionality and provides access to the distribution services via the wireless medium for associated STAs [48]. The DCF and EDCA protocols are described in Sections 2.2 and 2.3, respectively, in more detail.

The Channel Model is an OFDM modulation channel, with Rayleigh multi-path fading and Additive White Gaussian Noise (AWGN). The channel model is used by the machine model to determine whether a packet is transmitted successfully or not, and it returns the number of errors that occur in each transmitted packet. A count of zero means that a packet is received successfully; otherwise it is transmitted with errors.

The Workload Model generates four different types of traffic, hence multimedia workload model. It generates VoIP, video, P2P, and HTTP traffic types. Each traffic type is linked to its corresponding Access Category (AC) in EDCA, but a traffic type is randomly picked for DCF.

Experiments are conducted to measure the performance of the EDCA and DCF protocols for the network operating under each realistic channel and workload conditions. The first experiment validates the channel model by comparing the DCF, with and without realistic channel conditions, to a hardware testbed to determine whether channel modelling has any effects, and if so, if it is accurate. The main experiments are split into two sets; scalability experiments and comparison experiments. The scalability experiments are concerned with how the two protocols fare, in terms of overall network performance, as the number of STAs in the network increases, and this is done for different channel conditions. The comparison experiments are concerned with comparing the performance for an increasing signal-to-noise ratio (SNR). These experiments determine the overall performance but also determine the performance of each traffic type, in order to see which traffic type gets the best service.

All simulator components are implemented in Java.

Java also gives the developer the flexibility to choose the salient features, which may have an appreciable impact on the systems performance, to implement without having to be too concerned about the limitations of some simulation development tool.

1.3 REPORT OUTLINE

The dissertation is structured as follows:

Chapter 2: Background Related Work

This chapter provides the reader with necessary technical knowledge about the EDCA, DCF, channel modelling, and workload modelling. Work done on performance evaluation of the 802.11 protocols is described for each of the three modelling approaches introduced above, as well as previous channel and workload models.

Chapter 3: Frugal Modelling

In this chapter, salient features for each of the system components are identified and discussed. It specifies the simulator requirements and the features that it should possess. The identified features are included in the system.

Chapter 4: Design

This chapter is a breakdown of a complete and well-detailed design of all the system components.

Chapter 5: Implementation and Testing

Chapter 5 describes the implementation process and testing.

Chapter 6: Experimentation

Experimentation is discussed in Chapter 6, reporting the experimental design, model parameterisation, results, and findings.

Chapter 7: Conclusions and Future Work

The report is concluded in Chapter 7, and possible future extensions of this work are briefly discussed.

CHAPTER 2

BACKGROUND AND RELATED WORK

Any ongoing research is part of a bigger realm of work, and thus, it needs to be related to the work already done. This ensures that the work attains an overall relevance and purpose. The review of literature thus becomes a link between the new research done and the studies already done.

It tells the reader about aspects that have been already established or concluded by other authors and also gives a chance to the reader to appreciate the evidence that has already been collected by previous research and thus puts the current research work in the proper perspective. It also gives the reader the opportunity to understand any improvements, or simply a different perspective to the field of study in question.

This section discusses previous work done and published by other authors in order to fully understand the origins and the contributions of this work.

2.1 IEEE 802.11 HISTORY

802.11 is a collection of specifications developed by the Institute of Electrical and Electronics Engineers (IEEE) [48] for WLANs. These specifications describe and define an wireless-based interface between a wireless client, or station and an Access Point, or between two or more wireless clients.

According to data released by the Wi-Fi Alliance and In-Stat, sales of Wi-Fi chipsets were estimated at 300 million units for 2007 [55]. This represents a 41 percent growth rate from 2006, in which 213 million chipsets were shipped [55]. In-Stat predicts that by 2011 approximately 700 million devices will ship with Wi-Fi on board [55]. Nearly 50 percent of the chipsets sold in 2008 are predicted to adhere to the 802.11n draft standard [56].

The Wi-Fi technology sprang into existence as a result of a decision in 1985 by the United States Federal Communications Commission (FCC) to open several bands of the wireless spectrum for use without a government license [25]. These so-called garbage bands were already allocated to equipment such as microwave ovens. To operate in these bands though, devices are required to use spread spectrum technology, e.g. the Direct Sequence Spread Spectrum. This technology spreads a radio signal out over a wide range of frequencies making the signal less susceptible to interference and difficult to intercept.

In 1990, a new IEEE work group, called 802.11, was set up to look into getting a ratified standard started. It

wasn't until 1997 though, some 8 or 9 years later, that this new standard was published, although pre-standard devices were already being manufactured and shipped.

Two variants were ratified over the next two years 802.11b which operates in the Industrial, Medical and Scientific (ISM) band of 2.4 GHz and 802.11a which operates in the bands of 5.3 GHz and 5.8 GHz.

Wi-Fi's popularity took off with the growth of high-speed broadband Internet access in the home. It was and still remains, the easiest way to share a broadband link between several computers located in various parts of a home. The growth of hotspots, free and fee-based public access points, have added to Wi-Fi's popularity. A third variant, 802.11g was ratified in June 2003. It uses a more advanced form of modulation called Orthogonal Frequency-Division Multiplexing (OFDM). Using the 2.4 GHz band, 802.11g can achieve speeds of up to 54 Mbps. Other amendments have been approved since then and in 2007, working group 802.11 ratified another standard which also specified Quality-of-Service enhancements, called 802.11e. The latest standard is the 802.11n technology, which achieves data rates of up to 600 Mbps.

This work covers the 802.11g and 802.11e standards. The 802.11n standard was yet to be ratified when this work commenced but could be considered for future work.

2.2 DCF

The Distributed Coordination Function (DCF) is the medium access scheme of the IEEE 802.11 WLAN standard. It is a contention-based Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) random access scheme, where all Stations (STAs) wanting to transmit adhere to the DCF protocol in order to negotiate channel access between them.

In CSMA/CA, each STA wanting to transmit must sense the channel idle for a specified interval, called the Distributed InterFrame Space (DIFS). If the channel is sensed idle for this DIFS interval, the STA must back-off for a random number of slots using binary exponential back-off rules before it may transmit [48]. The back-off process is completed when the particular STAs back-off counter has decremented to zero.

If a STA senses the channel busy during its own back-off period, it suspends its back-off process, and must sense the channel idle for another DIFS interval before continuing the back-off process.

Once the back-off process is complete, the STA transmits its data. The purpose of the back-off procedure is to reduce the probability of collision, since there is a high chance that all STAs try to transmit after sensing the channel idle for the DIFS interval. Collision can still occur after backing off as two or more STAs can decrease their counters to zero simultaneously, but the chance of it happening is significantly lower than that which exists after listening to the channel for the DIFS interval.

CSMA/CA differs from Carrier Sense Multiple Access with Collision Detection (CSMA/CD) in that, whereas in CSMA/CA a STA wishing to transmit listens to the channel for a pre-determined length of time, in CSMA/CD it does not, but will terminate transmission as soon as a collision is detected.

The standard specifies two DCF access mechanisms, namely; Basic Access and Request-To-Send/Clear-To-Send (RTS/CTS).

The Basic Access scheme, shown schematically in Figure 2.1, is a two-way handshaking scheme where the

sending STA sends a data packet to the destination STA immediately after its backing off period, where on successful reception of this packet, the destination STA waits for a Short InterFrame Space (SIFS) interval before responding with an acknowledgement (ACK). This notifies the sender of successful transmission; otherwise the sender retransmits the packet.

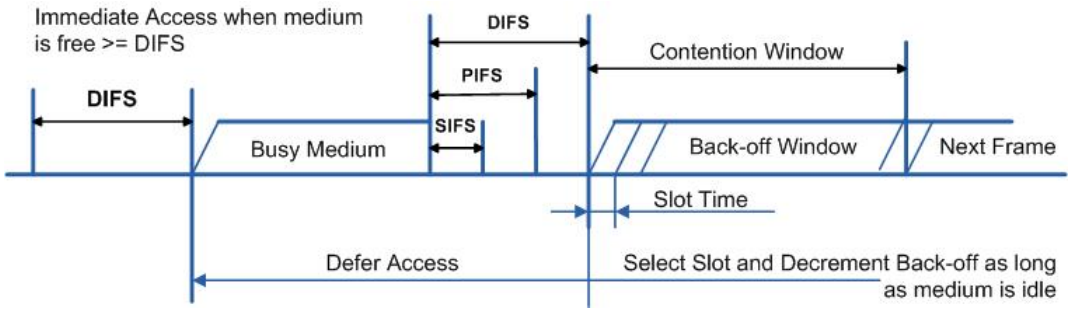

Figure 2.1: Basic Access: Illustration of the placement of InterFrame Spaces for DCF, adapted from the IEEE 802.11 Standard [48]

RTS/CTS, shown in Figure 2.2, is a four-way handshaking scheme that requires the sender to send an RTS control frame immediately after backing off indicating the duration of the potential transmission, after which the destination STA waits for a SIFS interval before responding with a CTS control frame. This notifies all STAs that receive the RTS and CTS frames that the channel is being allocated to the sender of the RTS and the other STAs update the Network Allocation Vectors (NAVs) so that they do not access the channel during this time. Once the sender has received the CTS control frame, it transmits its packet and waits for an ACK frame.

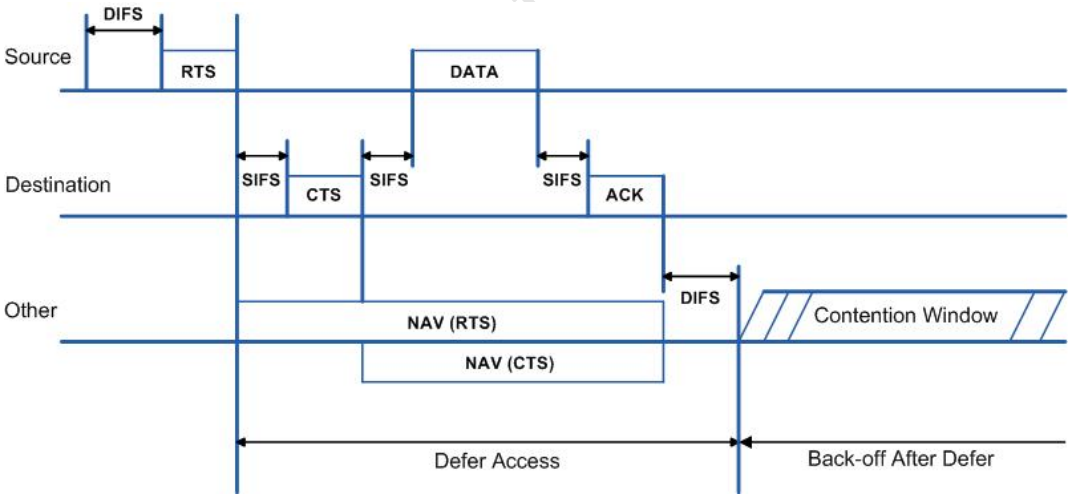

Figure 2.2: RTS/CTS: Illustration of the placement of InterFrame Spaces for DCF, adapted from the IEEE 802.11 Standard [48]

The use of the RTS/CTS mechanism reserves the channel for communication between the sender and the receiver. The RTS and CTS frames notify other STAs in the network to suspend their transmission for the duration of the packet transmission. This mechanism is motivated by the cost of a data packet collision normally being considerably greater than that of an RTS collision. It also eliminates the hidden node problem. All STAs accessing the same Access Point (AP) may not see each other; therefore, at least two STAs can sense the channel to be idle at the same time because they would not have sensed that the other node has already accessed the channel. In some cases, the network can dynamically switch between these two schemes depending on some network parameters.

The Point Coordination Function (PCF) InterFrame Space (PIFS), shown in Figure 2.1, allows PCF enabled STAs to wait for a PIFS rather than a DIFS to occupy a wireless medium [48]. The PCF is out of the scope of this work and will not be discussed.

2.3 EDCA

The IEEE 802.11-2007 specification [48] specifies QoS enhancements. The Hybrid Coordination Function (HCF) combines functions from the DCF and the Point Coordination Function (PCF) with QoS-specific mechanisms and frame subtypes to form the Enhanced Distributed Channel Access (EDCA) and the HCF Controlled Channel Access (HCCA), respectively, to allow a uniform set of frame exchange sequences to be used for QoS data transfers during both the Contention Period (CP) and Contention-Free Period (CFP).

In this work, however, focus is placed on the EDCA, since it is more widely used than the HCCA. It is also contention-based as the DCF so this allows for comparison between EDCA and DCF.

The EDCA mechanism provides contention-based, differentiated, distributed channel access to the Wireless Medium (WM) using eight different User Priorities (UPs), as given in Table 2.1. On average, a STA with high priority traffic waits a little less before it transmits its packet than one with low priority traffic and on average, sends more traffic over the medium. In this section, a discussion of the EDCA Reference Implementation specified in the standard [48] follows.

User Priority	Access Category (AC)	Designation
1	AC_BK	Background
2	AC_BK	Background
0	AC_BE	Best Effort
3	AC_BE	Best Effort
4	AC_VI	Video
5	AC_VI	Video
6	AC_VO	Voice
7	AC_VO	Voice

Table 2.1: IEEE QoS Traffic User Priorities and Access Categories [48]

The EDCA specifies four Access Categories (ACs), as shown in Table 2.1, that provide support for the delivery of the traffic with UPs at the STAs. These ACs are derived from the UPs.

A model of the reference implementation is shown in Figure 2.3 and it illustrates the mapping from frame type or User Priority (UP) to AC (the four transmit queues and the four independent enhanced distributed channel access functions (EDCAFs), one for each queue).

For each AC, an enhanced variant of the DCF, called the enhanced distributed channel access function (EDCAF), contends for Transmission Opportunities (TXOPs) using a set of EDCA parameters. Each priority level is assigned a TXOP, which is a time interval when a particular STA has the right to initiate frame exchange sequences onto the WM. This TXOP is defined by a starting time and a maximum duration. It is either obtained by the STA by successfully contending for the channel or assigned by the Hybrid Coordinator (HC), which is also the AP. The STA can send as many packets as possible, as long as the duration of the packet transmissions does not extend beyond the maximum TXOP duration. If a packet is too large to be transmitted in a single TXOP, it is fragmented into smaller packets. A TXOP time interval of 0 means that the QoS STA is limited to a single MAC service data

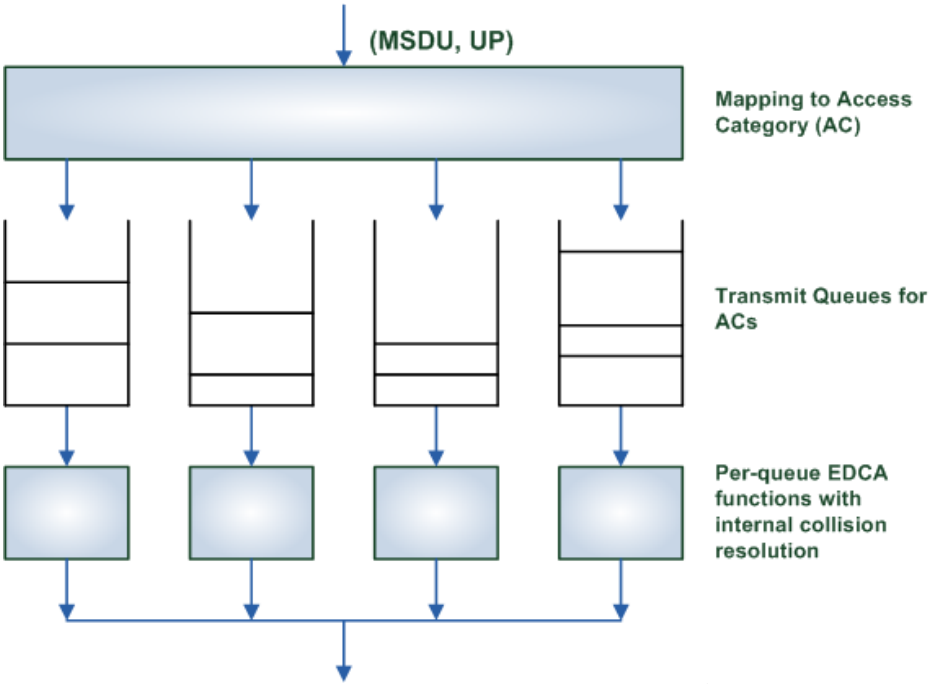

Figure 2.3: Reference Implementation Model for EDCA [48], where MSDU stands for MAC Service Data Unit

unit or MAC management protocol data unit. TXOP ensures that STAs with low priority traffic do not acquire the channel for long periods of time as low priority traffic have a TXOP of 0. A protective mechanism (such as transmitting using HCCA or the RTS/CTS scheme, should be used to reduce the probability of other STAs transmitting during the TXOP.

Each AC is allocated an Arbitration InterFrame Space Number (AIFSN). This value is used to calculate the minimum specified idle duration time, called the AIFS, which is a function of the AIFSN, Slot Time, and SIFS Time. Therefore, an AC has to listen to the channel idle for an AIFS interval before it can begin contending for the channel. It is not fixed, as with the Distributed InterFrame Space (DIFS) in DCF, but is dependent on the ACs. The higher priority ACs have a smaller AIFSN, which means that, on average, they listen to the channel idle for a shorter interval, resulting in favourable channel access.

The contention window limits, CWmin and CWmax, from which the random back-off is computed, are not fixed per physical layer (PHY), as with DCF, but are variable. Each AC has its own limits, with higher priority ACs having smaller limits. They are assigned by a management entity or an AP. The back-off procedure has been updated to cater for these ACs. Each AC in a STA operates independently, and coordinates its own channel access, therefore, it works as if it is a standalone STA. Each AC maintains its own back-off counter and retry limits. If the channel is idle after backing off, it then transmits its packet(s), depending on the TXOP duration.

There are two types of collisions, virtual or internal collisions, and external collisions. Virtual collisions occur between contending EDCAFs within a STA are resolved within the particular STA, so that the data frames from the higher priority AC receives the TXOP, and the channel, and the data frames from the lower priority colliding ACs behave as if there were an external collision on the WM.

External collisions are handled in the same manner as with the DCF, where each AC involved in a collision

doubles its CW, increments its retry counters, generates a new back-off counter, and then backs-off. If the retry limit is reached, the resulting data packet is discarded, and all respective counters reset.

Figure 2.4 shows the InterFrame Space (IFS) relationships in the 802.11 protocol, specifically focussing on the AIFS of the EDCA protocol.

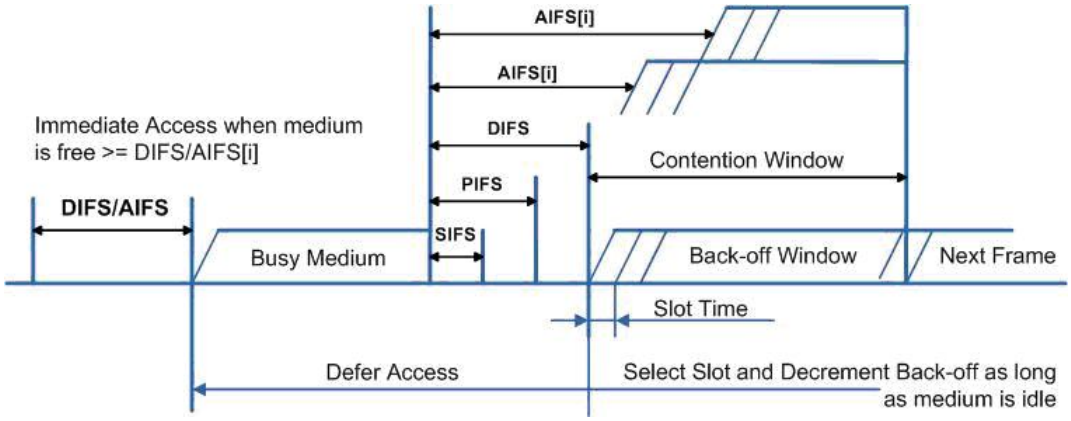

Figure 2.4: Basic Access: Illustration of the placement of InterFrame Spaces for EDCA, adapted from the IEEE 802.11 Standard [48]

The EDCA mechanism can be used in both the basic access and RTS/CTS schemes, which were explained in Section 2.2.

A recipient can either send an ACK frame for every successful packet transmission, for every block of packets (Block ACK), or no ACK at all.

The Block ACK mechanism improves channel efficiency by aggregating several acknowledgments into one frame. Two types of Block ACK mechanisms are specified: immediate and delayed. Immediate Block ACK is suitable for high-bandwidth, low-latency traffic while the delayed Block ACK is suitable for applications that tolerate moderate latency.

The usage of the No ACK mechanism is determined by the policy at the STA. When the No ACK policy is used, MAC-level recovery is absent, resulting in significantly reduced reliability of the traffic relative to the reliability of traffic with other ACK policies. This is due to the increased probability of lost frames from interference, collisions, or time-varying channel parameters.

A Wi-Fi network may make use of Admission Control to administer policy and/or regulate the available network bandwidth resources. Admission control is also required when a STA desires guarantee on the amount of time that it can use the channel. The AP controls admission control in the network. There are two distinct admission control mechanisms: one for contention-based access and another for controlled channel access.

Admission control normally depends on the vendors implementation of the scheduler, available channel capacity, link conditions, retransmission limits, and the scheduling requirements of a given stream [48]. All these criteria affect the available resources of a particular stream.

2.4 CHANNEL ACCESS

Network performance of any communication system mostly depends on the utilised medium. This medium, whether optical fibre or hard disk drive or a wireless link, is referred to as a communication channel. Communication channels are divided into two groups, wired channels, which have a physical connection between transmitter and receiver, and wireless links, which do not have a solid link.

2.4.1 Overview

There is a very large difference between the two channels mentioned above. Wired channels can be very robust. Wireless channels, on the other hand, can be very unreliable. In wireless channels the state of the channel may randomly change within a very short space of time. This random and drastic behaviour makes communication over such channels a very complex task.

Wireless communications channels suffer from fading and interference and noise. These are factors that will always be present, especially in unlicensed frequency bands, and therefore they affect network throughput.

As networks evolve, the design of communication protocols increases in complexity and effort required. There is need for evaluation of such networks, as this provides insight into techniques and possible optimisations required to improve overall network performance and efficient use of resources.

Wireless communications channels are considered to comprise propagation channels, radio channels, modulation channels, or digital channels [1].

A propagation channel lies between the transmitter and receiver antennas, and it is influenced by the phenomena that influence the propagation of electromagnetic waves. The transmitted signal is affected by attenuation, reflection, etc; therefore, this constituent of a channel has a multiplicative effect on the transmitted signal.

The radio channel consists of a propagation channel and both the transmitter and receiver antennas. When the antennas are considered to be linear, bilateral, and passive, the channel is also linear and reciprocal. The transmitted signal is only affected by attenuation but, unlike propagation channels where the attenuation might be modified by the used antennas, the antenna influence is strictly linear. The signal is the same as with the propagation sub-channels, but it might also be scaled by the use of antennas.

A modulation channel consists of a radio channel plus all system components involved from the output of the modulator on the transmitter side to the input of the demodulator on the receiver side, such as amplifiers. Due to the process of amplifying the received signals, some additive effects that distort the signal come into play, such as noise and interference.

A digital channel consists of the modulation channel plus the modulator and demodulator. This channel is non-linear and non-reciprocal. The input to this channel is bits, which might come from packets. The bits are then grouped into analogue representations, called symbols, which are then transmitted. The bits are often grouped along with some check bits, which are used to calculate whether the signal has been transmitted accurately.

In this work, we focus on Orthogonal Frequency Division Multiplexing (OFDM) digital modulation, which is used by the 802.11g and 802.11e standards at a 54Mbps data rate [48], as well as fading and noise.

2.4.2 **OFDM Transmission**

In digital communications, there is an increasing demand for high-speed, high quality digital mobile and portable transmission and reception of signals. This means that the receiver often has to deal with a signal that is weaker than desirable and contains many echoes. Simple digital systems do not work well in these multi-path environments, and therefore raises the need for a better digital system.

OFDM is a digital multi-carrier modulation method in which a signal is split into several narrowband channels at different frequencies [75]. Data is divided among a large number of closely spaced orthogonal sub-carriers. Each sub-carrier is modulated with a conventional modulation scheme, such as quadrature amplitude modulation or phase shift keying, at a low symbol or bit rate, maintaining total data rates similar to conventional single-carrier modulation schemes in the same bandwidth.

The entire bandwidth of the system is filled from a single source of data, and the data is transferred in parallel [52] [75]. Since only a small amount of the data is carried on each sub-carrier, and there are lower bit rates per carrier, the influence of inter-symbol interference is significantly reduced [75].

In 802.11g/e at 54Mbps, OFDM with 52 sinusoidal sub-carriers, with frequency separation of 1/T are used, where T is the symbol period. A detailed explanation of OFDM is given is Section 4.4.1

2.4.3 **Fading**

When multi-path versions of the same signal interfere with one another, there will be fluctuations of amplitudes and phases in the received signal. This phenomenon, and its effect on the received signal, is known as small scale, multi-path fading [52].

There are several factors that influence fading in radio channels:

- Multi-path propagation
- Speed of the mobile device

Multi-path propagation is the phenomenon that results in radio signals reaching the receiving antenna by at least two or more paths, mainly due to refraction of the signals, or reflection from water bodies and terrestrial objects, such as mountains or buildings. The speed of a mobile device can cause Doppler shifts on the components of the multi-path signals

The speed of obstructions in the environment may result in the obstruction of the line-of-sight, resulting in the signals reflected of these objects. The reflected signals will have distortions due to Doppler shifts.

The effects of fading can be characterised by multi-path time delay spread or Doppler spread. Multi-path time delay spread determines whether a channel experiences flat or frequency selective fading, and Doppler spread determines whether a channel experiences slow or fast fading. Flat fading occurs when all of the spectral components of the signal are equally affected by fading. The coherence bandwidth of a channel defines the bandwidth over which a channel will experience flat fading. If the bandwidth of a transmission is less than the coherence bandwidth, the channel experiences flat fading otherwise, it experiences frequency-selective fading. Frequency-selective fading is more difficult to model, as it does not affect all parts of the transmission equally. Flat fading channels are often

referred to as narrowband channels, and frequency-selective fading channels are known as wideband channels.

The rate at which fading fluctuates determines whether or not the channel experience fast or slow fading. If the bit period is greater than the average period of a fading signal, the channel is fast fading. Otherwise, it is slow fading.

In OFDM the total channel is a frequency selective channel, however, the channel experienced by each subcarrier in an OFDM system is a flat fading channel.

Flat fading channels are also known as amplitude varying channels [34], and the distribution of the variation of the amplitudes in these channels can be modelled with one of two statistical distributions, depending on the environment. If there is no dominant propagation along a line-of-sight between transmitter and receiver, and all reflected signals are received with similar signal strengths, then the channel experiences Rayleigh fading, and the envelope of a single multi-path component can be characterised by a Rayleigh distribution [34]. If the dominant propagation results in a signal much stronger that the others along a line-of-sight, then the fading is known as Rician fading, and the amplitude gain is characterized by a Rician distribution [34].

Rayleigh fading is mainly experienced in heavily built-up urban environments; hence, it is the focus of this work.

A detailed discussion is found in Section 4.4.2

2.4.4 **Interference and Noise**

Interference is anything which alters, modifies, or disrupts a signal as it travels along a channel from transmitter to receiver. In general, it refers to the addition of unwanted signals to a useful signal [75].

Noise is defined as an unwanted perturbation to a wanted signal.

Noise is always present, and emanates from several sources, such as atmospheric disturbances, electronic circuitry, machinery. It is stochastic in nature and varies with time. Noise can be sub-grouped into three groups; thermal noise, filtered white Gaussian noise, and Man-made noise.

Thermal noise is a result of the movement of charged particles inside electronic components existent in every receiver system and is therefore unavoidable. Filtered white Gaussian noise normally occurs in limited-bandwidth networks, where the received signal no longer has all frequencies in the same proportions. Man-made noise comes from machines operated by humans, and it is considered noise because it is energy radiated by machines which are not supposed to generate that energy.

If at least two interference sources are active or exist, interference may be modelled as a white Gaussian process, adopting many of the characteristics of noise. Therefore, in this work, we focus on Additive White Gaussian Noise (AWGN).

AWGN is one of the most common noise models in communication systems. The term White refers to the frequency spectrum, which is deemed continuous and uniform within the frequency band of interest. This term originated from the fact that white light contains equal energy over the visible frequency band. The thermal noise at the receiver amplifier shows uniform noise power per unit bandwidth at all frequencies.

In communication systems, many different noise sources exist, even though we represent them as one noise source in the noise model. This noise from different sources adds up and the summation of them has a Gaussian distribution. The autocorrelation of the random signal is a delta function in time domain. The sufficient condition for a random process to have a delta function as the autocorrelation is that the random variables at different times are independent and have zero means.

A detailed discussion is found in Section 4.4.3

2.5 WORKLOAD

The performance of a network largely depends on the traffic being serviced. Thus, the performance evaluation of a network requires the use of representative workloads in order to produce dependable results. This is achieved through collecting data about real workloads and then creating statistical workload models that capture their salient features.

There is need to develop an aggregated model that represents all traffic types. The 802.11e QoS enhancements classify traffic into 8 user priorities, resulting in four access categories; voice, video, best effort, and background traffic [48].

2.5.1 VoIP Traffic

The characteristics of voice traffic require special attention on networks that were designed with data traffic in mind [33]. The traffic consists of very short frames that require regular delivery. There are networks that were specifically designed to deal with transmission of voice traffic, such as the telephone network, which offers minimal delay and has tightly controlled timing.

Transmitting voice traffic on a data network is challenging because characteristics of data networks do not guarantee good VoIP performance. Packets must always be transmitted almost immediately upon receipt. The best way to ensure that a packet can be transmitted immediately is to keep the transmit queues relatively empty so that waiting frames can be transmitted immediately. Good data throughput, however, depends on keeping transmit queues near maximum length so that long packet chains can be built. The best way to compromise is to build protocols that treat packets from different applications with appropriate priority. Although prioritizing voice packets appropriately is the biggest hurdle for wireless LAN administrators, there are several other challenges as well. Users will expect that telephone calls will be smoothly handed off across access points, and that telephone calls receive the necessary share of limited network capacity of 802.11 [33].

The most popular 802.11 standard to date, the 802.11g DCF standard, does not differentiate traffic, therefore, voice traffic is treated as any other data packet, and traffic is sent on a first-come-first-serve basis. It is interesting to see how this standard handles voice traffic compared to the QoS-specific 802.11e EDCA standard. Voice traffic has the highest priority in 802.11, so one would expect its transmission to be efficient.

2.5.2 Video Traffic

Video frames are larger, and they typically span multiple 802.11 packets. Packet headers may contain information that is critical for keeping the traffic synchronised [58]. Video quality suffers a lot from packet loss. If one packet is lost, this will affect other correctly received packets [58].

Video is a problem for wireless networks because of its highly variable bit rate; complex video scenes tend to use higher bitrates. Inter-frame data dependency also means that overall quality suffers if errors occur, and some frames are more important than others. These problems are compounded in wireless technologies because such networks suffer from interference, path loss, and fading, there are a limited number of channels in unlicensed frequency bands, which often overlap, and channel characteristics are dynamic.

In this work it is important to see how video traffic is handled between the DCF (undifferentiated) and the QoS-specific EDCA.

2.5.3 Best-Effort Traffic

Best effort traffic can be classified as all other kinds of traffic that are not deemed sensitive to QoS metrics, such as jitter, packet loss, and latency. Peer-to-Peer (P2P) traffic and Email are typical examples.

In this work, emphasis is put on P2P traffic. P2P file sharing applications are often designed to use any and all available bandwidth, which impacts QoS-sensitive applications, like voice and video.

P2P file sharing applications have gained tremendous popularity [57]. More objects are continuously being made available, enticing even more users to join.

802.11 WLANs have less bandwidth compared to wired technologies (typically 54Mbps for 802.11g and 802.11g), and the use of P2P applications with these networks results in higher network latency for the users.

2.5.4 Background Traffic

As people use the internet daily, they log into domains, send instant messages, and remotely administer networking equipment [73]. All this data and more, goes through various network checkpoints while en route to its destination. One of the checkpoints likely to be encountered is a Network Intrusion Prevention System (IPS), such as a firewall, since a network IPS sits inline in corporate networks inspecting traffic for various attacks aimed at remotely exploitable vulnerabilities.

It is important to model this traffic when evaluating network performance because of two reasons:

- Background traffic consumes network resources
- Not including realistic background traffic does not reflect what is seen in the real world

In this work, we are mainly interested in HTTP traffic, since this type of traffic constitutes a large amount of internet traffic. It would make sense to include this type of traffic in the discussions for completeness.

2.6 Related Work on Modelling

Performance evaluation of the 802.11 standard has been done numerous times using all three modelling approaches; Analytic, Simulation, and Hardware Experiments.

Analytical models are mathematical models where the solution to the equations used to describe changes in a system can be expressed as a mathematical analytic function. Simulation models are software models that mimic

actual behaviour of a system which they represent. Hardware experimentation constitutes an actual representation of the system being modelled, albeit mostly, but not always, in a controlled environment.

From analysis of the three modelling approaches, it was discovered that analytic modelling is less complicated, in terms of effort required to configure the model, cost, and difficulty, than simulation and hardware test beds in terms of the abstractions and simplifying assumptions made, with hardware test beds being the most complicated of the three. Analytic modelling is usually based on stochastic theory, including Markov theory, whereas simulation requires more than just calculations. It requires features that mimic the actual system and also keeping track of the current state of the system. Hardware test beds consist of radio nodes that create an actual network environment. One also needs to model the underlying physical layer for both simulation and analytic models. Analytic modelling has more simplifying assumptions compared to simulation modelling, which in turn, also makes more simplifying assumptions compared to hardware experimentation. Analytic models are mainly stochastic. Simulation models will require more computer memory, models of Gaussian and Rayleigh fading channels, and normally a great deal of processor time. Hardware test beds require an expenditure on actual hardware components, in particular, the Network Interface Cards. They also take time to set-up, and they usually require a large area with low interference set up.

Analytic models execute faster compared to simulation because, in most cases, they are realised as a set of closed mathematical formulae. Simulation models, on the other hand, are slower because they tend to mimic the operation of the actual system step by step through time, going through iterations. Hardware test beds operate much like an actual network.

The number of assumptions made for analytic models make the results less accurate compared to those of simulation. Packet inter-arrival times are usually generated using statistical distributions such as the Poisson, Geometric and Exponential distributions and these do not accurately depict actual network traffic inter-arrival rates. Simulators, on the other hand, tend to accurately mimic the performance of an actual network. Packet inter-arrival rates and packet sizes are mostly derived from actual network traces. These are obtained by measuring traffic from an actual network for, say, 24 hours, and deriving the best distributions for the network workload model. Hardware test beds are the most accurate of the three modelling approaches. They make the least assumptions and they implement the PHY and MAC layers accurately, although their operation environment is often, for practical reasons, not realistic.

Both analytic and simulation models scale well, with analytic models normally being better since, depending on the theory, memory and processor speed requirements are fewer than in the case of simulations. Hardware test beds, on the other hand, are too expensive to have a lot of nodes in the network and they take up a lot of space. Currently, there are few hardware test beds, and they normally range from 25-50 nodes.

After careful consideration of the three modelling approaches, we decided on simulation for our purpose since it better represents actual systems than analytic models, and is much cheaper than hardware experimentation.

The end-product of this work is an extensible and integrated simulation model that has DCF, EDCA, Channel Model and Workload Model. In this section, a discussion of previous work on these four components is presented. Each component is discussed separately. The author found it important to tackle these components separately, so that full emphasis can be put on each and then salient features are selected from each component discussion. An overview of the integrated model is covered in Section 3.

The first discussion covers the DCF, focussing simulation. The same approach is adopted in the next discussion for the EDCA. The third discussion touches on channel modelling, focussing on OFDM modulation, channel fading, and noise and interference. The last section covers the workload modelling, and it discusses the four types of traffic mentioned earlier in Section 2.5; voice, video, best effort, and background.

2.6.1 **DCF Modelling**

Performance evaluation of the 802.11 DCF standard has been done numerous times using all three modelling approaches; Analytic, Simulation, and Hardware Experiments. In this section, various performance models, described in the literature, are discussed for simulation since it is the focus of this work.

There are three types of simulation paradigms, namely

- Discrete-Event Simulation,
- Process Interaction, and
- Activity Scanning.

We have seen the development of simulation environments specifically designed to model communications networks. Three models are discussed next, with each DCF simulation using a different simulation paradigm.

Weinmiller *et al.* [74] developed a process-oriented simulation model using PLOTEMY [23], an object-oriented simulation tool. The system behaviour in a process interaction approach is described by a collection of almost-parallel executing programs, called processes or objects.

Cocorada [11] used the OMNeT++ [46] simulation environment to model the 802.11g standard using the discrete-event simulation paradigm. Events represent changes in the system state and are used in this model together with a global time and an event scheduler, which identifies the next event to be activated and executes the necessary routines.

Chen *et al.* [8] implemented an activity scanning model that modifies the NS-2 [45] 802.11 model. In activity scanning, activity routines are cyclically entered, and the next activity starts at the terminating time of the current activity and all the other entities in the system are updated at this time.

All simulation models [74] [11] [8] mentioned above consider non-saturated conditions, variable packet lengths and implement blocking of queues, that is, limited packet buffers at each STA. In [74], the distribution of packet sizes and associated parameters were determined from a trace that contained arrival times and corresponding packet sizes for Ethernet traffic. This approach was taken because it has been found that local and wide area traffic is not well modelled using the Poisson process [49].

All reported simulation models also model physical layer properties.

Li *et al.* [37] considered the medium access control (MAC) for very high speed wireless LANs, which is designed to support rich multimedia applications such as high definition television. The legacy MAC layer greatly restricts performance improvement due to its large overhead, and it is estimated that it utilizes less that 20

A comparison of the simulations results does not give true insight into the models relative accuracy since the authors made different considerations, such as packet inter-arrival rates, and different number of STAs.

2.6.2 EDCA Modelling

In this section, we discuss previous work done on the IEEE 802.11 QoS protocol, looking at simulation.

He *et al.* [19] developed a simulation model for the IEEE Enhanced Distributed Coordination Function (EDCF) using the NS-2 [45] simulation toolkit. They used their model to prove the effectiveness of MAC service differentiation provided by the EDCF, compared to the Distributed Coordination Function (DCF), which works in non-QoS configured networks. They implemented their model with four AC queues, and used the exact values of the QoS parameters explicitly specified in the standard.

Blasi *et al.* [6] proposed a modification of the EDCA, called Extended EDCA (E2DCA) [6] to improve the utilization and effective usage of channel capacity at high network loads. They developed a simulation model using NS-2 [45], and they modified the EDCA by not assigning Transmission Opportunity) (TXOP) intervals using the AP, but by providing a systematic way in which QoS STAs choose TXOP after acquiring the wireless channel and gained the right to transmit. This results in a distributed mechanism of allocating channel bandwidth.

Choi *et al.* [10] evaluated the EDCF against legacy DCF, using simulation. The aim of the exercise was to show that the EDCF can provide differentiated channel access among different priority traffic. Along with these two, they also evaluated the performance of the optional contention-free burst (CFB) [48], which allows a STA to transmit multiple MPDUs with the SIFS gaps, within the time bound of the TXOP limit.

Through simulation study, they concluded that the EDCF can provide differentiated channel access for different traffic types, and furthermore, the CFB enhances EDCF performance by increasing the overall system throughput and achieving more acceptable streaming quality in terms of frame losses and delays. However, the CFB has a higher cost of delay for certain traffic types, such as voice traffic.

Thottan *et al.* [70] evaluated the impact of EDCA on TCP application traffic consisting of both long and short-lived TCP flows [70], using the NS-2 [45] simulation toolkit. Based on simulations, they discovered that the performance of TCP applications is very dependent on the settings of the EDCA parameters and buffer lengths at the AP. They also discovered that for optimal QoS assurance for different traffic classes, it is necessary to jointly optimize the AIFS interval, CW interval, the queue sizes of different ACs, and the admission control policy.

Different ACs should be kept in mind when allocating the AIFS and CW interval. The optimal allocation of queue sizes to the different ACs considers the limitations imposed by the system resources. The choice of admission control policy can play a role in reducing the response times for high-priority flows while keeping small queue buffers to minimize delay.

Manitpomsut *et al.* [43] conducted simulations and presented results that showed the behaviours of WLANs that co-exist in a restricted area, and which utilize the same channel. They considered networks which operate in DCF, PCF, and EDCA modes in order to support QoS-sensitive traffic, such as VOIP.

Through simulation, they showed that there are problems with all the three access mechanism in delivering good QoS to sensitive applications. PCF, though, can support more voice streams, thus making it suitable for use with delay-sensitive applications, but it still introduces an undesirable behaviour under these conditions. With heavy

traffic load, PCF WLANs can make both the DCF and EDCA WLANs suffer from very large delays. Even with a light traffic load, the interference between two PCF WLANs can result in unsatisfactory QoS support. EDCA WLANs can provide QoS for voice traffic when it is only one network, but with multiple EDCA networks sharing the same channel, data traffic will suffer from delay and high packet loss.

Siris *et al.* [64] proposed a similar model to [65] for efficient resource control of elastic traffic over the HCCA and EDCA mechanisms. This is an extension to the work done in [65], and the extensions involve models that consider the TXOP parameter for controlling resource sharing in both EDCA and HCCA mechanisms in multi-rate 802.11e systems.

These models can also be used to derive congestion prices, which can be transmitted to the wireless STAs through some feedback mechanism, thus providing them with appropriate incentives to guide the system to its optimal operating point.

The models assume knowledge of the utilities of wireless STAs. An additional centralised entity located at the AP is responsible for optimally selecting the transmission probabilities, TXOPs, and the percentage of time spent in the contention and contention-free periods. These parameters are then broadcast to the wireless STAs, as defined in the IEEE 802.11e standard [48].

Gu *et al.* [17] modelled the 802.11 EDCA based on the IEEE 802.11e specification [48] and modelled it using the Opnet simulation tool [15], with four STAs. Experimental results showed that the EDCA works well for differentiated data services.

2.6.3 **Channel Modelling**

Konrad *et al.* [32] presented a novel algorithm for modelling network channels that experience time varying error statistics. They used a Markov-based Trace Analysis (MTA) algorithm to separate a non-stationary network trace into stationary traces and to accurately model the traces using Discrete-Time Markov Chains (DTMC). The underlying concept behind the MTA algorithm is the assumption that a trace with non-stationary properties can be decomposed into asset of piecewise stationary traces consisting of two states: lossy and error-free. Thus, the algorithm defines these states, and parameterizes transitions between them as a function of a preset parameter, known as the change-of-state constant.

The error-free states contain only the successfully transmitted frames, while the lossy states exhibit stationarity [32]. The sequence of lossy states can be modelled by a traditional DTMC. Stationarity, in this instant, means that the error statistics remain relatively constant over time.

The MTA algorithm views a given trace as a process with two states: lossy and error-free. The algorithm divides the entire trace into a lossy trace, which consists of a concatenation of lossy states, and an error-free trace, which consists of a concatenation of error-free states. Two random processes with a discrete space E = 0, 1, 2, ... are defined:

- Lossy state length process [B_n for n greater or equal to 0], where B_n represents the number of elements in the nth lossy state (i.e. the length of the state) [32]
- Error-free length process [G_n for n greater or equal to 0], where G_n represents the length of the nth error-free state [32]

The distributions of both B_n and G_n are obtained by plotting the cumulative density function (CDF) and then finding the best fitting distributions. The error-free trace is deterministic, where all values are set to zero. The lossy trace, on the other hand, is a stationary random process, and it can be modelled by a DTMC with a certain memory. The MTA algorithm therefore calculates the memory of the lossy state and determines its transition probabilities.

The performance of the MTA algorithm was compared to two Markov chain-based models, the traditional Gilbert model of order one (i.e. with two states), and a 3rd order Markov model (i.e. with eight states), and to collected traces from a GSM wireless digital cellular network. The Gilbert model predicts the state of the next frame by just looking at the previous received frame. The 3rd order Markov model keeps track of the previous three frames, increasing its prediction accuracy at the cost of additional complexity in the Markov chain.

Comparisons of results showed that the MTA algorithm traces are closer to the real distributions of collected traces than the distribution of traces generated from the Gilbert and 3rd order Markov models.

McDougall *et al.* [44] investigated the appropriateness of wireless network simulations that employ a two-state Markov model to approximate a flat Rayleigh fading channel. In a Rayleigh fading channel, each realisation represents the fading encountered throughout the entire duration of a fixed length frame. Received frames are demodulated using a minimum distance detector, and the results from the received frames are recorded as a frame error process. The frame error process is then analysed for frame error rate (FER), average burst error length (ABEL), and variance in burst error length (VBEL).

The ABEL and FER results from the Rayleigh fading channel are used to generate a two-state Markov model channel approximation. The Markov model is defined by a probability, p, the probability of successfully transmitting a frame given the previous frame was successfully transmitted, and probability 1-q, the probability of successfully transmitting a frame given the previous frame was in error. The Markov model was designed to almost exactly replicate the FER and ABEL of the 802.11b protocol.

A two-node network was employed in NS-2 [45], with one node sending application data fames of 512 bytes fixed length to the other node through an 802.1b link. The NS-2 [45] simulation conducted employed the DCF at the MAC layer and used a queue length of 50 frames.

Another model, a two-state run-length model (RLM), was developed in order to better the frame error statistic VBEL. This model is similar to the Markov model, the RLM also has two states, good and bad, however, the channel probabilities are fixed, and the duration of a state is variable. The RLM was able to approximate the FER, ABEL, and VBEL accurately.

Ji *et al.* [28] compared four different modelling approaches towards modelling frame-level errors in GSM channels. They compared the MTA model [32], a k-th order Markov model, a hidden Markov model (HMM), and a new On/Off model. The On/Off model consists of two states, good and bad. The states of the On/Off model are characterised by the mixtures of geometric phases. From the analysis of results, the On/Off model most accurately captures the FER of the GSM trace.

Chen *et al.* [9] presented a stochastic process called the punctured autoregressive (AR) process, which is used to model both variable bit rate (VBR) video traffic and wireless channel dynamics. They used a punctured AR process modulated by a doubly Markov process to model the VBR video traffic. The doubly Markov process models the state of each video frame while the AR process describes the number of bits for each frame at each state.

This process considers the timing information between frames of the same state and thus gives better modelling performance. The model also captures both long range dependency and short range dependency characteristics of the video traffic.

Malone *et al.* [42] presented a novel technique for accurate estimation of the proportions of packet losses arising from collisions and from other sources of loss (channel noise, hidden nodes, etc).

They made two main assumptions in their model:

- The probability that at least another node transmits in an arbitrary slot does not depend on whether STA 1 transmits or not.
- The collision probability of node 1 is independent of the back-off stage of station 1.

Experiments were conducted in NS-2 [45] using 802.11b parameters and STAs transmit packets of length 1500 bytes.

Lee *et al.* [36] proposed an adaptive unequal error protection (UEP) and packet size assignment scheme for scalable video transmission over a burst error channel analytic model. A two-state Markov model or a Gilbert model is used to simulate burst loss patterns over a wired or wireless channel. The two states are error-free and error. The model is descried by an average bit error rate and average burst bit error length. Simulation results show that the proposed model can effectively adapt to various channels and react to poor channel conditions, with a smaller and smoother drop in quality. This proposed model does not rely on any specific network configurations, therefore, can be employed on any packet-oriented network.

2.6.4 **Workload Modelling**

As mentioned in Section 2.5, the performance of a wireless network, or any system for that matter, is only relevant in its response to the workload imposed upon it.

There is need to develop an aggregated model that represents all type of traffic. The 802.11e QoS enhancements classify traffic into 8 user priorities, resulting in four access categories; voice, video, best effort, and background traffic. This section discusses previous work done on modelling the behaviour of these four types of traffic.

The discussion spreads across different communications network systems, since the fundamental concept is the behaviour of traffic, regardless of the type of communication network.

Voice

Thus section discusses several VoIP network workloads that have been used in modelling communications networks.

Seger [60] proposed a theoretical approach to model an aggregated traffic flow of VoIP connections on a packet basis. The approach was to be able to create a VoIP traffic stream with the knowledge of tele-traffic theory and the traffic mostly given by statistical estimations or measurements. In the work, an examination of different voice codecs respecting packet flow was performed. The problem of packet flow modulation was divided into three sub-models:

1. Model for CBR voice over IP streams (Simple VoIP Traffic),
2. Model for talk-spurts and silence gaps (Voice Activity Detection (VAD)), and
3. Model for traffic circuits on a link (Aggregated Flow on Real Links)

Simple VoIP Traffic

Packet streams in VoIP connections are almost always coded in CBRs, therefore packet intervals and packet sizes will be constant. However, different codecs produce different intervals and packet sizes, even if the effective bandwidth is the same.

An example is a PCM-codec (G.711) which produces a 64Kbit stream. Each packet may contain at least 8 Bits, which is the amount of bits for one sample, and must then be sent every 125s to get a bit stream of 64Kbit/s.

VAD

More recent voice codecs like G.729B, G.723.1A and NetVoT have the ability to detect talk-spurts and silence gaps within a conversation [72] [59]. In order to model such behaviour, traditional approaches, such as assuming that the length of spurts and gaps follow an exponential distribution, are applied. It has been proved, however, that this approach does not work anymore [29], because, in most cases, the spurt is slightly more heavy-tailed than exponential, whereas the gap distribution deviates strongly from an exponential model.

Better fits can be used [3], such as the Gamma distribution, which fits better for the spurt (ON-) period, and the Weibull distribution, which fits better for the gap (OFF-) period.

Aggregated Flows on Real Links

The tele-traffic theory seemed to be the most promising approach because the theory of calculating tele-traffic circuits is well known and could easily be adjusted in the approach. Most telephone equipment integrates measurement tools, such as investigation of average hold time and average inter-arrival time.

There are two sub-models within the tele-traffic model:

- Inter-arrival time model
- Call hold time model

The tele-traffic theory was used to determine the call inter-arrival time and the call hold time for a link [60]. The main difference between this packet based approach within QoS-enabled diffServ networks [60] and a circuit switched network is the non blocking character of a link. There is no natural detection for the caller that there are no resources left on a link, because the bandwidth will be divided among all connection flows.

Discrete event simulation was used to simulate the raised calls on a link [60], which means that the change of state of process is determined via calculation of the next arrival or departure time.

These three sub-models were combined to produce an aggregated packet flow model. The process starts with the calculation of the next arrival time of a packet. When the time is reached, several things are triggered immediately:

- Tele-traffic
  - The call holding time is determined due to an exponential distribution.
  - The time until next call (inter arrival time, TA) is determined due to an exponential or hypo-exponential distribution (depending on external influences)
- Constant bit rate packets
  - The determination of the codec, due to the used codec application probability
  - The start of packet generation with interval TI
- Spurts and gaps
  - The spurt time is determined due to the Gamma distribution
  - The gap time is determined due to the Weibull distribution
  - Every new determined spurt and gap is concatenated to a voice stream until the call ends (due to the call hold time)

Shah-Heydari *et al.* [63] presented a study on the use of Markov-Modulated Poisson Processes (MMPP) in modelling aggregate ATM traffic for queuing performance evaluation. MMPP is a double stochastic Poisson process in which a Markov chain determines the transitions between phases. A Poisson process is used to determine the number of arrivals in a time frame at each phase.

Heffes *et al.* [20] used a two-state MMPP to model aggregate traffic from voice sources. The importance of the MMPP model is that MMPP is capable of catching inter-frame dependency between consecutive frames. This is indicated by the Index of Dispersion of Counts (IDC). IDC is defined as the variance to mean ratio for the number or arrivals fro increasing size of arrivals in non-overlapping blocks. The IDC value for a Poisson process is always 1 at every lag, while it increases with the time lag up to a limit for MMPP.

A packetized voice model can be represented by an on-off model with exponentially distributed sojourn times at each on and off states. It was shown that a superposition of N on-off voice sources can be modelled by a two-state MMPP [20]. A number of methods were proposed for matching the four parameters of a two-state MMPP [20] [31] [22] [2] [13], including mean arrival rates and mean sojourn times at two states, to the parameters of the aggregated traffic of on-off sources. For example, by considering the counting process of the number of arrivals in fixed-size time intervals, the parameter matching can be based on moments [20], overload-underload situations [31] [2], or a combination of IDC and overload-underload situations [22].

Moment based matching [20] uses the first moment, two points of the IDC curve, which represents the second moment, and one point of the third central moment of the superposition of N on-off sources to generate the four parameters of a equivalent two-state MMPP. The IDC curves of the aggregated on-off sources and the equivalent two-state MMPP model are well matched. This method, however, suffers from the lengthy calculation of the third central moment using inverse Laplace transforms.

The -matching technique is discussed in [31] [2], and the superposition of N on-off sources is modelled as a Markov chain of N + 1 states. This technique is divided into two parts in order to model it with a two-state MMPP. The transition rates of the MMPP model are determined by equating the mean sojourn time in each overload/underload state to that in all of the corresponding states in the Markov chain using a recursive equation [31].

This results in unmatched IDC curves of the aggregate traffic and the two-state MMPP model. Therefore, the derived two-state MMPP model does not effectively capture the inter-frame correlation of the aggregate traffic and, thus, cannot be used to predict the performance under heavy load.

The IDC matching technique, discussed in [22], uses the same overload-underload technique as [31] [2]. However, the sojourn times of the two-state MMPP model are calculated by matching the values of the IDC at large lags. This results in a simple calculation since the IDC values for both the on-off source and the two-state MMPP are known [20].

The performance evaluation of these techniques was done through simulation, using the OPNET network simulator, and using 100 voice sources and the same parameters as provided in [20].

Dang *et al.* [13] presented a fractal analysis study of VoIP traffic after an investigation of the characteristics of measured VoIP traffic on both call and packet level. The performance analysis of VoIP traffic includes the analysis of the appropriate queuing model. in dedicated voice systems, voice streams are served on a First-Come-First-Serve (FCFS) basis. However, if links are shared between both data and video streams, priority scheduling has to be applied.

Video

This section discusses previous work done in video traffic modelling.

Sarker [58] described two video traffic modelling techniques; VBR modelling techniques using Markov modulated fluid models, and Transform-Expand-Sample (TES).

Markov Modulated Fluid Models

Digital video sources, which are compressed using inter-frame variable rate coding are considered [58]. Each sources coded bit stream is stored in a separate pre-buffer, which assembles the data into blocks and packetizes the blocks.

The pre-buffer is used to eliminate complicated properties in the nature of the source model [18].

The packets from the all the pre-buffers are put into one common buffer in the multiplexer, where the packets are queued for transmission over a high-speed communication line as shown in Figure 2.5.

The data rates considered are on the order of Mbps, where the packet length is less than a kilobit. This makes it possible to ignore the discrete packet nature, and as a result, the sources can be modelled continuous bit streams at quantized data rate levels, with probabilistic transmissions between the various rate levels. It is also possible to model the statistical multiplexer queue as a fluid-flow pipe, which takes in bits from the various pre-buffers and serves them at a constant rate. The fluid-flow approximation is a powerful tool and it allows the use of the analytical models, taking into account the source correlations in the queuing analysis.

The experiment results indicated that an exponential correlation model for the data rate process is a very good approximation for video phone scenes with a uniform activity level, like showing a person talking.

Other types of video traffic, such as broadcast, television, video conferencing and long video phone sequences,

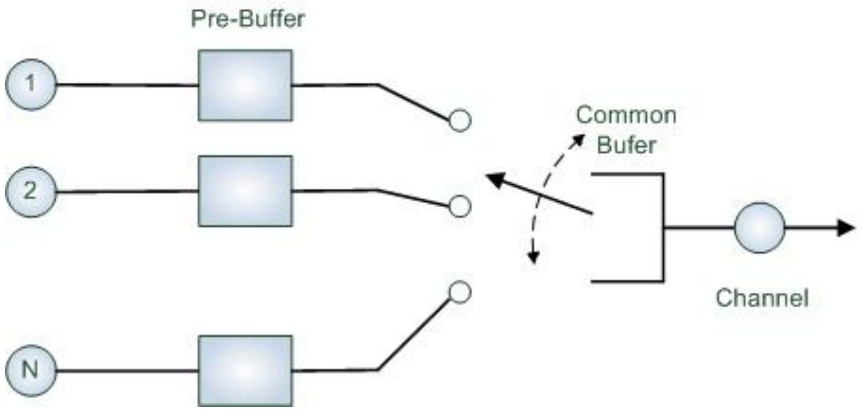

Figure 2.5: Statistical Multiplexer [18]

indicate the following structure. If an environment where the video sources feeding the network are a mix of these types is considered, then two important correlations are evident:

- a relatively fast decaying short term correlation corresponding to uniform activity levels, with a time constant of a few hundred millisecond, and
- a slow decaying long term correlation corresponding to sudden changes in the gross activity level of the scene, for example, when a scene changes in broadcast TV or a change

between listener and talker models in a video telephone conversation, with a time constant on the order of a few seconds

Maglaris [41] describes video modelling that captures only short-term correlation. The birth-death Markov chain, illustrated in Figure 2.6, is used for its simplicity in [41]. In this model, the bit rate, while it is in state i, is constant and is given be iA, where A is the quantization step size. The transition rate is chosen such that lower bit-rate states tend to jump to higher-bit-rate states and otherwise. In keeping with this, jumps are only allowed to neighbouring states in the birth-death Markov chain, so the model lacks the ability to capture abrupt changes in the arrival rate between frames.

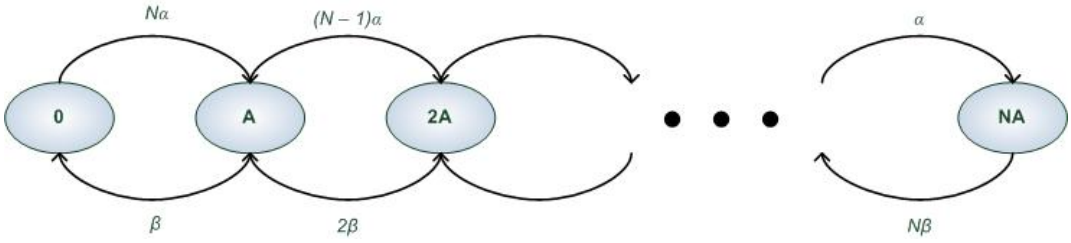

Figure 2.6: State transition diagram for birth-death Markov chain

The model was extended by Sen [61] to capture scene changes by allowing the rate to be integer multiples of two basic levels: high level A_h, and low level A_i. It uses a two-dimensional Markov chain in which the state is defined by two indices i and j, where 0 i M and 0 j N. While in state (i, j)), the flow rate is . Figure 2.7 illustrates such case of a single user.

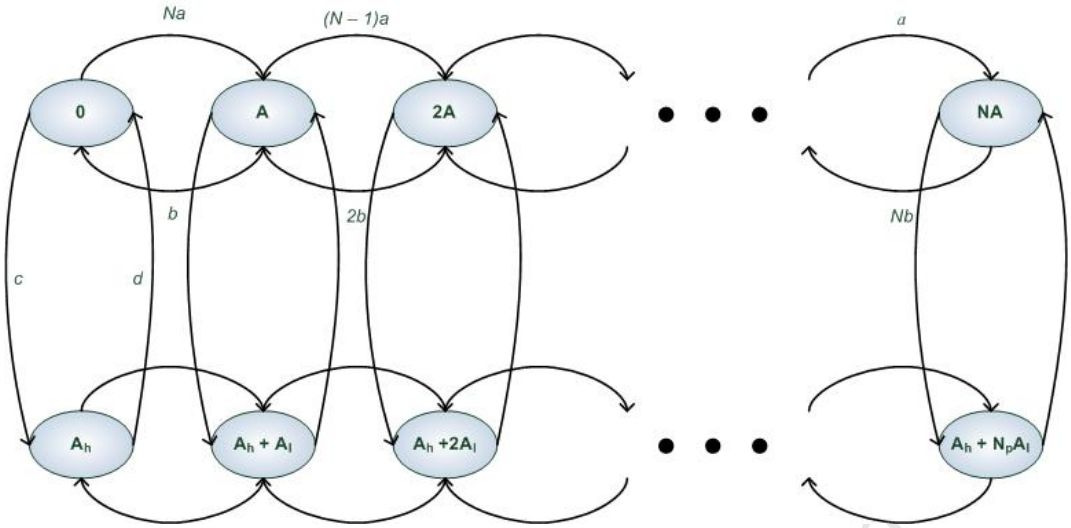

Figure 2.7: Fluid flow model for two levels of active VBR sources

This study considers the MPEG-1 syntax applied to a CCIR 601 (720x480 pixels/interlaced frame) video input VBR compression encoder, without the customary rate control algorithm specified in the MPEG reference model [21]. In MPEG, an input video sequence is divided into unites of Group-of-Picture (GOP) consisting of intra-coded (I) predictive (P) and bidirectional (B) elements.

An example of a GOP with parameters N = 9 and M = 3 is: IBBPBBPBB [58].

In its nominal or open loop operating mode, the VBR encoder uses a fixed set of quantizers, one for each of the mentioned frame types, I, B, and P, resulting in uniform image quality and variable bit rate. Each encoded frame of video is collected in over the next frame interval (33 ms). Thus, the inter-cell spacing associated with the VBR Asynchronous Transfer Mode (ATM) codec will very from frame to frame in a stochastic manner that depends upon scene content [58]. In addition to this, a particular VBR codec for ATM may have limited rate control in order to comply with call set-up parameters, such as the peak rate, long term average rate, peak continuous time.

The MPEG codec has a fixed structure that includes the encoding priority, and relative high open-loop peak-to-average ratio, mainly due to I frames. The periodic nature of the bit-rate traces causes the correlations and period to follow a similar trend and this periodic nature is difficult to capture, for example, with a single low order autoregressive (AR) model. However, the autoregressive structure can be captured using higher order linear autoregressive models using a combination of decaying exponentials. The marginal distribution of the MPEG VBR bit rate differs considerably from normal distributions, and thus AR models are cannot match the empirical marginal distributions accurately. TES modelling, on the other hand, can be used to obtain an accurate match for the marginal distribution and autocorrelation function simultaneously.

The model was constructed as a deterministic or stochastic superposition of three component source models, one each sequence of I, P, and B frames. The component bit-rates were then interleaved or superposed in the appropriate MPEG cycles, and each component subsequence was accurately modelled as a TES process.

The construction of TES models is a two-step process. Step one is the definition of a background TES process. The background TES sequence serves an auxiliary role. The next step is to define the foreground TES process, and

this denotes the distortion.

CMDI [24] developed a log-normal video traffic generator for video traffic. The model is based on analysis carried out on data from the PPLive P2P streaming video network [53].

Approximately 99.5 percent of video traffic is from UDP traffic flows [24], with TCP flows only accounting for 0.46

Best-Effort

P2P and Email applications produce the most best-effort traffic on the web, with P2P having the majority share. P2P file sharing accounts for an astonishing volume of current Internet traffic. This section discusses several traffic models for both P2P and email traffic.

Krishna P *et al.* [47] performed an extensive study of P2P file sharing systems and the forces that drive them. Their work was done in three stages. First, they analysed a 200-day trace of over 20 terabytes on Kazaa P2P traffic collected at the University of Washington [47]. Second, they developed a model of multimedia workloads that they used to isolate, vary, and explore the impact of key system parameters. Third, they explored the impact of locality-awareness in Kazaa.

The analysis of the 200-day traffic trace revealed that P2P file sharing workloads are driven by considerably different processes than web traffic. Kazaa is a batch mode system mode with extremely patient users, who often wait for days, or even weeks, for their objects to be fully downloaded. As the Kazaa clients demands decrease with time. The majority of the traffic is large immutable video and audio traffic, and clients fetch objects only once, in contrast to web requests, where a client may fetch the same page thousands of times, e.g. Google.

They [47] developed a model of P2P file sharing behaviour, with client requests based on an underlying Zipf distribution. The model captures key parameters of the P2P file-sharing workload, such as request rates, number of clients, number of objects, and changes to the set of clients and objects. These parameters are used to produce a stream of client requests for objects that are then analysed. By varying the model parameters, the underlying processes that lead to certain observable workload characteristics and how changes to the parameters affect the system and its performance are explored.

The study of locality awareness showed that there is a tremendous amount of untapped locality in the Kazaa model. A majority of the external requests (approx. 86

Svoboda *et al.* [51] developed a traffic model for POP3 email traffic generation in 3G mobile networks. The model was based on traces that were extracted in a live 3G network of a mobile operator. The input traces were captured on a live GPRS/UMTS network with the METAWIN monitoring system [54].

The traffic model was developed in three steps:

1. Extracting Parameters
2. Service Request Rate
3. Probability of login-only sessions

4. Size of an Email Message

Extracting Parameters

This step started with evaluating the number of users connecting to the e-mail services. It was measured that 22.1

Service Request Rate

This step was to evaluate the number of service calls generated per user per day. Both technologies, GPRS and UMTS, are very similar to each other. This indicates that the service usage is decoupled from the access technology. Further investigation also revealed that other parameters, such as the e-mail, size did not differ between the technologies. Therefore, they did not extract the following parameters for GPRS and UMTS separately. Considering a request generation process, combined for all users in the network, they extracted the inter-arrival time between two consecutive service calls in the trace. The service generation was implemented using an exponential distributed rate.

Probability of Login-Only Sessions

This email model had to discriminate between simple login without email downloads and login with email download.

An analysis of the POP3 protocol [99] revealed that a normal login process has fewer than 13 packets, including the TCP handshake. This enabled them to set an arbitrary limit to 15 packets for uplink and downlink direction for connections which were considered to be login only. Further, they set a lower limit of 5 packets in uplink and downlink direction for a connection to be counted as valid. This rule filtered out most of the port scans and attacks polluting todays internet traces. The final number was 22.4

Size of an Email Message

The email size was the next parameter we evaluated. The email size is a bit misleading since the POP3 service can transfer many emails in one connection. The parameter is the volume in bytes that a user transfers in case there is new mail at the server.

To benchmark the model, they simulated three different scenarios:

1. They first analysed the impact of the login process compared to common models without login. With the login emulation, the service footprint is very similar to the measurement results.

2. The second scenario showed that the common models can not generate TCP footprints similar to the measurements.

3. An ADSL-like environment had a small impact on their model.

Sendil [62] analysed P2P traffic, and proposed that P2P traffic is best represented by a Pareto distribution [62]. The model by Sendil [62] proposed a Pareto probability density function for the IAT distribution. Analysis

from P2P sources showed that the traffic has a heavy-tailed distribution [62], which is best represented by a Pareto distribution. This comes from the analysis of the data flow size against the flow holding time [62]. The assumptions of Pareto distribution were verified by several heavy-tailed tests: De Haans moment method, Hill estimator, and quantil-quantil (Q-Q) plot [62].

Background

This section discusses the (M, P, S) traffic model, and one HTTP model.

Lucas *et al.* [38] developed a background traffic model, (M, P, S), for use in wide-area packet-switched network simulators. They modelled aggregate traffic generated by a large campus network, and then partitioned the aggregate traffic into sub-streams, one for each campus-level destination in the backbone network. The model was based on a collection of packet traces from a large campus network and the global Internet. The traces used were generated by the University of Virginia campus network (UVAnet), which hosts approximately 10,000 computers [38],

The model was done in three steps:

- M Aggregate Wide-Area Network Traffic Model
- P Partitioning the Aggregate Stream
- S Short-Term Packet Arrival Model

Aggregate Wide-Area Network Traffic Model

The goal of the aggregate traffic model, M, is to generate a sample path such that the correlation structure and arrival distribution of the sample path match the UVAnet traces. The model M is developed as follows:

- Use a self-similar traffic source to model the time-dependent component of the traces, and
- Transform the sample path to match the arrival density of the UVAnet traces

Accurately representing the arrival correlation is critical since packet bursts dynamically affect the packet drop rate, variation in network transmission delay, and available network throughput observed by an application [38].

Partitioning the Aggregate Stream

P takes a set of m target arrival distributions [d_1, d_2, d_3, , d_m] as input, where m gives the number of campus destinations on the wide-area network. Each target distribution function, d_i, is defined as a set of probabilities, [$p_{i,0}$, $p_{i,1}$,, $p_{i,n}$], where $p_{i,k}$ is the probability of k packet arrivals on sub-stream i during a time interval of arbitrary length. The goal of P is to construct m sub-streams such that the packet arrival distribution for each sub-stream, I, matches the target arrival density d_i.

The aggregate traffic stream, M, is expressed as a sequence of packet arrivals [x_0, x_1, , x_{n-1}], where xj gives the number of packet arrivals during an arbitrary time interval. P then constructs a set of m sequences [Y_1, Y_2, ,

Y_m]. For each target density function d_i, a discrete-time birth-death process, a Markov chain, is constructed with Ni states such that the steady-state probability that process i is in state k is exactly $p_{i,k}$.

Short-Term Packet Arrival Model

The final step is to distribute the packet arrivals generated by P into arrivals per millisecond. The analysis undertaken in [39] showed that packet arrivals, when viewed at sub-millisecond levels, occur in bursts rather than as a continuous flow. S models the short-term arrival using the well known packet-train model [27] as follows:

- S takes m parameters [m_1, m_2, , m_n] as input, where gives the mi gives the mean number of packets per burst (or train).
- For each sample $y_{i,j}$ generated by P (where i refers to ith sub-stream, and j refers to the jth sample), the mean number of trains $t_{i,j}$ is determined.
- S then uses a Poisson process with mean $t_{i,j}$ to give the size of each train.
- Finally, the train inter-arrival time is given by a Poisson process with mean r/$t_{i,j}$. r in this case is the time interval.

The (M, P, S) technique appears promising for developing realistic background traffic models in wide-area backbone network simulators, despite on being validated on a few empirical traces and only one simple method for partitioning the aggregate streams.

The World Wide Web (WWW) is a major source of traffic in the internet. HTTP traffic is traffic generated when a user is using a WWW browser. We developed a HTTP model to represent background traffic since most background traffic comes as HTTP traffic.

Cao *et al.* [7] studies have shown that the marginal distribution of packet IATs is well approximated by a Weibull distribution with a heavier upper tail than the exponential [26].

2.6.5 Summary

This section discussed previous work done in performance evaluation in wireless networks. It covered the IEEE 802.11 DCF and EDCA protocols, and also looked at channel and workload modelling, respectively.

The DCF and EDCA discussions focussed on analytic modelling, simulation, hardware experimentation, and a comparison of these modelling approaches.

The channel modelling discussion focussed on OFDM transmission, channel fading, and channel interference and noise. Suitable workload models were discussed for voice, video, best-effort, and background traffic.

CHAPTER 3

FRUGAL MODELLING

Frugal modelling is, as put by Frost and Melamed [16], is; ... *modelling that includes only those network functions that have an appreciable impact on the desired performance metrics.*

In the previous section (Section 2), a detailed analysis of related work has been carried out for the different components of the proposed model. This section collates the more desirable features from the analysis that can be incorporated into the simulation model. The goal is to develop a system that accurately represents the real system as closely as possible.

This discussion is divided into the four sub-components comprising the simulation model; the 802.11g DCF, the 802.11e EDCA, the Channel Model, and the Workload Model. Afterwards, a discussion of the simulator components and the development paradigm are presented.

3.1 DCF MODELLING

Bianchi's [4] model is one of the most cited analytic models. Both the Basic Access and RTS/CTS schemes (features are listed in Table 3.1), a finite number of STAs, and propagation delay are accounted for in the model. Also considered, are ideal channel conditions. This means that the channel medium does not suffer from any interference or noise from nearby networks, nor is there path loss or channel fading. However, the random back-off is not modelled as specified in the IEEE 802.11 standard [48]. Enhancements to this model incorporated the standard-specified back-off mechanism, and also modelled non-saturated conditions and variable length packets.

The simulation models discussed in Section 2.6.1 incorporated more features than the analytic models, and generally captured and modelled the salient features of the protocol better. Hardware experimentation represented actual networks, meaning that the results obtained are very reliable.

For simulation, it is very difficult to capture and model all the features in the hardware experiments, but it is important to incorporate all the features that have an appreciable impact on network performance. For instance, it is important to model all the IFS intervals where they should be, and the back-off mechanism should resemble that of a real STA that is in a network. Collisions and collision handling should also follow standard-stipulated protocols, since a deviation from this can bias the performance.

Table 3.1 lists and describes the salient features for the DCF model.

DCF Feature	Description
Distributed InterFrame Space (DIFS)	All STAs should listen to the channel for a DIFS interval before transmission. Transmission will only be possible if the channel was idle for the DIFS interval; otherwise transmission is suspended. The DIFS interval should be calculated in the correct manner, using the SIFS interval and the slot time.
Short InterFrame Space (SIFS)	Each time, before a CTS response, packet transmission in RTS/CTS scheme, or ACK response, a STA is required to wait for a SIFS interval.
Extended InterFrame Space (EIFS)	When resolving packet collision, all other STAs should wait for an EIFS interval before attempting to transmit while the STAs associated with the collision are backing off. This is done in order to give the colliding STAs a chance to re-transmit their packets.
Packet Buffer	Each STA should have a packet buffer to queue all packets for transmission. The buffer size should correspond to the standard-specified size. All packets that do not fit into the buffer should simply be discarded.
ACK Frames	Each successful packet transmission should be completed by an acknowledgement. If a STA does not receive an ACK within a predetermined time frame, it assumes that the transmission failed and will have to re-send the data packet.
RTS/CTS Frames	A STA wishing to transmit must send an RTS frame to the destination in the RTS/CTS scheme. The destination node must respond with a CTS frame before data packet transmission begins. The RTS and CTS frames should also be used to notify other STAs in the network that the network will be reserved for the duration of transmission.
Binary Exponential Back-off	After sensing the channel idle for the DIFS interval, all the listening STAs must back-off by decrementing their back-off counters. If the back-off counter was initially zero, a new back-off value must be generated using back-off rules. The first STA to decrement its counter to zero acquires the channel. The contention window must be doubled as per standard for every transmission failure, and reset after successful transmission.
Packet Collision	Packet collision must be accounted for and resolved each time more than one STA acquires the transmission channel. When STAs access the channel at the same time, the transmitted packets are received in error. The colliding STAs should get another chance to transmit their packets following the collision handling rules.
CW	The standard specifies a CWmin and CWmax. On every transmission failure, the CW should be doubled using exponential rules, until it reaches its maximum, and should stay at the maximum, until the packet has been transmitted successfully, or the number of retry limits has been reached, upon which the packet is discarded, and the CW is reset.
Long Retry Limit and Short Retry Limit	Whenever there is packet transmission failure, the retry counter is incremented. There are two retry limits, the short retry limit and the long retry limit, therefore there are two retry counters. The short retry limit is incremented when the size of the packet is less than the RTS threshold; otherwise the long retry count is incremented. If the retry counter reaches the retry limit, the packet is discarded, and the counter is reset to 0.
Network Allocation Vectors (NAV)	Each time the RTS/CTS scheme is used; all other STAs NAVs must be updated in order to prevent them from trying to access the channel during packet transmission.
CTS Time-Out and ACK Time-Out	When a packet has been sent, a STA waits for an ACK time out interval before attempting to retransmit again if an ACK frame has not been received. Likewise, if a STA sends an RTS frame, it waits for a CTS time out interval before attempting to retransmit the RTS frame, if a CTS frame has not been received.

Table 3.1: Salient features of the IEEE 802.11 DCF Protocol [48]

Performance metrics to be obtained from the DCF model are:

- Normalised aggregate throughput

- Collision Probability
- Channel Efficiency
- Access Delay
- Number of Dropped Packets

3.2 EDCA MODELLING

Section 2 also has a discussion on the DCA protocol. After looking at previous work done, it is important to identify all features that have a desirable impact on network performance. Since the EDCA is adapted from the DCF, several components are the same.

Table 3.2 lists and describes the salient features for the EDCA model.

EDCA Feature	Description
Arbitration InterFrame Space (AIFS)	This is a function of the AIFSN of the AC, Slot Time, and SIFS Time. It is not fixed, as with the Distributed InterFrame Space (DIFS) in DCF, but is dependent on the ACs. The AC with higher priority has a smaller AIFSN, which results in a shorter AIFS interval and gives it an advantage when contending for the channel.
Short InterFrame Space (SIFS)	Each time, before a CTS response, packet transmission in RTS/CTS scheme, or ACK response, a STA is required to wait for a SIFS interval.
Extended InterFrame Space (EIFS)	When resolving packet collision, all other STAs ACs should wait for an EIFS interval before attempting to transmit while the STAs ACs associated with the collision are backing off. This is done in order to give the colliding STAs ACs a chance to re-transmit their packets.
Access Categories (ACs)	The ACs provide support for the delivery of the traffic with UPs at the STAs. There are four ACs: Background, Best Effort, Video, and Voice traffic, with the latter having the highest priority. Each STA should have four traffic queues for each AC, and all 8 user priorities (UPs) should be implemented.
Packet Buffer	Each AC in a STA should have its own packet buffer to queue all packets for transmission. The buffer size should correspond to the standard-specified size. All packets that do not fit into the buffer should simply be discarded.
Transmission Opportunity (TXOP)	This is an interval of time when a particular STA has the right to initiate frame exchange sequences onto the channel. A TXOP is defined by a starting time and a maximum duration. The TXOP is either obtained by the STA by successfully contending for the channel or assigned by the AC. A STAs AC should send as many packets as it can in the TXOP duration, and if a packet cannot be sent within the duration, it should be fragmented.
ACK Frames	Each successful packet transmission should be completed by an acknowledgement. If a STA does not receive an ACK within a predetermined time frame, it assumes that the transmission failed and will have to re-send the data packet.
RTS/CTS Frames	A STA wishing to transmit must send an RTS frame to the destination in the RTS/CTS scheme. The destination node must respond with a CTS frame before data packet transmission begins. The RTS and CTS frames should also be used to notify other STAs in the network that the network will be reserved for the duration of transmission.
Binary Exponential Back-off	After sensing the channel idle for the DIFS interval, all the listening STAs ACs must back-off by decrementing their back-off counters. Each AC maintains its own back-off counter, and it operates independently. The first AC to decrement its counter to zero acquires the channel. The contention window must be doubled as per standard for every transmission failure, and reset after successful transmission.
Internal/Virtual Packet Collision	Virtual collision should be handled within the STA. If two ACs attempt to access the channel at the same time, the lower priority AC will back down and the higher priority AC will attempt to transmit. This type of collision is identified and resolved before any packet has been transmitted on the channel.
External Packet Collision	Packet collision must be accounted for and resolved each time more than one AC from different STAs acquires the transmission channel. When STAs ACs access the channel at the same time, the transmitted packets are received in error. The colliding ACs should get another chance to transmit their packets following the collision handling rules.
CW	The standard specifies a CWmin and CWmax for each of the ACs. On every transmission failure of a particular AC, the corresponding CW should be doubled using exponential rules, until it reaches its maximum, and should stay at the maximum, until the packet has been transmitted successfully, or the number of retry limits has been reached, upon which the packet is discarded, and the CW is reset.
Long Retry Limit and Short Retry Limit	Whenever there is packet transmission failure, the retry counter is incremented. There are two retry limits for each AC, the short retry limit and the long retry limit; therefore each AC maintains two retry counters. The short retry limit is incremented when the size of the packet is less than the RTS threshold; otherwise the long retry count is incremented. If the retry counter reaches the retry limit, the packet is discarded, and the counter is reset to 0.
NAV	Each time the RTS/CTS scheme is used; all other STAs NAVs must be updated in order to prevent them from trying to access the channel during packet transmission.
CTS Time-Out and ACK Time-Out	When a packet has been sent, a STA waits for an ACK time out interval before attempting to retransmit again if an ACK frame has not been received. Likewise, if a STA sends an RTS frame, it waits for a CTS time out interval before attempting to retransmit the RTS frame, if a CTS frame has not been received.

Table 3.2: Salient features of the IEEE 802.11 EDCA Protocol [48]

Performance metrics to be obtained from the EDCA model are:

- Normalised aggregate throughput
- Collision Probability
- Channel Efficiency
- Access Delay
- Number of Dropped Packets

3.3 CHANNEL MODELLING

Since the IEEE 802.11g and IEEE 802.11e standards use the OFDM modulation scheme, it gives a better representation of channel conditions experienced in real networks, to incorporate a model the OFDM modulation scheme. Wireless networks suffer from fading, noise and interference, and these factors affect the performance, so it is also important to represent these conditions in the channel model.

Table 3.3 lists and describes the salient features for the channel model.

EDCA Feature	Description
OFDM Transmission	OFDM is a digital multi-carrier modulation method in which a signal is split into several narrowband channels at different frequencies. Data is divided among these narrowband channels, which are orthogonal. Each sub-carrier is modulated with a conventional modulation scheme, such as quadrature amplitude modulation or phase shift keying, at a low symbol or bit rate, maintaining total data rates similar to conventional single-carrier modulation schemes in the same bandwidth.
Channel Fading	This is the fluctuation of the attenuation that a signal experiences over the communication channel, and often results in scattered signals arriving at an antenna. This phenomena is caused by obstructions and other factors such as reflections to surfaces, etc.
Interference and Noise	Noise emanates from several sources, such as atmospheric disturbances, electronic circuitry and machinery. It is stochastic in nature. It can be sub-grouped into three groups; thermal noise, filtered white Gaussian noise, and Man-made noise, and these have to be accounted for. Interference is caused by another transmitting electronic device, and like, noise, is stochastic in nature. Noise comes from systems that will normally not be expected to be producing any electromagnetic disturbance patterns, whereas interference comes from systems in which their primary goal is to produce electromagnetic disturbance patterns.

Table 3.3: Features to be included in the channel model

The channel model should record the bit error rate of the channel.

3.4 WORKLOAD MODELLING

The workload model should represent all types of traffic that are present in a typical network. Since the EDCA specifies four traffic types, voice, video, best-effort, and background, these traffic types need to be represented.

Table 3.4 Lists and describes the salient features for the workload model.

EDCA Feature	Description
Voice Traffic	Markov-Modulated Poisson Process (MMPP)
Video Traffic	Log-Normal video traffic model
Best Effort Traffic	Pareto distribution IAT Process
Background Traffic	Weibull distribution IAT Process

Table 3.4: Features to be included in the workload model

Each traffic type should produce an Inter-Arrival Time (IAT) and a size for each packet.

3.5 SIMULATOR COMPONENTS

A discrete-event and modular simulator is developed. It comprises three sub-models; called the Machine Model (MM), Channel Model (CM), and Workload Model (WLM). The MM and CM interact during transmission and there is an interface between the MM and the WLM that generates packet arrivals.

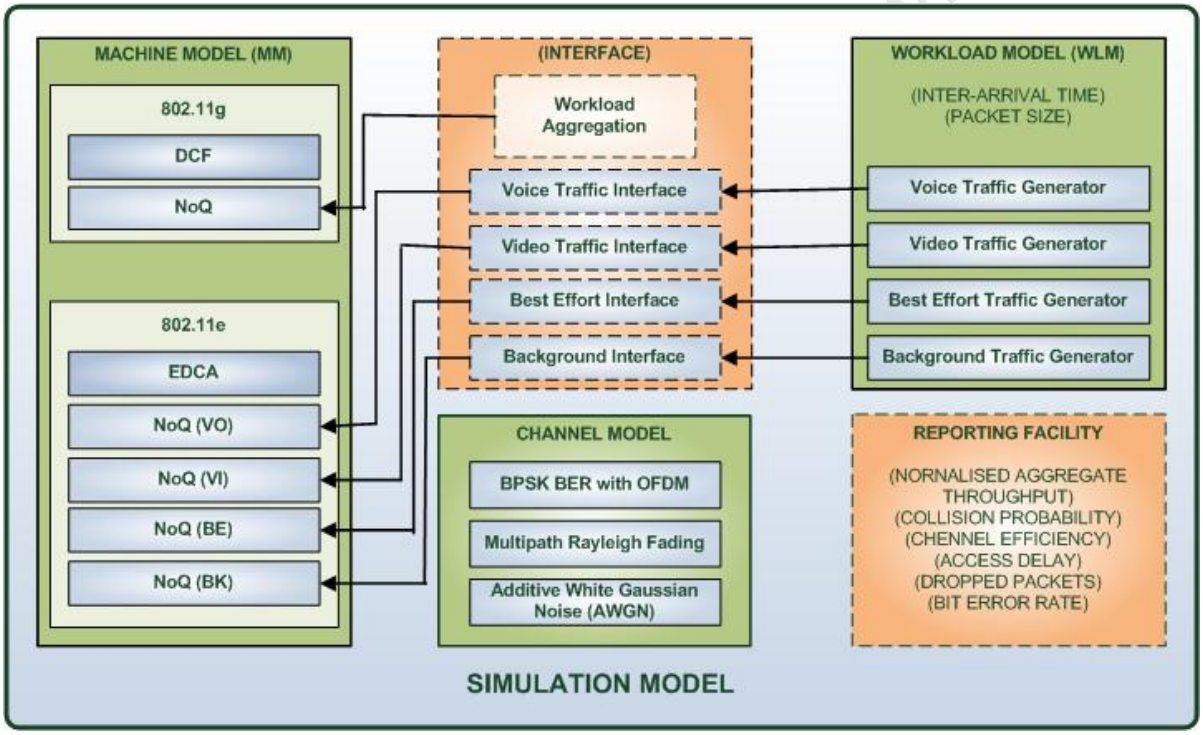

Figure 3.1: Simulation Model showing the integrated components

The MM is the Network-of-Queues (NoQ) model with the IEEE 802.11 network access protocol. The MM, shown in Figure 3.1, represents the mechanical and logical components of the operational system. It comprises two sub-components, the 802.11g and the 802.11e. The 802.11g sub-component implements the DCF protocol and the 802.11e sub-component implements the EDCA protocol. The times at which events occur are implemented as events, for example, the end of DIFS; therefore, each protocol is broken down into specific points at which events will occur that constitute the events.

The NoQ aspect of the model is in the form of STAs for the 802.11g sub-component and STAs for the 802.11e sub-component. There is a finite, but arbitrary, number of STAs, and each contains a memory buffer to store data packets, along with other protocol-specific variables. DCF STAs only have one memory buffer, since there is no

traffic differentiation. On the other hand, EDCA STAs have four memory buffers, one for each traffic class (Voice, Video, Best Effort and Background).

The WLM, shown in Figure 3.1, must provide the traffic that is served by the MM system components. Packets, and associated parameters, such as inter-arrival time and packet size, are generated by the underlying traffic generator. These packets are stored in the STAs memory buffers, which may be limited so that packets arriving when the buffer is full are discarded. The WLM generates both fixed and variable packet sizes using set parameters. Two traffic models are implemented:

1. Saturated traffic model, in which the network is forced to operate under fully saturated conditions, i.e. there is always a packet to send at each STA at any given moment in time.
2. Measured traffic model, which is an analytic traffic model that exhibits necessary statistical features typical of wireless network traffic, parameterised from a trace of measured wireless

internet traffic. There is a packet generator for each of the traffic classes. The latter is an attempt to better capture realistic network traffic behaviour with the objective to better estimate of the actual network performance.

The WLM generates four traffic streams, one for each traffic class; therefore four generators are needed. These traffic types are aggregated into one traffic stream by the workload aggregator when the DCF is in use, since there is only one memory buffer per STA.

The MM provides an interface to the WLM in order to facilitate the initialisation of the latter, as well as injection of packets into STAs memory buffers. There is an interface for each traffic class when EDCA is in use. When the DCF protocol is used, however, only one interface is used, since all traffic is aggregated.

The interface specifies four methods:

- An initialisation method in order to initialise the WLM. The initialisation requires the WLM to schedule packet arrivals for each STA.
- A method which is used to get the next packet inter-arrival time at each STA.
- A method, which works in conjunction with next packet inter-arrival time, to get the associated packet size of the next data packet to arrive.
- A synchronize method, which is used to let the WLM point to the next packet to arrive from the measured workload traces. This is only used for the measured workload.

The simulator allows the user to specify an 802.11 Profile, between the 802.11g and the 802.11e, to model, the Simulation Duration, in seconds, for which to simulate one run, and the Maximum Number of STAs, no less than eight, to simulate. Each 802.11 profile contains the specific 802.11 standard along with the PHY layer variables that it implements.

The Reporting Facility of the simulator is responsible for executing the simulation runs and writing the results to file. Execution starts with one STA, and increases to the maximum specified. For each number of STAS, 30 runs are

executed, after which, the number of STAs increases by one. The relevance of having 30 simulation runs is for the results to achieve statistical significance with normally distributed data enabling the system to calculate the sample mean and confidence intervals for each number of STAs [35]. For each number of STAs, the simulator calculates the average normalised throughput and the confidence interval (this comprises of the upper and lower confidence boundaries for a given level of confidence), collision probability, channel efficiency, access delay, dropped packets, and the bit error rate (BER) and this information is written to file. For the EDCA, these results are also calculated per AC. There are four levels of confidence; 90, 95, 98, and 99 percent. There are eight output files, two for each level of confidence; one is a text (.txt) file and the other one a comma separated variable (.csv) file.

The simulator gives the user the option to use either realistic or ideal channel conditions. Realistic channel conditions simulate OFDM modulation with Rayleigh fading and Additive White Gaussian Noise (AWGN), and ideal channel conditions assume that no channel failures occur.

CHAPTER 4

DESIGN

This section presents a detailed design for the simulator. The design of each simulator component is discussed separately in what follows.

The MM design is split into the 802.11 g, and 802.11e specifications, and each specification is presented in two distinct sections; the Model Outline and the Simulator Design. The Model Outline is a high-level representation of the simulator components, comprising of the Scheduling Framework, the System Performance Model, and the Network-of-Queues. This is followed by the design of the actual simulator, which describes the make-up of the event-driven simulator design.

The Channel Model design focuses on the OFDM modulation scheme, Rayleigh multi-path, and Additive White Gaussian Noise (AWGN) and illustrates how these three are incorporated to model wireless channel conditions.

The Workload Model design focuses on the multimedia traffic sources that are represented in the model. Four traffic types are included:

- Voice
- Video
- Best-Effort
- Background

4.1 802.11G MACHINE MODEL OUTLINE

This section discusses the machine model design.

4.1.1 Scheduling Framework

The Scheduling Framework, shown in Figure 4.1, identifies and relates the scheduling components of each STA at the MAC and PHY layer, indicating data flow through these components. Uplink traffic is traffic flowing towards an Access Point (AP) while downlink traffic flows from the AP to a STA. Uplink data, originating at

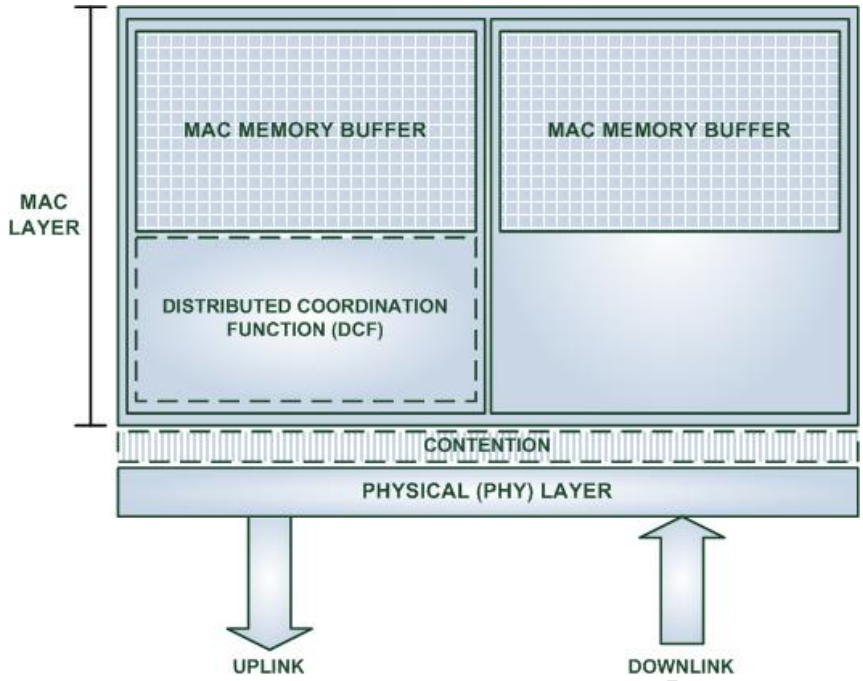

Figure 4.1: Scheduling Framework of an IEEE 802.11g STA

the STA, passes into the MAC layer by entering the Uplink MAC Memory Buffers. The AP provides control data (ACK frames) to alert the STA of successful data transmission. When the STA may transmit a packet (in accordance with the DCF protocol) the packet is sent over the transmission medium. PHY layer considerations are accounted for by specifying parameters for the network to determine channel capacity. For completeness, the framework includes the downlink traffic stream that passes through the PHY layer and enters the downlink MAC Memory Buffers. In this work, the STAs uplink data flow and associated components and the downlink flow of control information are considered.

The MAC layer memory buffer temporarily stores data packets while the Distributed Coordination Function (DCF) component manages medium access. When the memory buffer is at maximum capacity, subsequent data packets are simply discarded. Each STA contains a DCF component, which implements the DCF algorithm that determines which STA acquires the channel for data transmission. This channel access is globally negotiated and is indicated in Figure 4.1 as contention.

The PHY layer specifies the Short InterFrame Space (SIFS) and the Slot Time used. These values are used to calculate the Distributed InterFrame Space (DIFS) and the Extended InterFrame Space (EIFS). The SIFS, DIFS and EIFS are all features specified in the standard.

4.1.2 System Performance Model

The System Performance Model (SPM), shown in 4.2, is based on the Scheduling Framework, shown in Figure 4.1.1, and provides an overview of the system to model. The system consists of multiple STAs and one transmission medium. All STAs are situated in the same wireless footprint and therefore only one STA can transmit successfully at any time. Each STA has a data source which generates data that is stored in the STAs memory buffer upon arrival at the STA. Each STA also has a DCF component which executes the DCF algorithm. The DCF components are

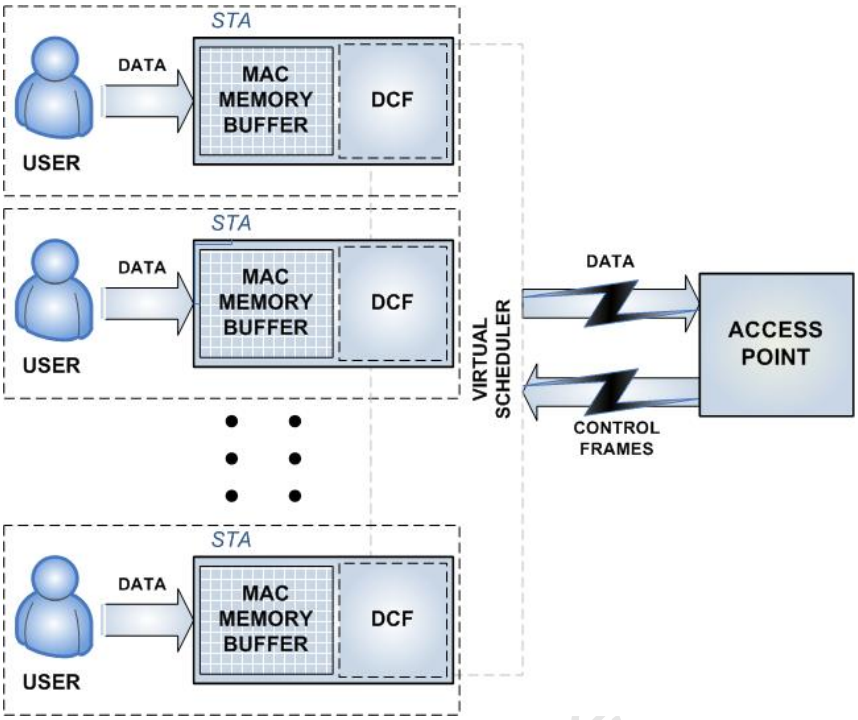

Figure 4.2: System Performance Model (SPM) for an 802.11g Network

responsible for coordinating channel access of the STAs waiting to transmit.

4.1.3 Network of Queues

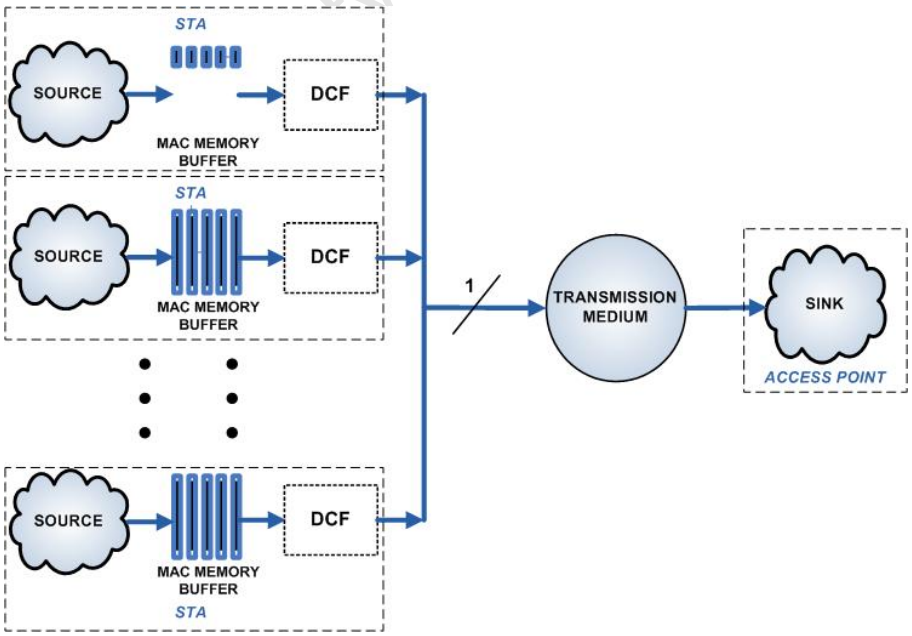

Figure 4.3: Network of Queues model for an 802.11g Network

The Network of Queues (NoQs), shown in Figure 4.3, is based on the SPM, shown in Figure 4.2. Each STA contains a finite capacity memory buffer, represented by a waiting-line to store data packets. The data packets

at each STA are serviced on a first-come-first-served (FCFS) basis and, if the buffer limit is exceeded, arriving packets are simply lost. Only one STA can transmit data successfully over the channel at a given point in time.

4.2 802.11E MACHINE MODEL OUTLINE

4.2.1 Scheduling Framework

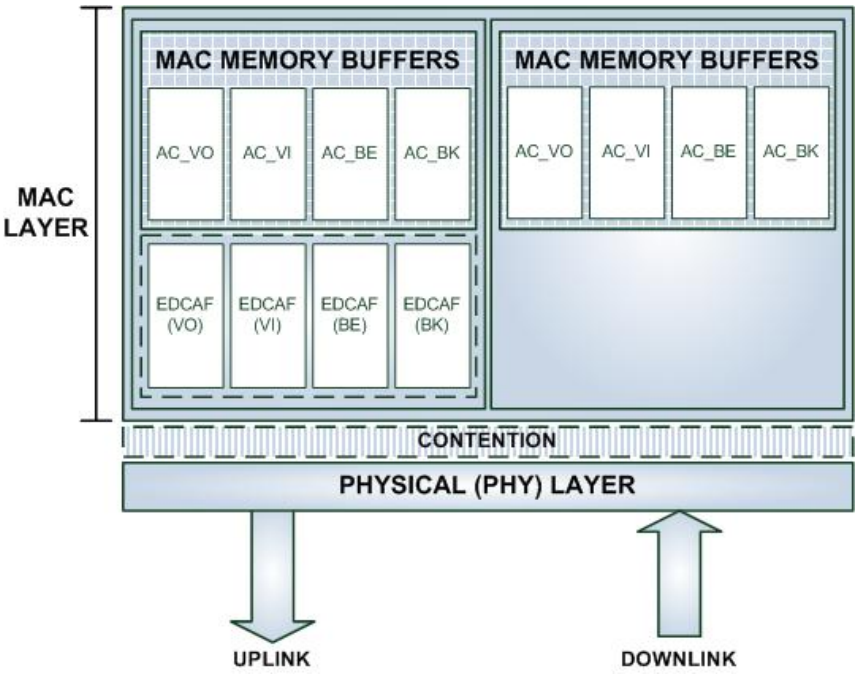

Figure 4.4: Scheduling Framework of an IEEE 802.11e STA

The 802.11e Scheduling Framework, shown in Figure 4.4, like the 802.11 scheduling framework, identifies and relates the scheduling components of each STA at the MAC and PHY layer, indicating data flow through these components.

Uplink data passes into the MAC layer by entering the Uplink MAC Memory Buffers. There are four buffers in each STA, one for each AC. The MAC layer memory buffers temporarily store data packets while the EDCAF component manages medium access. When the memory buffer is at maximum capacity, subsequent data packets are discarded.

There are also four independent EDCA functions (EDCAFs), one for each AC. Each EDCAF functions as if it is the only one in the STA, that is, it contends for the channel as an independent STA. The PHY layer specifies the Arbitration InterFrame Space Number (AIFSN) for each AC, Short InterFrame Space (SIFS) and the Slot Time used. These values are used to calculate the Distributed InterFrame Space (DIFS) and the Extended InterFrame Space (EIFS). The SIFS, DIFS and EIFS are all features specified in the standard.

4.2.2 System Performance Model

The System Performance Model (SPM), shown in Figure 4.5, is based on the Scheduling Framework, shown in Figure 4.4, and provides an overview of the system to model. The system consists of multiple STAs and one transmission medium. All STAs are situated in the same wireless footprint and therefore only one STA can

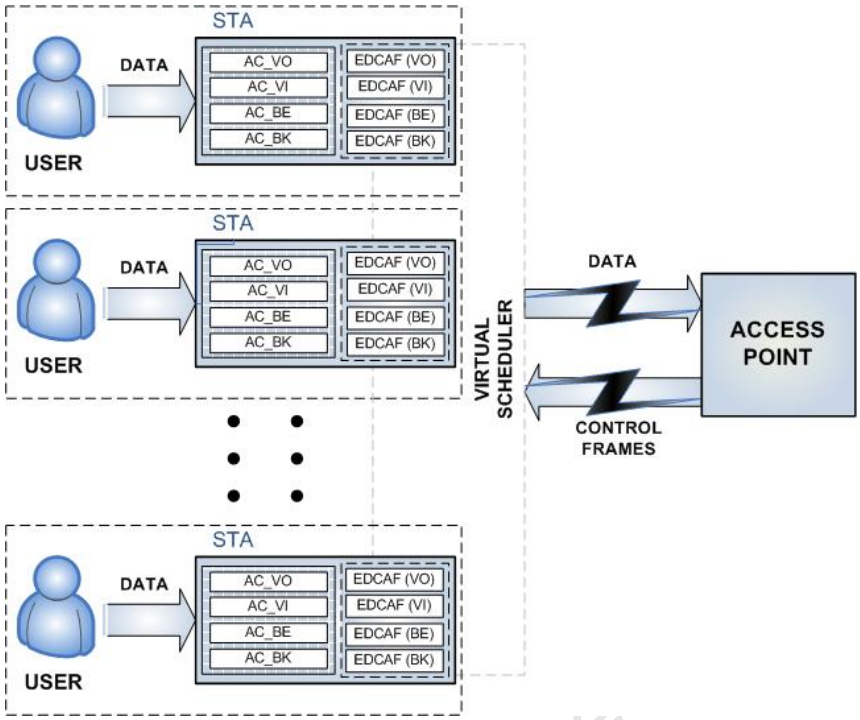

Figure 4.5: System Performance Model (SPM) for an 802.11e Network

transmit successfully at a single point in time. Each STA has four MAC memory buffers and a data source for each AC, which generates data that are stored in the STAs memory buffer upon arrival at the STA. Each STA also has four EDCAF components, one for each AC, which execute the EDCA algorithm. The EDCAF components are responsible for coordinating channel access of the STAs waiting to transmit.

4.2.3 Network of Queues

The Network of Queues (NoQs), shown in Figure 4.6, is based on the SPM, shown in Figure 4.5. Each STA contains four finite capacity memory buffer (represented by a waiting-line) to store data packets, with one memory buffer per AC. The data packets at each STA are serviced on a first-come-first-served (FCFS) basis and, if the buffer limit is exceeded, arriving packets are lost. Only one EDCAF from only one STA can transmit data successfully over the channel at a given point in time.

4.3 Simulator Design

The simulator implements the event-driven simulation paradigm as its process flow diagram illustrates in Figure 4.7. When the simulator starts, initialization takes place. *Model Initialization* is when the user specifies the parameters to use in the simulator. This parameterization includes specifying the 802.11 standard and the PHY layer to simulate, setting the maximum number of STAs, setting the simulation interval and selecting the workload model.

After initialization, control is handed over to the *Scheduler*. The Scheduler executes cyclically until the *End of Simulation* is reached. It activates the next event on the *Event List*, and also updates the *Simulation Clock* to the time of the currently executing event. The Event List maintains a list of all the events and their respective next activation time; therefore, the Scheduler activates the event with the smallest activation time from the Event List.

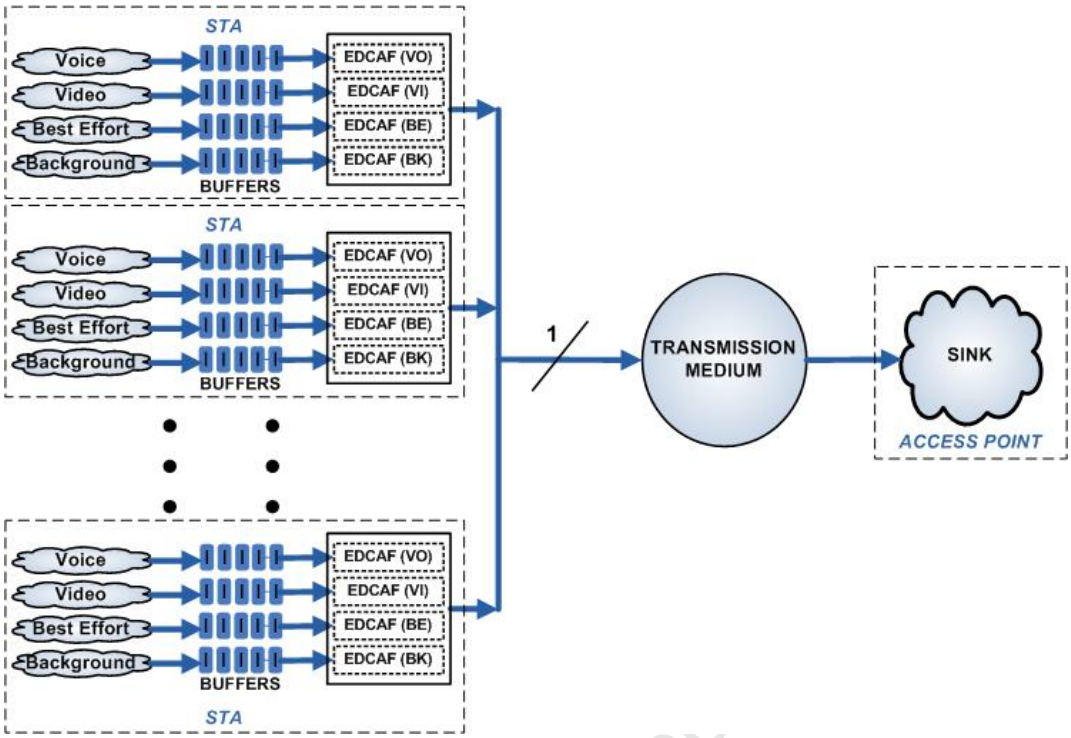

Figure 4.6: Network of Queues model for an 802.11e Network

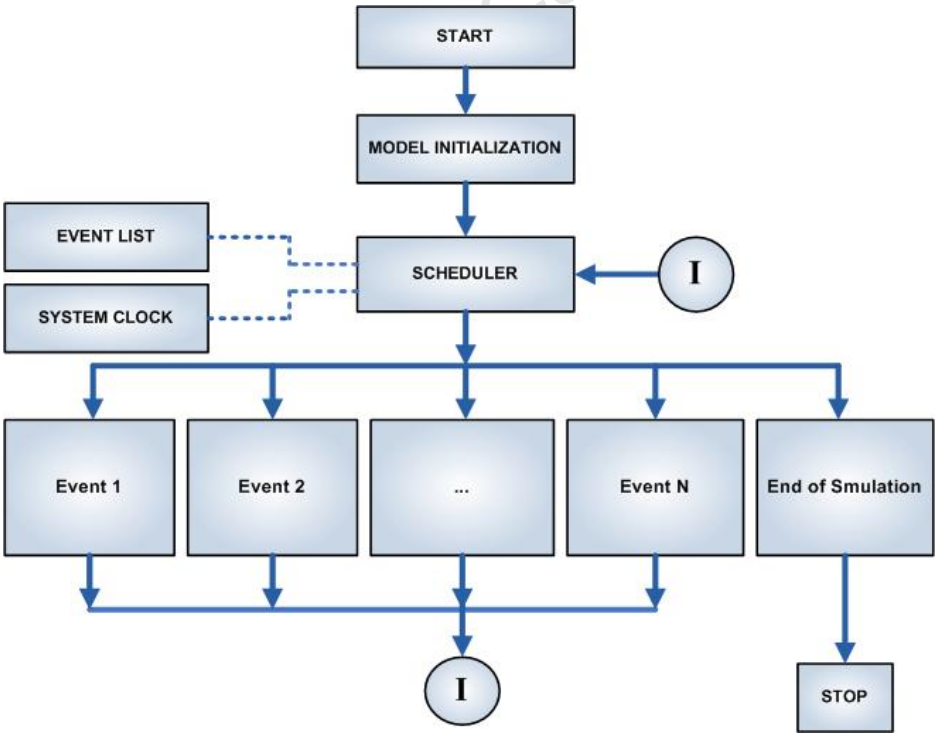

Figure 4.7: Process Flow Chart of the discrete-event simulator showing all its components and how they are linked

The Simulation Clock is discrete and maintains the current elapsed simulation time. End of Simulation is reached when the user-specified interval has elapsed, after which the simulator calculates and reports the simulation data obtained.

4.3.1 Events Identification

Events are identified through consideration of both the 802.11 protocol in question and its associated NoQ model; therefore, for the 802.11g events, we abstract the DCF algorithm, described in Section 2.2, and its associated NoQ model from Figure 4.3, and for the 802.11e model, we abstract the EDCA algorithm, described in Section 2.3, and its associated NoQ model from Figure 4.6.

802.11g Events

In this section, we identify the events from both the DCF algorithm and the NoQ model. Table 4.1 details the DCF-specific events, and Table 4.2 details the NoQ-specific events. Together, these events abstract the 802.11g individual component behaviours and interactions between them.

Event Name	Description
EndOfDIFS	This event is activated after a STA has sensed the transmission medium idle for a DIFS interval. This signals that a STA has listened to the transmission for a DIFS interval and found the channel to be idle during this interval. The STA then initiates its *back-off procedure*.
EndOfBackOff	This event is activated after all STAs wishing to transmit have backed off, with the STA that has a back-off counter of zero acquiring the transmission medium and transmitting either a data packet or an RTS control frame.
EndOfSIFS	This event is activated after waiting for a SIFS interval before an ACK response, a CTS response, or data transmission in the RTS/CTS scheme, as described in Section 2.2.
EndOfACK	This event is activated after an ACK frame has been received by a STA to complete a successful data packet transmission.
EndOfACKTimeOut	This event signals that either an ACK frame was not received or was received with error by a STA.
EndOfRTS	This event is activated after an RTS frame has been sent by a STA.
EndOfCTS	This event signals the receipt of a CTS frame by a STA.
EndOfCTSTimeout	This event signals that either an RTS of CTS frame was not received or was received with error by a STA.
EndOfEIFS	This event is used to handle packet collisions. If at least two STAs send packets at the same time, they are given another chance to contend for the transmission medium by suspending all other STAs for an EIFS interval. This event is activated after the EIFS interval has elapsed so that all other STAs can contend for the transmission medium.

Table 4.1: DCF-related events and their descriptions

802.11e Events

This section identifies events from both the EDCA algorithm and its related NoQ model. Table 4.3 details the EDCA-specific events, and Table 4.4 details the NoQ-specific events. Together, these events abstract the 802.11e individual component behaviour and interactions.

4.3.2 Stations

The number of STAs must be specified for the simulator. The simulator operates on a finite number of STAs.

Event Name	Description
PacketArrival	This event is activated when a data packet arrives at a STA. The STAs buffer is checked to see if it has not reached its limit, and if so, the packet is queued in the buffer, otherwise the packet is discarded. The packets are generated from a Workload Model Generator, and packet arrivals are independent amongst STAs, i.e. multiple packet arrivals can occur at the same time; therefore each packet arrival will specify the STA to which the data packet is addressed.
EndOfTransmission	This event signals an end of a packet transmission.
EndOfSimulation	This event signals the end of simulation. After the user specified simulation interval has elapsed, the End of Simulation event is activated. It calculates the average normalized throughput of the simulation run before initialising the simulator variables for the next simulation run.

Table 4.2: DCF NoQ-related events and their descriptions

Event Name	Description
EndOfAIFS (AC0 - AC3)	There are four such events, with one event for each AC. Each event is activated after a STA has sensed the transmission medium idle for its associated AIFS interval. This signals that a STA has listened to the transmission for an AIFS interval and found the channel to be idle during this interval. The STA then initiates its *back-off procedure*.
EndOfBackOff (AC0 - AC3)	There is one event for each AC. Each event is activated after all STAs wishing to transmit for a particular AC have backed off, with the STA that has a back-off counter of zero acquiring the transmission medium and transmitting either a data packet or an RTS control frame. These events also handle virtual and external collisions amongst the EDCAFs.
EndOfSIFS	This event is activated after waiting for a SIFS interval before an ACK response, a CTS response, or data transmission in the RTS/CTS scheme.
EndOfACK	This event is activated after an ACK frame has been received by a STA to complete a successful data packet transmission.
EndOfACKTimeOut	This event signals that either an ACK frame was not received or was received with error by a STA.
EndOfRTS	This event is activated after an RTS frame has been sent by a STA.
EndOfCTS	This event signals the receipt of a CTS frame by a STA.
EndOfCTSTimeout	This event signals that either an RTS of CTS frame was not received or was received with error by a STA.
EndOfEIFS	This event is used to handle packet collisions. If at least two STAs send packets at the same time, they are given another chance to contend for the transmission medium by suspending all other STAs for an EIFS interval. This event is activated after the EIFS interval has elapsed so that all other STAs can contend for the transmission medium.

Table 4.3: EDCA-related events and their descriptions

802.11g Stations

The following is a desfription of the instance variables and components for each STA in DCF.

Station ID - This is an integer value used to uniquely identify each STA for packet arrivals and packet transmissions and it is set during the initialisation of the simulator.

Packet Buffer - This is a queue of limited length used to store the waiting packets.

Back-off Counter - This variable stores the stations back-off value.

Event Name	Description
PacketArrival	This event is activated when a data packet arrives at a STAs buffer. The STAs buffer is checked to see if it has not reached its limit, and if so, the packet is queued in the buffer, otherwise the packet is discarded. The packets are generated from a Workload Model Generator, and packet arrivals are independent amongst STAs, i.e. multiple packet arrivals can occur at the same time; therefore each packet arrival will specify the STA and the AC to which the data packet is addressed.
EndOfTransmission	This event signals an end of a packet transmission. It determines to see if a packet has been transmitted successfully or not, and also determines whether another packet has to be transmitted.
EndOfSimulation	This event signals the end of simulation. After the user specified simulation interval has elapsed, the End of Simulation event is activated. It calculates the average normalized throughput of the simulation run before initialising the simulator variables for the next simulation run.

Table 4.4: EDCA NoQ-related events and their descriptions

Total Number of Packets Sent - This is a counter which is incremented by one after each packet transmission.

Total Number of ACKs Received - This is a counter which is incremented each time an ACK frame is received. It signals a successful transmission.

Network Allocation Vector(NAV) - This variable is used to alert a STA that the transmission medium is reserved for a fixed time interval in which it cannot access the medium.

Station Short Retry Count (SSRC) - This is a counter that is incremented each time a packet transmission fails. It has an initial value of zero. It is reset after successful transmission of the packet. If the counter reaches its limit, the packet is discarded and the counter reset to zero. The SSRC is only incremented for packets that are smaller than or equal to the RTS threshold.

Station Long Retry Count (SLRC) - This is a counter that is incremented each time a packet transmission fails. It has an initial value of zero. It is reset after successful transmission of the packet. If the counter reaches its limit, the packet is discarded and the counter reset to zero. The SLRC is only incremented for packets that are larger than the RTS threshold.

Minimum Contention Window Size (CW_{min}) - This variable holds the minimum contention window size for the STA.

Maximum Contention Window Size (CW_{max}) - This variable holds the maximum contention window size for the STA.

Contention Window Size (CW) - This variable holds the current contention window size for the STA. The CW first takes the value of CW_{min} and is incremented each time a packet transmission fails until it reaches CW_{max}, where it remains at that value, until either the packet is successfully transmitted or either the SSRC or SLRC has reached its limit, thereby discarding the packet. Either way, the contention window size is reset to CW_{min}.

Next Packet - This holds the next packet arrival time along with the packet size. The next packet arrival time is used when scheduling the next packet arrival during simulation. If there is space in the buffer, the packet is pushed to the end of the queue.

Delay Start Time - This holds the delay start time for a packet at the front of the queue to the current system clock time, when the packet is ready for transmission.

Can Transmit (*Status Variable*) - This status variable shows whether the AC has a packet to transmit. It is set to true if there is a packet in the buffer, otherwise it is set to false.

Collision (*Status Variable*) - This status is set to true if a packet collision has been detected; otherwise it is set to false.

Acquired Channel (*Status Variable*) - This status is set to true if the STA has acquired the transmission medium and is set to start packet transmission; otherwise it is set to false.

Waiting for ACK (*Status Variable*) - This status variable is set to true when the STA is waiting for an ACK frame; otherwise it is set to false.

The packet buffer will store the data packets as jobs and each job will contain the packet arrival time and the packet size.

There is an in-built packet size threshold, which determines the access mechanism (that is, basic access scheme or RTS/CTS scheme) to implement for packet transmission. If the packet size is smaller or equal to the threshold, the basic access scheme is used; otherwise the RTS/CTS scheme is used. The threshold can be set to zero for RTS/CTS scheme only, or to an arbitrarily large number for the basic access scheme only.

802.11e Stations

Like the 802.11g stations, the 802.11e stations also store station-specific variables that show the status of the station at each point during execution.

The following variables form part of the EDCA protocol.

Station ID - This is an integer value used to uniquely identify each STA for packet arrivals and packet transmissions and it is set during the initialisation stage of the simulator.

Packet Buffer (AC0) - This is a queue of limited length used to store the waiting packets for AC 0.

Packet Buffer (AC1) - This is a queue of limited length used to store the waiting packets for AC 1.

Packet Buffer (AC2) - This is a queue of limited length used to store the waiting packets for AC 2.

Packet Buffer (AC3) - This is a queue of limited length used to store the waiting packets for AC 3.

Back-off Counter (AC0) - This variable stores the ACs back-off value for AC 0.

Back-off Counter (AC1) - This variable stores the ACs back-off value for AC 1.

Back-off Counter (AC2) - This variable stores the ACs back-off value for AC 2.

Back-off Counter (AC3) - This variable stores the ACs back-off value for AC 3.

Total Number of Packets Sent (AC0) - This is a counter which is incremented by one after each AC 0 packet transmission.

Total Number of Packets Sent (AC1) - This is a counter which is incremented by one after each AC 1 packet transmission.

Total Number of Packets Sent (AC2) - This is a counter which is incremented by one after each AC 2 packet transmission.

Total Number of Packets Sent (AC3) - This is a counter which is incremented by one after each AC 3 packet transmission.

Total Number of ACKs Received (AC0) - This is a counter which is incremented each time an ACK frame is received for AC 0. It signals a successful transmission.

Total Number of ACKs Received (AC1) - This is a counter which is incremented each time an ACK frame is received for AC 1. It signals a successful transmission.

Total Number of ACKs Received (AC2) - This is a counter which is incremented each time an ACK frame is received for AC 2. It signals a successful transmission.

Total Number of ACKs Received (AC3) - This is a counter which is incremented each time an ACK frame is received for AC 3. It signals a successful transmission.

Network Allocation Vector (NAV) - This variable is used to alert a STA that the transmission medium is reserved for a fixed time interval in which it cannot access the medium.

Station Short Retry Count (SSRC) (AC0) - This is a counter that is incremented each time a packet transmission fails. It has an initial value of zero. It is reset after successful transmission of the packet. If the counter reaches its limit, the packet is discarded and the counter reset to zero. The SSRC is only incremented for packets that are smaller than or equal to the RTS threshold. This counter is for AC 0.

Station Short Retry Count (SSRC) (AC1) - This is the SSRC counter for AC 1.

Station Short Retry Count (SSRC) (AC2) - This is the SSRC counter for AC 2.

Station Short Retry Count (SSRC) (AC3) - This is the SSRC counter for AC 3.

Station Long Retry Count (SLRC) (AC0) - This is a counter that is incremented each time a packet transmission fails. It has an initial value of zero. It is reset after successful transmission of the packet. If the counter reaches its limit, the packet is discarded and the counter reset to zero. The SLRC is only incremented for packets that are larger than the RTS threshold. This counter is for AC 0.

Station Long Retry Count (SLRC) (AC1) - This is the SLRC counter for AC 1.

Station Long Retry Count (SLRC) (AC2) - This is the SLRC counter for AC 2.

Station Long Retry Count (SLRC) (AC3) - This is the SLRC counter for AC 3.

Minimum Contention Window Size (CW_{min}) (AC0) - This variable holds the minimum contention window size for AC 0.

Minimum Contention Window Size (CW_{min}) (AC1) - This variable holds the minimum contention window size for AC 1.

Minimum Contention Window Size (CW_{min}) (AC2) - This variable holds the minimum contention window size for AC 2.

Minimum Contention Window Size (CW_{min}) (AC3) - This variable holds the minimum contention window size for AC 3.

Maximum Contention Window Size (CW_{max}) (AC0) - This variable holds the maximum contention window size for AC 0.

Maximum Contention Window Size (CW_{max}) (AC1) - This variable holds the maximum contention window size for AC 1.

Maximum Contention Window Size (CW_{max}) (AC2) - This variable holds the maximum contention window size for AC 2.

Maximum Contention Window Size (CW_{max}) (AC3) - This variable holds the maximum contention window size for AC 3.

Contention Window Size (CW) (AC0) - This variable holds the current contention window size for AC 0. The CW first takes the value of CW_{min} (AC0), and is incremented each time a packet transmission fails until it reaches CW_{max} (AC0), where it remains at that value, until either the packet is successfully transmitted or either the SSRC (AC0) or SLRC (AC0) has reached its limit, thereby discarding the packet. Either way, the contention window size is reset to CW_{min} (AC0).

Contention Window Size (CW) (AC1) - The AC 1 contention window.

Contention Window Size (CW) (AC2) - The AC 2 contention window.

Contention Window Size (CW) (AC3) - The AC 3 contention window.

Next Packet (AC0) - This holds the next packet arrival time along with the packet size for AC0. The next packet arrival time is used when scheduling the next packet arrival during simulation. If there is space in the AC 0 buffer for the packet, it is pushed at the end of the queue.

Next Packet (AC1) - This holds the next packet for AC 1.

Next Packet (AC2) - This holds the next packet for AC 2.

Next Packet (AC3) - This holds the next packet for AC 3.

Delay Start Time (AC0) - This holds the delay start time for a packet at the front of the AC 0 queue to the current system clock time, when the packet is ready for transmission.

Delay Start Time (AC1) - Holds the delay start time for AC 1.

Delay Start Time (AC2) - Holds the delay start time for AC 2.

Delay Start Time (AC3) - Holds the delay start time for AC 3.

TXOP Limit (AC0) - This holds the maximum TXOP duration that the AC can transmit packets continuously. If the TXOP limit is set to zero, the AC can only transmit one packet, as is the case with AC 0.

TXOP Limit (AC1) - This holds the maximum TXOP duration for AC 1. It is also set to zero, which means only one packet can be transmitted when the AC acquires the channel.

TXOP Limit (AC2) - This holds the maximum TXOP duration for AC 2.

TXOP Limit (AC3) - This holds the maximum TXOP duration for AC 3.

Can Transmit (*Status Variable*) (AC0) - This status variable shows whether the AC has a packet to transmit. It is set to true if there is a packet in the buffer, otherwise it is set to false.

Can Transmit (*Status Variable*) (AC1) - This holds the transmit status for AC 1.

Can Transmit (*Status Variable*) (AC2) - This holds the transmit status for AC 2.

Can Transmit (*Status Variable*) (AC3) - This holds the transmit status for AC 3.

Collision (*Status Variable*) (AC0) - This status is set to true if a packet collision has been detected; otherwise it is set to false. This status is for AC0.

Collision (*Status Variable*) (AC1) - This holds the collision status variable for AC 1.

Collision (*Status Variable*) (AC2) - This holds the collision status variable for AC 2.

Collision (*Status Variable*) (AC3) - This holds the collision status variable for AC 3.

Acquired Channel (*Status Variable*) (AC0) - This status is set to true if AC 0 has acquired the transmission medium and is set to start packet transmission; otherwise it is set to false.

Acquired Channel (*Status Variable*) (AC1) - This holds the channel acquired status for AC 1.

Acquired Channel (*Status Variable*) (AC2) - This holds the channel acquired status for AC 2.

Acquired Channel (*Status Variable*) (AC3) - This holds the channel acquired status for AC 3.

Waiting for ACK (*Status Variable*) (AC0) - This status variable is set to true when AC 0 is waiting for an ACK frame; otherwise it is set to false.

Waiting for ACK (*Status Variable*) (AC1) - This holds the waiting for ACK status for AC 1.

Waiting for ACK (*Status Variable*) (AC2) - This holds the waiting for ACK status for AC 2.

Waiting for ACK (*Status Variable*) (AC3) - This holds the waiting for ACK status for AC 3.

4.4 CHANNEL MODELLING

In this section, the OFDM modulation scheme, Rayleigh multi-path fading, and AWGN are discussed in detail. Mathematical formulae will be given to demonstrate the advantages of this technique. In addition, the design of the channel model is presented, incorporating OFDM, multi-path Rayleigh fading, and AWGN. An example by Pillay [52] is consolidated with work by Xiao [75] to derive an OFDM channel model that reflects Rayleigh multi-path fading and AWGN.

4.4.1 OFDM

Overview

The OFDM modulation scheme was introduced in the 1960s, and has hardly changed. As mentioned in Section 2.4.2, the key idea of OFDM is that a single signal makes use of orthogonal sub-carriers divided into several frequency bands. Since the bandwidth is divided into narrower sub-channels, each sub-channel requires a longer symbol period. As a result, an OFDM system uses lower bit rates, but achieves higher data rates than convention communication systems. Lower bit rates also mean longer transmission distances, and longer symbol periods reduce the inter-symbol interference problem.

OFDM, however, has its drawbacks, such as the complexity of generating orthogonal sub-carriers. Other problems include the peak to average power ratio, and inter-carrier interference (ICI), which might not occur in other modulation schemes.

Modulation

The minimum frequency separation for two sinusoidal signals with arbitrary phases to be orthogonal is 1/T, where T is the symbol period [52]. In OFDM, multiple signals with frequency separation 1/T are used. The signals g_k(t) used in OFDM can be defined as [52]:

$$g_k(t) = \frac{1}{\sqrt{T}} e^{\frac{j2\pi kt}{T}} w(t) \tag{4.1}$$

where

k = 0, 1, ..., K - 1 correspond to the frequency of the signal,

w(t) = a rectangular window over [0, T-1], and

$\frac{1}{T}$ is the frequency separation

Figure 4.8 depicts the parallelisation process of a data stream. The input data stream in converted into N parallel data streams through a serial-to-parallel (S/P) port. The data duration is extended N times.

Figure 4.9 depicts the serial-to-parallel conversion process.

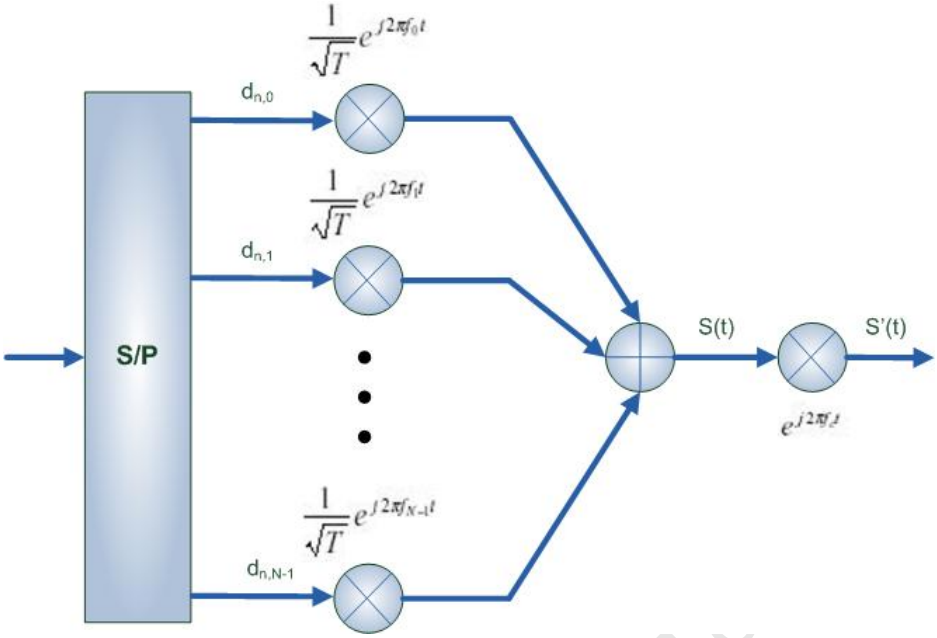

Figure 4.8: Principal OFDM modulation block diagram, adapted from [75]

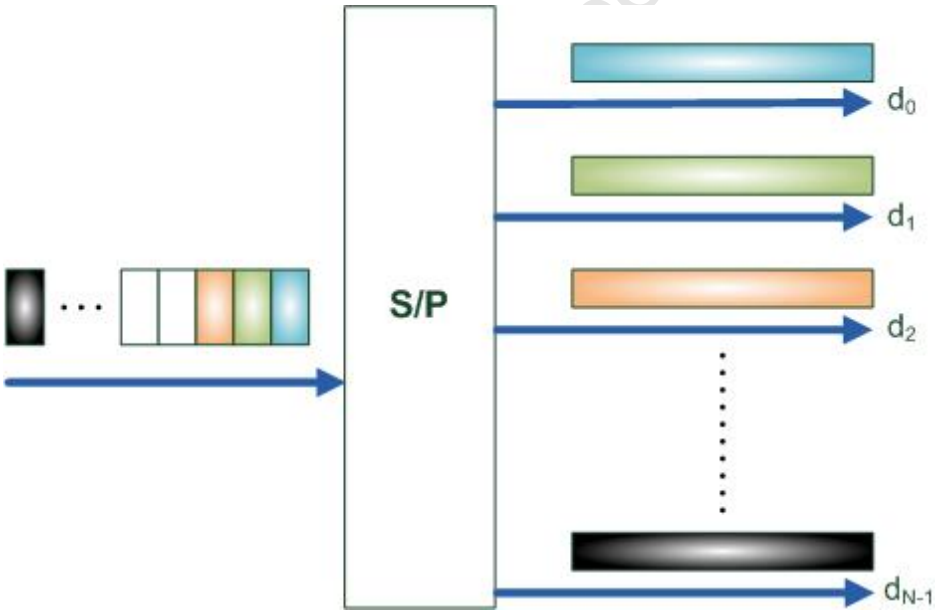

Figure 4.9: Serial to Parallel conversion, adapted from [75]

Orthogonal Sub-Carriers

When the parallel data streams are generated, each data stream is modulated and carried at different centre frequencies. The sub-carriers are centred at frequencies f_0,f_1,f_2,...f_{N-1}, and these frequencies must be orthogonal to each other. Orthogonality is defined by [52] as:

$$\int^{T} \cos(2\pi f_n t) x \cos(2\pi f_m t) dt = \delta(n-m) \tag{4.2}$$

where

δ(n - m) is the Dirac-Delta function

In OFDM, the sub-carrier frequency f_n is defined as:

$$f_n = n\Delta f \tag{4.3}$$

where

$$\Delta f = \frac{f_s}{N} = \frac{1}{NT} \tag{4.4}$$

Here, $f_s = \frac{1}{T}$ is the entire bandwidth, and N is the number of sub-carriers.

Substituting equations 4.3 and 4.4 into 4.2 can justify orthogonality for all f_0, f_1, f_2, ..., f_{N-1}, meaning that the sub-carriers are orthogonal.

Guard Interval and Cyclic Prefix

Guard Intervals (GI) are used in telecommunication to ensure that distinct transmissions do not interfere with one another. These transmissions can come from different users, or from the same user, as is the case with OFDM. Cyclic Prefix (CP) is the most common GI, and it is discussed in Section 4.4.4. The GI is introduced initially to eliminate inter-block interference, which is normally referred to as inter-symbol interference in OFDM, since multiple data symbols are associated with a single transmitted waveform.

Band-pass Signalling

After modulation by the orthogonal sub-carriers, all N sub-carrier waveforms are added to be converted to the pass-band [75]. The resulting waveform is then transmitted with a carrier frequency of 2.4 GHz, 5 GHz, 11 GHz, or 60 GHz [75]. The band-pass OFDM signal waveform is then sent to power amplifiers and antennas. Thus, the transmitted OFDM signal, x(t), from [75] can be expressed as:

$$x(t) = \sum_{n=0}^{N-1} \frac{exp(j2\pi(f_c + n\Delta f)t)}{\sqrt{T}} s(n) \tag{4.5}$$

where

s(n) represents the input data stream,

represents the symbol duration, and

f_c represents the carrier frequency

Travelling through a wireless channel, the distorted x(t) results in R(t), as shown in Figure 4.10.

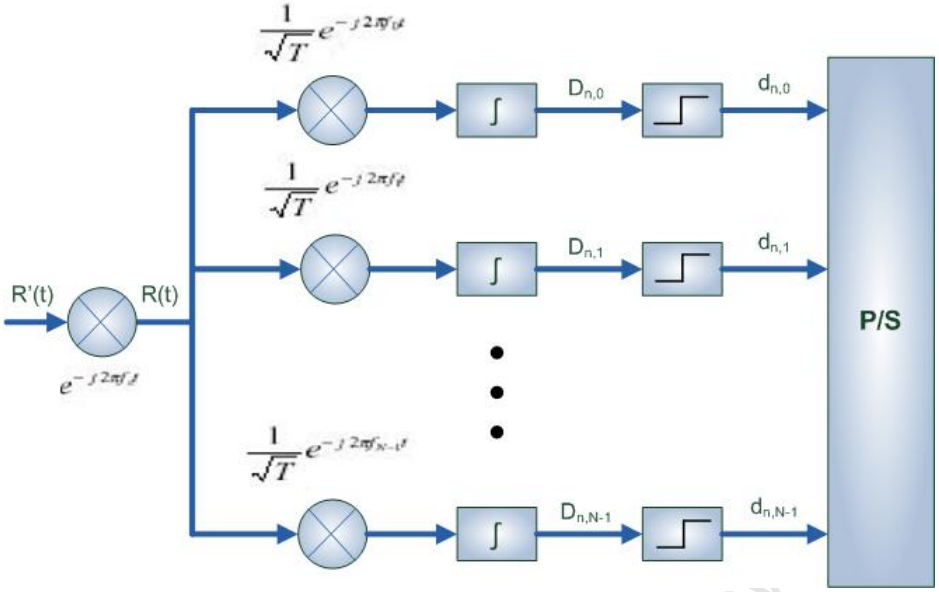

Figure 4.10: OFDM receiver, adapted from [75]

Demodulation

At the receiver, shown schematically in Figure 4.10, the received signal is converted to form a base-band signal first. Low pass filters and de-sub-carriers are then applied to separate sub-carrier waveforms. Orthogonality of waveforms ensures that only the targeted sub-carrier waveform is preserved in each sub-band. Ideally, the final extracted symbols will be identical to those transmitted in Figure 4.8.

Summary

Figure 4.11 best illustrates the OFDM modulation process of a signal from the transmitter to the receiver.

4.4.2 Rayleigh Multi-Path Fading

In this section, a discussion of the Rayleigh fading model is presented.

In a multi-path environment, we can visualize an impulse that has been transmitted from the transmitter reaching the receiver as a train of impulses, as illustrated in Figure 4.12.

As shown in Figure 4.12, the transmit signal reaches the receiver through multiple paths.

Rayleigh Fading Model

Though the channel is frequency selective (meaning that the channel spectral response is not flat and it dips or fades in the response due to reflections by obstructions, causing cancellation of certain frequencies at the receiver [14]), the channel experienced by each sub-carrier in an OFDM modulation system is a flat fading channel (meaning that all frequencies in the channel fade at the same amount [34]), with each sub-carrier experiencing independent Rayleigh fading.

The phase of each path can change by 2 radians when the delay $r_n(t)$ changes by $1/f_c$. If f_c is large, relative

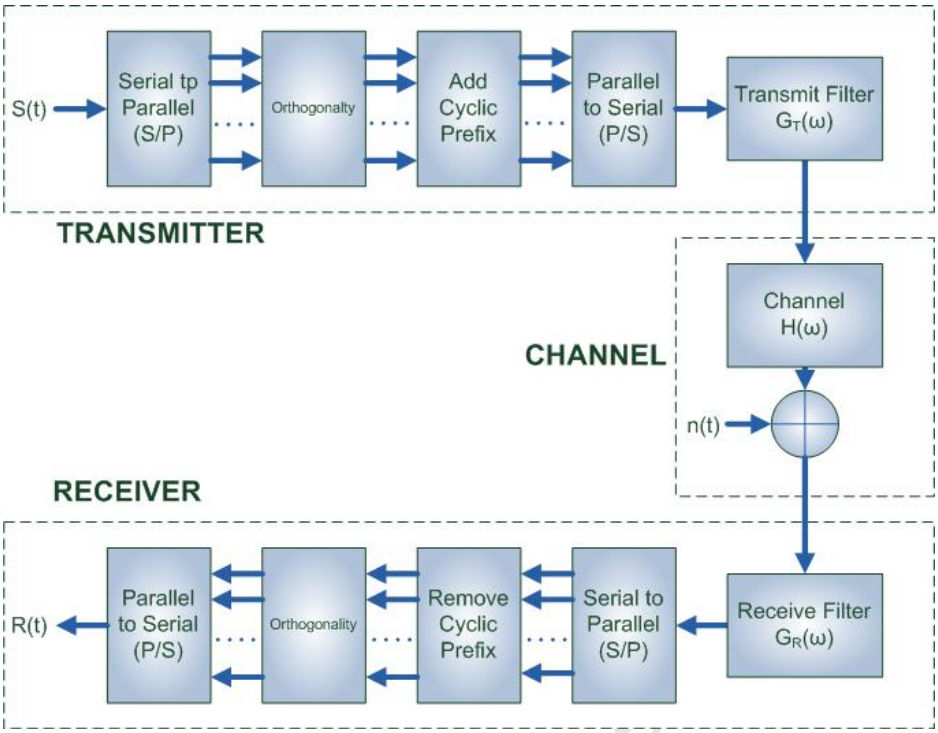

Figure 4.11: OFDM modulation-demodulation process from transmitter to receiver, adapted from [75]

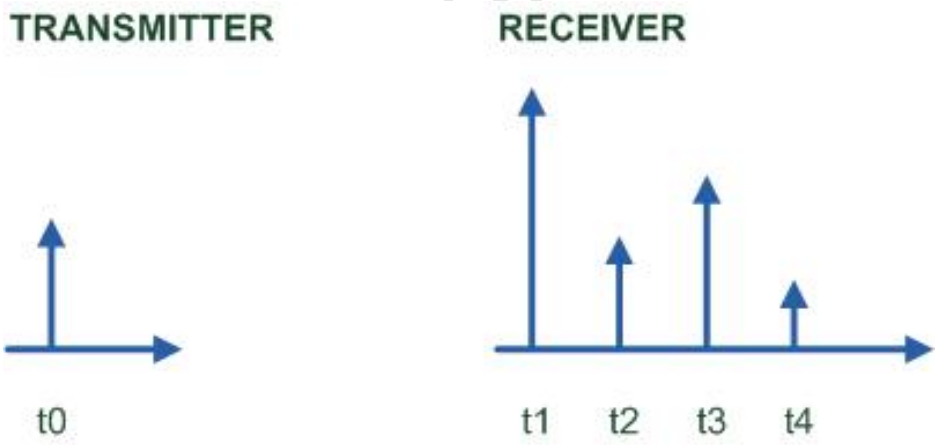

Figure 4.12: Impulse response of a multi-path channel, adapted from [52]

small movements in the medium can change by 2 radians. The distance between the transmitter and the receiver is much larger than the wavelength of the carrier frequency, f_c, and since this is the case, it is reasonable to assume that the phase is uniformly distributed between 0 and 2 radians, and that the phases of each path are independent.

When the number of paths is large, the Central Limit Theorem can be applied, and each path can be modeled by a circularly symmetric complex Gaussian random variable with time as the variable [71]. This model is called the Rayleigh channel model.

A circularly symmetric complex Gaussian random variable is given in the form:

$$Z = X + jY \tag{4.6}$$

where the real and imaginary parts have zero mean and are independent and identically distributed (iid) Gaussian random variables. In order to get a circularly symmetric complex random variable:

$$E[Z] = E[e^{j\theta}Z] = e^{j\theta}E[Z] \tag{4.7}$$

The statistics of a circularly symmetric complex Gaussian random variable is specified by the variance:

$$\sigma^2 = E[Z^2] \tag{4.8}$$

The magnitude —Z— is called a Rayleigh random variable, and has a probability density [34]:

$$p(z) = \frac{z}{\sigma^{3e}}^{\frac{-z^2}{2\sigma^2}} \tag{4.9}$$

where

z is greater than or equal to 0

This model is called the Rayleigh fading channel model, and is reasonable for an environment where there are a large number of reflectors, e.g. urban settings.

4.4.3 AWGN

When data is being sent through a wireless medium the transmitted waveform gets corrupted by noise It this work Additive White Gaussian Noise (AWGN) is considered. The term Additive means that the noise is added as opposed to multiplied by the received signal, White means that the spectrum of noise is flat for all frequencies, and Gaussian means that the values of the noise , follow a Gaussian distribution with probability density function [52]:

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(x-\mu)^2}{2\sigma^2}} \tag{4.10}$$

where

$\mu = 0$, and

$\sigma^2 = \frac{N_0}{2}$

4.4.4 The Channel Model

As mentioned above, the channel model developed in this system is an OFDM modulation system which experiences Rayleigh multi-path fading and AWGN. In this section, an integration of the OFDM modulation, Rayleigh fading multi-path, and AWGN models is presented. Figure 4.13 illustrates the channel model. The model is

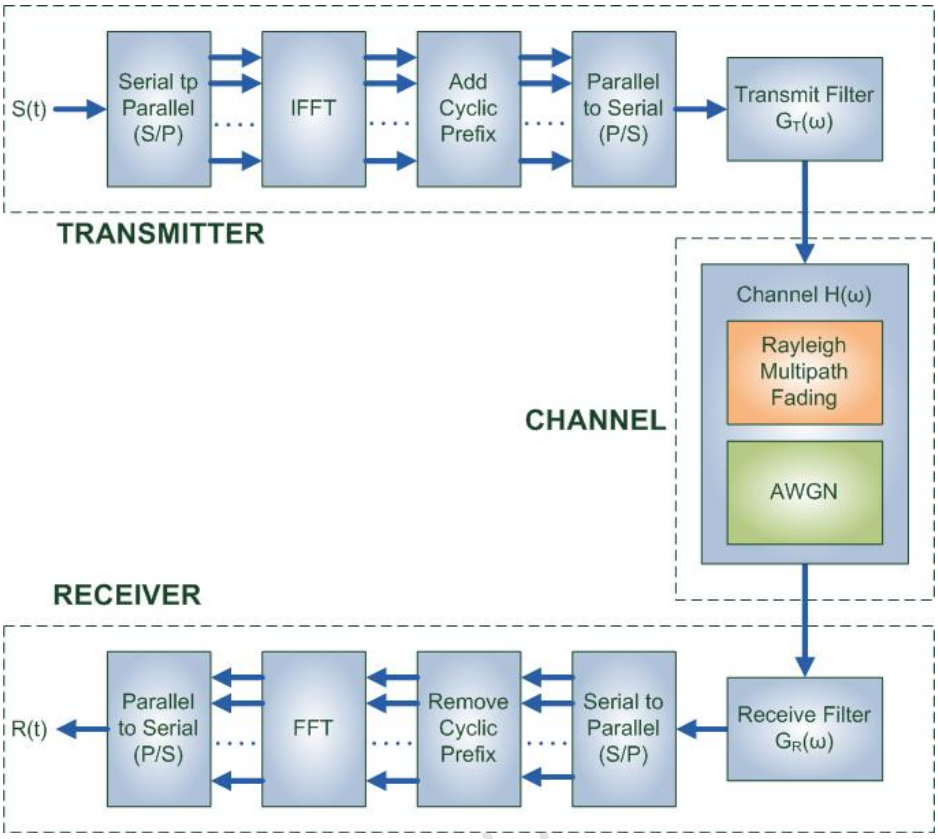

Figure 4.13: Channel Model Overview

adapted from Xiao [75], Pillay [52] and [34].

Figure 4.13 is an extension of Figure 4.11, and shows how Rayleigh multi-path fading and AWGN are added.

Back when OFDM first came onto the scene, it was a complicated process to achieve N orthogonal sub-carrier oscillators, especially when N was large. Therefore, the principal OFDM model could not be widely used. However, the introduction of Discrete Fourier Transform (DFT) and Inverse DFT (IDFT) algorithms in 1980 made OFDM a popular wireless modulation scheme since then.

In this, work the Fast Fourier Transform (FFT) and Inverse FFT algorithms [52] are used in the OFDM model.

FFT and IFFT

FFT and IFFT are efficient algorithms used to compute the DFT and its inverse, respectively. A DFT decomposes a sequence of values into components of different frequencies, but computing it directly from the definition is often too slow to be practical. An FFT is a technique to compute the same result more quickly. A DFT computation of N points is of order $O(N^2)$, while an FFT can compute the same result in only O(NlogN).

Cyclic Prefix

In multi-path fading channel models, many time-delayed versions of a transmitted waveform might be found at the receiver. Without a Guard Interval (GI) (see Section 4.4.1.4), the waveforms would interfere with each other,

as illustrated in the top half of Figure 4.14. Nevertheless, in those cases where the GI is employed, the portions of waveforms received in the GI duration are totally discarded, as illustrated in the bottom half of Figure 4.14.

This ensures that the inter-symbol interference is completely eliminated accordingly. The GI duration should be larger than the maximum channel delay in order to completely eliminate inter-symbol interference; otherwise the inter-symbol interference is not entirely eliminated.

There are several options for the GI. Among the choices is zero padding, where no waveform is transmitted in the GI duration. However, a zero padded waveform would destroy the orthogonality of sub-carriers and results in inter-carrier interference.

Figure 4.14: Guard Interval (GI) elimination of Inter-Symbol Interference, adapted from [75]

The cyclic prefix (CP) is the most commonly used GI technique. In the CP scheme [75], the GI is a copy of the partial waveform. The orthogonality of sub-carriers is preserved because Fourier bases are periodic functions.

Figure 4.15: Cyclic prefix generation, adapted from [75]

Figure 4.15 shows how the CP is added to a waveform. As depicted, the end of the waveform is copied and inserted prior to the beginning of the waveform. The time duration of the CP, denoted as TG, is chosen according to the following:

$$T_G = \frac{T}{2^K} \tag{4.11}$$

where
k is an integer, and should also depend on the maximum delay time of the channel

When the CP is applied instead of zero-padding, both inter-carrier interference and inter-symbol interference are eliminated.

Channel Model Design

The channel is a Binary Shift Phase Keying (BPSK) Bit Error Rate (BER) with OFDM modulation with 52 sub-carriers, which experiences Rayleigh multi-path fading and AWGN [52]. Successful or failed packet transmissions are dependant on the number of errors in the received packet, therefore, during simulation, the transmitted packet is compared to the received packet, and the bit errors are counted. If there are no errors, then the packet is received successfully, otherwise, there is a transmission failure.

The model requires the channel Bit-to-Energy Ratio (Eb/No), and the packet size, in bits, of the packet to be transmitted. Several steps are taken from when packet transmission resumes, until the receiver gets the packet. The steps are as follows:

1. Calculation of the Signal-to-Noise Ratio (Es/No).
2. Generation of a binary number sequence, the size of the packet.
3. BPSK modulation, that is, a 0 bit is represented as -1, and a 1 bit represented as +1.
4. Assigning modulated symbols to the 52 subcarriers [-26 to -1, +1 to 26].
5. Orthogonality of sub-carriers using IFFT and normalising the transmit power of symbols to 1.
6. Appending Cyclic Prefix (CP).
7. Convolving each OFDM symbol to a 10-tap Rayleigh multi-path fading channel. The fading on each symbol is independent. The frequency response of the fading channel on each symbol is computed and stored. This is done by applying the FFT algorithm.
8. Concatenation of multiple symbols to form a long transmission sequence.
9. Adding White Gaussian noise.
10. Grouping the received vector into multiple symbols.
11. Removing CP.
12. Converting the time domain received symbol into frequency domain. This achieved by using the FFT algorithm.
13. Dividing the received symbol with the known frequency response of the channel.
14. Extracting the required sub-carriers.
15. BPSK demodulation, that is, change all -1s to 0s and all 1s remain as 1s. This leaves a binary sequence of the received packet.
16. Counting the number of bit errors, if any.

4.5 WORKLOAD MODELLING

The performance of a network is a function of the traffic load. Hence a performance model of a network is incomplete without a model of representative workloads.

In this dissertation, the purpose of the workload model is to generate representative workloads of actual network traffic. A multimedia workload model was developed. This model consists of four different traffic sources:

- Voice over IP (VoIP)
- Video
- Best Effort
- Background

This section describes and discusses the design of each of these traffic sources.

4.5.1 VoIP Traffic

We developed a VoIP traffic model from Heffes *et al.* [20], who used a two-state Markov Modulated Poisson Process (MMPP) to model aggregate traffic from different voice sources. MMPP is a double stochastic Poisson process in which a Markov chain determines the transitions between phases. A Poisson process is used to determine the number of arrivals in a time frame at each phase. Figure 4.16 shows a Markov Arrival Process (MAP), which is the basis of the MMPP model. The MMPP is a special case of the MAP. The system has parameters $\lambda 1$, $\lambda 2$, $\omega 1$, and $\omega 2$ as input, and uses the values to determine the next state or transition.

$\lambda 1$ and $\lambda 2$ are the arrival intensities in states 1 and 2, respectively, and $\omega 1$ and $\omega 2$ are transition intensities for states 1 and 2, respectively.

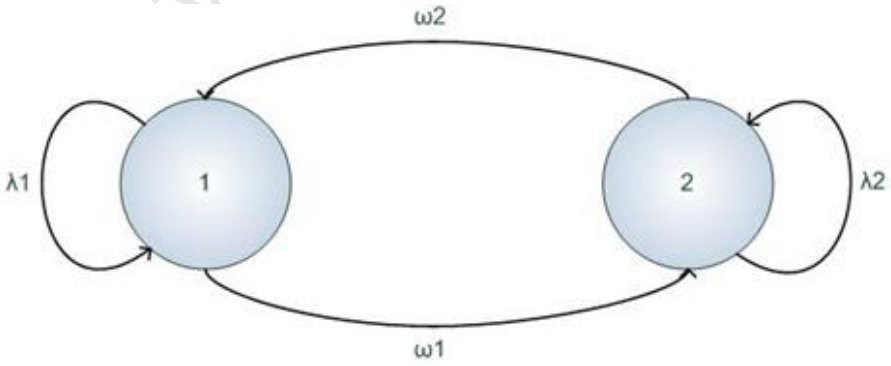

Figure 4.16: Markov Arrival Process

The importance of the MMPP model is that it is capable of modelling inter-frame dependency between consecutive frames. This is indicated by the Index of Dispersion for Counts (IDC). IDC is defined as the variance-to-mean ratio for the number of arrivals of increasing size in non-overlapping blocks [20].

Packet sizes were chosen from [12], and are randomly generated with each packets size between 100 and 1500

bytes. Voice packets are generally small, and the packet size range of 100 to 1500 bytes represents the bulk of the voice packets.

4.5.2 **Video Traffic**

A Log-Normal traffic generator for video traffic was developed from a model by CMDI [24]. The model is based on analysis carried out on data from the PPLive P2P streaming video network [53].

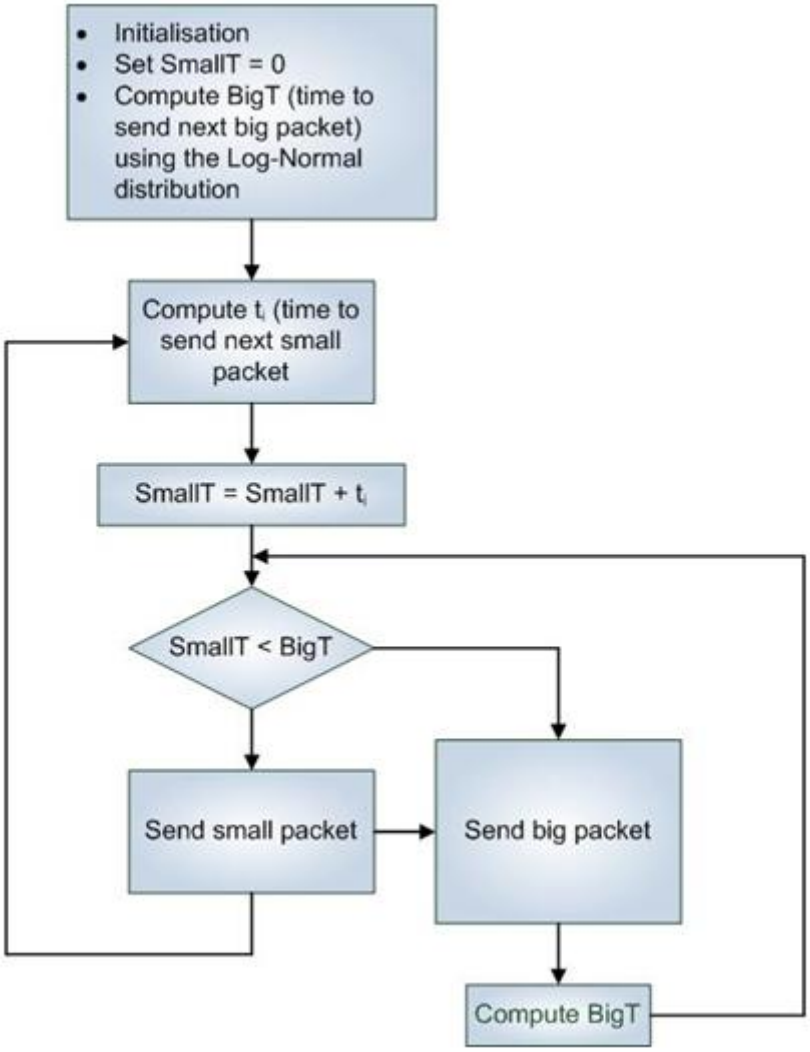

Figure 4.17: Markov Arrival Process

Figure 4.17 illustrates the video traffic generator.

Investigations by [24] showed that the majority of the recorded data are from UDP flows (which constitutes over 99.5 percent of delivered traffic), with TCP flows accounting for only 0.46 percent of the data [24] PPLive uses UDP as the main bearing protocol for video content. Network traffic was measured by [53] by connecting to a popular financial program at 9 AM, monitoring 2 hours of traffic flow, and then all the UDP packets were divided to analyze the time between consecutive packets as well as the packet size distribution, throughput, and flow size [24]. The results of interest in this work, however, are the probability distribution of the time between

consecutive packets and the packet size distributions. The Kolmogorov-Smirnov Test (K-S Test) [69] was used to determine the best fitting PDF for modelling video traffic IATs A K-S test tries to determine if two datasets differ significantly, and it has an advantage of making no assumption about the distribution of data [69].

Short packet lengths are between 100 and 1500 bytes, and long packet lengths are between 1500 and 8015 bytes [53]. Video packets are generally long packets, but there are some shorter packets as well. These packet size ranges are representative of the video packet sizes [53].

4.5.3 Best-Effort Traffic

Best effort traffic can be classified as all other kinds of traffic that are insensitive to QoS, such as jitter, packet loss, and latency. Peer-to-Peer (P2P) traffic and Email are typical examples [62].

In this work, emphasis is put on P2P traffic due to the vast volumes of traffic generated by this application [62].

We developed a Pareto process to generate packet inter-arrival times (IAT). The model was based on work done by Sendil [62]. The model by Sendil [62] proposed a Pareto probability density function for the IAT distribution to model P2P traffic. Analysis from P2P sources showed that the traffic has a heavy-tailed distribution [62]. Heavy-tailed distributions are probability distributions whose tails are not exponentially bounded, meaning that they have heavier tails than the exponential distribution [68]. This comes from the analysis of the data flow size against the flow holding time [62]. The assumptions of Pareto distribution were verified by several heavy-tailed tests: De Haans moment method, Hill estimator, and quantil-quantil (Q-Q) plot [62].

Packet sizes are randomly generated with each packets size between 100 and 1500 bytes. A study undertaken by Perenyi *et al.* [50] showed that small packet sizes for P2P traffic generating applications range between 100 and 1500 bytes in size.

4.5.4 Background Traffic

As mentioned earlier, the World Wide Web (WWW) is a major source of traffic in the internet. HTTP traffic is traffic generated when a user is using a WWW browser. We developed a HTTP model to represent background traffic since most background traffic comes as HTTP traffic.

For this model, a Weibull process was developed to generate the packet IATs. Analysis from several HTTP traffic sources suggested the Weibull distribution as a fitting distribution for the IATs as discussed by Cao *et al.* [7]. Studies have shown that the marginal distribution of packet IATs is well approximated by a Weibull distribution with a heavier upper tail than the exponential [49], [26].

Packet sizes are randomly generated with each packets size between 100 and 1500 bytes to fully represent the HTTP packet sizes [66].

4.5.5 Traffic Mapping

The traffic sources are mapped to the corresponding ACs in the EDCA protocol, as follows:

- AC0 $\Longrightarrow$ Background Traffic
- AC1 $\Longrightarrow$ Best-Effort Traffic

- AC2 $\Longrightarrow$ Video Traffic
- AC3 $\Longrightarrow$ VoIP Traffic

Traffic in the DCF protocol is not classified, therefore the system randomly selects the next packet class to send traffic from any of the traffic sources. The system generates a random number between 0 and 3 inclusive, and gets the next packet as follows:

- 0 is generated $\Longrightarrow$ Background Traffic
- 1 is generated $\Longrightarrow$ Best-Effort Traffic
- 2 is generated $\Longrightarrow$ Video Traffic
- 3 is generated $\Longrightarrow$ VoIP Traffic

4.6 PARAMETERISATION

When the system starts, it prompts the user to select the number of stations for simulation. The user must select at least 8 stations. It then prompts for the simulation duration, in seconds. The 802.11 standard is selected next, either 802.11g or 802.11e. The next step is to select the workload model, between the fixed packet size saturation workload and the realistic workload model. Finally, the user is prompted to select the channel conditions, between ideal channel conditions and realistic channel conditions, after which, the system initializes all system parameters and hands control over to the respective scheduler and the simulation starts. Figure 4.18 best illustrates this process.

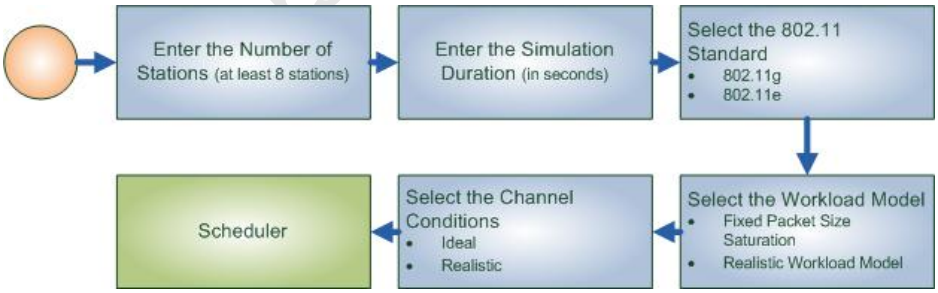

Figure 4.18: Simulation initialization process

Both 802.11 standards use the CSMA/CA access method at 54Mbps data speed, and 24Mbps control frame speed.

The system uses random numbers to generate back-off counters for the stations, to generate binary sequences for the packets, and to generate Gaussian random numbers for Rayleigh multi-path fading and AWGN. This raises the need to provide a seed for the random number generators in order to produce deterministic results.

The same seed is used across the whole system. This makes it easier to set the seed in only one place, and will make it easier to reproduce the same experiments.

4.6.1 802.11g DCF

When the 802.11g standard is selected, the 802.11g-specific parameters are selected. These are listed in Table 4.5 and they are specified in the IEEE standard [48].

Parameter	Value
SIFS	16 Microseconds
Slot Time	9 Microseconds
ACK Frame Size	112 Bits
RTS Frame Size	160 Bits
CTS Frame Size	112 Bits
UDP	64 Bits
Snap	64 Bits
IP	160 Bits
Preamble Length	16 Microseconds
PLCP Header	4 Microseconds
Signal Extension	6 Microseconds
MAC Header	304 Bits
RX Start Delay	25 Microseconds
Propagation Delay	1 Microsecond
Channel Speed	54 Bits/Microsecond
Control Frame Speed	24 Bits/Microsecond
DIFS	SIFS + (2 x Slot Time)
EIFS	SIFS + (8 x ACK Size / Control Frame Speed) + Preamble Length + PLCP Header + DIFS
CW Min	15
CW Max	1023

Table 4.5: 802.11g MAC and PHY Layer Parameters [48]

4.6.2 802.11e EDCA

The 802.11e parameters are listed in table 4.6 and are specified in the IEEE standard [48].

In 802.11e, each AC has a different CW size, with a different CW_{min} and CW_{max}. These are determined as shown in Table 4.7 and are specified in the IEEE standard [48].

4.6.3 Channel Model

BPSK OFDM modulation requires parameterization before simulation starts. The values are set within the system, and they are listed in Table 4.8. These values are specified in the IEEE standard [48].

4.6.4 Workload Model

All four traffic models that constitute the multimedia workload model need to be parameterized at system start up. The parameters are listed in Tables 4.9, 4.10, 4.11, 4.12.

Parameter	Value
SIFS	16 Microseconds
Slot Time	9 Microseconds
ACK Frame Size	112 Bits
RTS Frame Size	160 Bits
CTS Frame Size	112 Bits
UDP	64 Bits
Snap	64 Bits
IP	160 Bits
Preamble Length	16 Microseconds
PLCP Header	4 Microseconds
Signal Extension	6 Microseconds
MAC Header	304 Bits
RX Start Delay	25 Microseconds
Propagation Delay	1 Microsecond
Channel Speed	54 Bits/Microsecond
Control Frame Speed	24 Bits/Microsecond
AIFS (AC0)	7 Microseconds
AIFS (AC1)	3 Microseconds
AIFS (AC2)	2 Microseconds
AIFS (AC3)	2 Microseconds
EIFS	SIFS + (8 x ACK Size / Control Frame Speed) + Preamble Length + PLCP Header + DIFS
TXOP Limit (AC0)	0 Microseconds
TXOP Limit (AC1)	0 Microseconds
TXOP Limit (AC2)	(6.016 x 1000) Microseconds
TXOP Limit (AC3)	(3.264 x 1000) Microseconds
CW Min	31
CW Max	1023

Table 4.6: 802.11e MAC and PHY Layer Parameters [48]

CW Parameter	Value
CW Min (AC0)	CW Min (31)
CW Max (AC0)	CW Max (1023)
CW Min (AC1)	CW Min (31)
CW Max (AC1)	CW Max (1023)
CW Min (AC2)	(CW Min + 1)/2 -1 (15)
CW Max (AC2)	CW Min (31)
CW Min (AC3)	(CW Min + 1)/4 -1 (7)
CW Max (AC3)	(CW Min + 1)/2 -1 (15)

Table 4.7: Determination of CW Min and CW Max for IEEE 802.11e ACs [48]

Parameter	Value
Bit Energy-to-Noise Ratio (Eb/NO)	User-specified
FFT Size	64
Used Sub-Carriers	52
Bits per Symbol	52 (same as the number of sub-carriers for BPSK [52])
FFT Sampling Frequency	20 MHz
Sub-Carrier Spacing	312.5 KHz
Used Sub-Carrier Index	[-26 to -1, +1 to +26]
Cyclic Prefix Duration	0.8 Microseconds
Data Symbol Duration	3.2 Microseconds
OFDM (Total) Symbol Duration	4.0 Microseconds

Table 4.8: BPSK OFDM Modulation scheme parameters

Parameter	Value
Lambda 1	User-specified
Lambda 2	User-specified
Omega 1	User-specified
Omega 2	User-specified
Random Generator Seed	User-specified

Table 4.9: Model Parameterization for the VoIP Traffic Model

Parameter	Value
Variance For Large Packets (D1)	User-specified
Mean For Large Packets (D2)	User-specified
Variance For Small Packets (E1)	User-specified
Mean For Small Packets (E2)	User-specified
Random Generator Seed	User-specified

Table 4.10: Video Traffic Model Parameterization

Parameter	Value
Lambda 1	User-specified
Lambda 2	User-specified
Omega 1	User-specified
Omega 2	User-specified
Random Generator Seed	User-specified

Table 4.11: Model Parameterization for the Best-Effort Traffic Model

Parameter	Value
Lambda 1	User-specified
Lambda 2	User-specified
Omega 1	User-specified
Omega 2	User-specified
Random Generator Seed	User-specified

Table 4.12: Model Parameterization for the Background Traffic Model

The random generator seed is the seed of the random generator that generates the packet sizes.

4.7 SIMULATION OUTPUTS

In order to evaluate the performance of any system, the system in question needs to produce some sort of quantitative output. The same holds for this system, and in this section, we present the types of results to be expected.

The evaluation is measured in terms of the system experience and the user experience. Therefore, there are system-specific performance results and user-specific performance results. The system-specific performance is measured through:

- Aggregated Normalised Throughput
- Collision Probability
- Channel Efficiency
- Bite Error Rate

The user-specific performance is measured through:

- Access Delay
- Dropped packets

All these performance metrics also have confidence intervals, measured at the 90, 95, 98, and 99 percent levels. For the 802.11e results, the normalised throughput, collision probability, channel efficiency, access delay and dropped packets metrics are measured for each AC, and also for the overall system.

4.8 CLASS DIAGRAM

The conceptual view of the entire simulator is illustrated in Figure 4.19. The Main class is the driver class of the simulator. It starts up and prompts for input from the user. It also initialises the PHY layer profile in use. Depending on the standard selected by the user, it calls the Scheduler (for 802.11g) or the QScheduler (for 802.11e) to run the simulation. The Scheduler or QScheduler initialises the Workload Model Interface, the Workload Model Generator and the BPSK BER model, the System Clock and the Event List or calendar, the Events or QEvents, and the STA list.

The Scheduler checks the Calendar for the next event that is to occur, updates the System Clock to that of the next event, and then activates that event. Below is a summary of each of the classes.

The System Clock is discrete, and maintains the current simulation time that has expired. The event list maintains a list of all the events, along with the time that the event will occur next.

The Workload Model Generator is responsible for generating data packets, and the interface acts as the link between the stations buffers and the workload model generator. The Events and QEvents classes implement the

DCF and EDCA algorithms, respectively, and maintain the STAs. They contain all the events required for the DCF and EDCA algorithms and update the status of the channel when transmission either begins or stops. The BPSKBER class runs the OFDM modulation scheme with Rayleigh multi-path fading and AWGN, and uses the Complex, FFT and Transformations classes to manipulate the data.

The complex class generates complex numbers and performs arithmetic operations on complex numbers. The FFT class performs FFT and IFFT operations on a given array of data. The Transformations class runs processes such as *Matrix Transpose, FFT Shift, Convolution*, and *Reshaping* of data.

Finally, the SimulationOutput and QSimulationOutput classes temporarily hold simulator output before it is written to files.

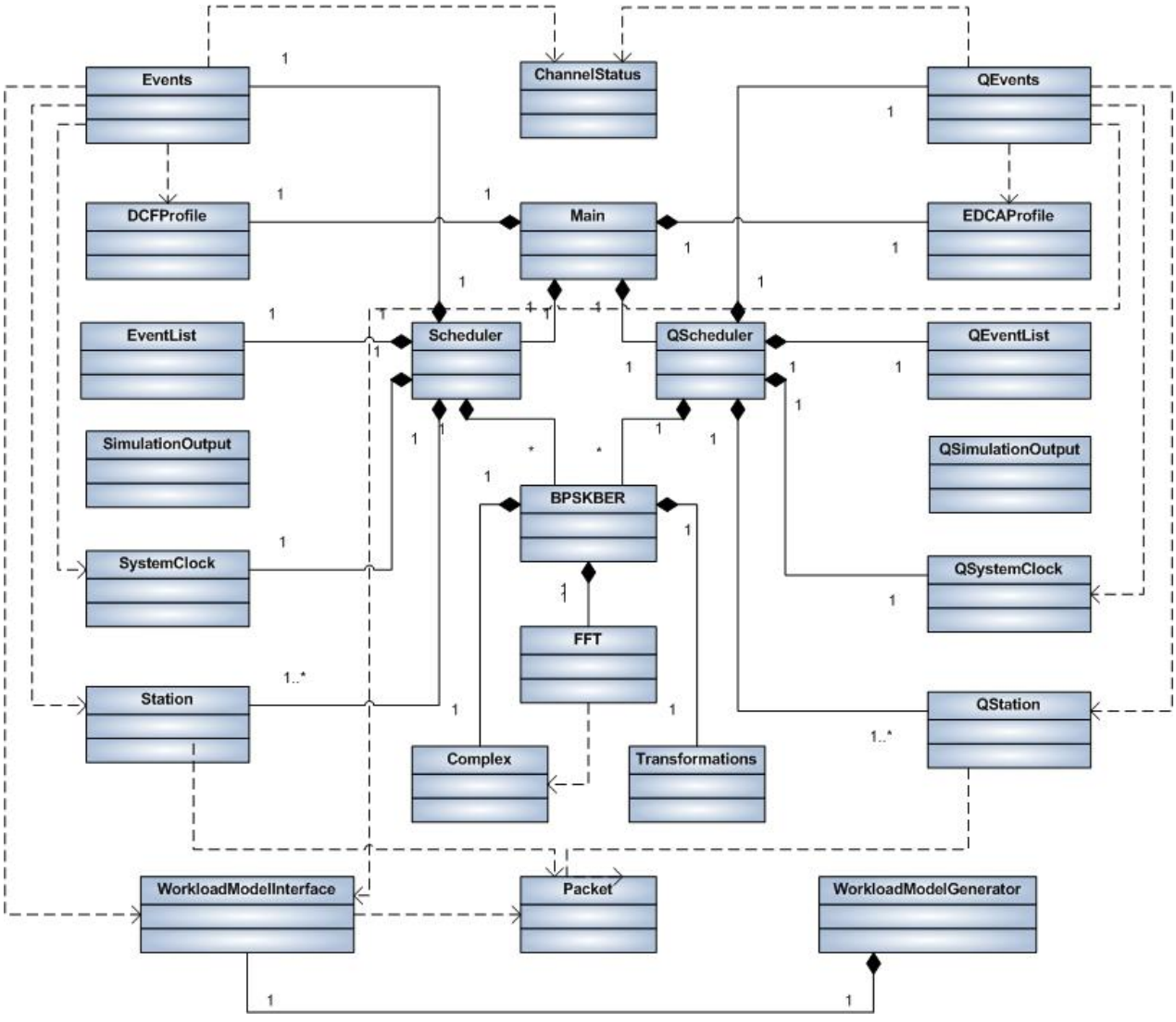

Figure 4.19: Conceptual class diagram

CHAPTER 5

IMPLEMENTATION AND TESTING

In this section, the implementation of the simulator is discussed. The 802.1g DCF is discussed first, followed by the EDCA, the channel model, and finally, the workload model.

5.1 802.11G DCF

The 802.11g DCF is implemented using the event-driven paradigm, with events discussed in Section 4.3.1. Figure 5.1 is an illustration of the DCF simulator, showing how all components are associated.

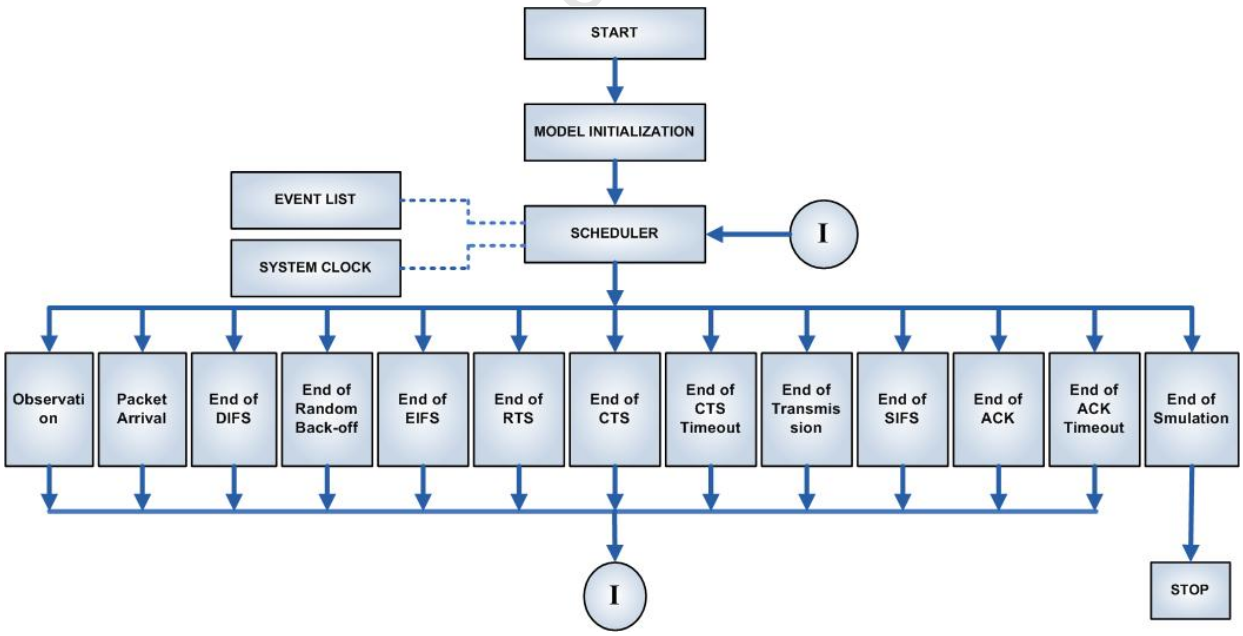

Figure 5.1: DCF process flow chart

The Scheduler controls the operation of the entire simulator. It picks the next event to execute, and maintains the system clock and the event list (calendar).

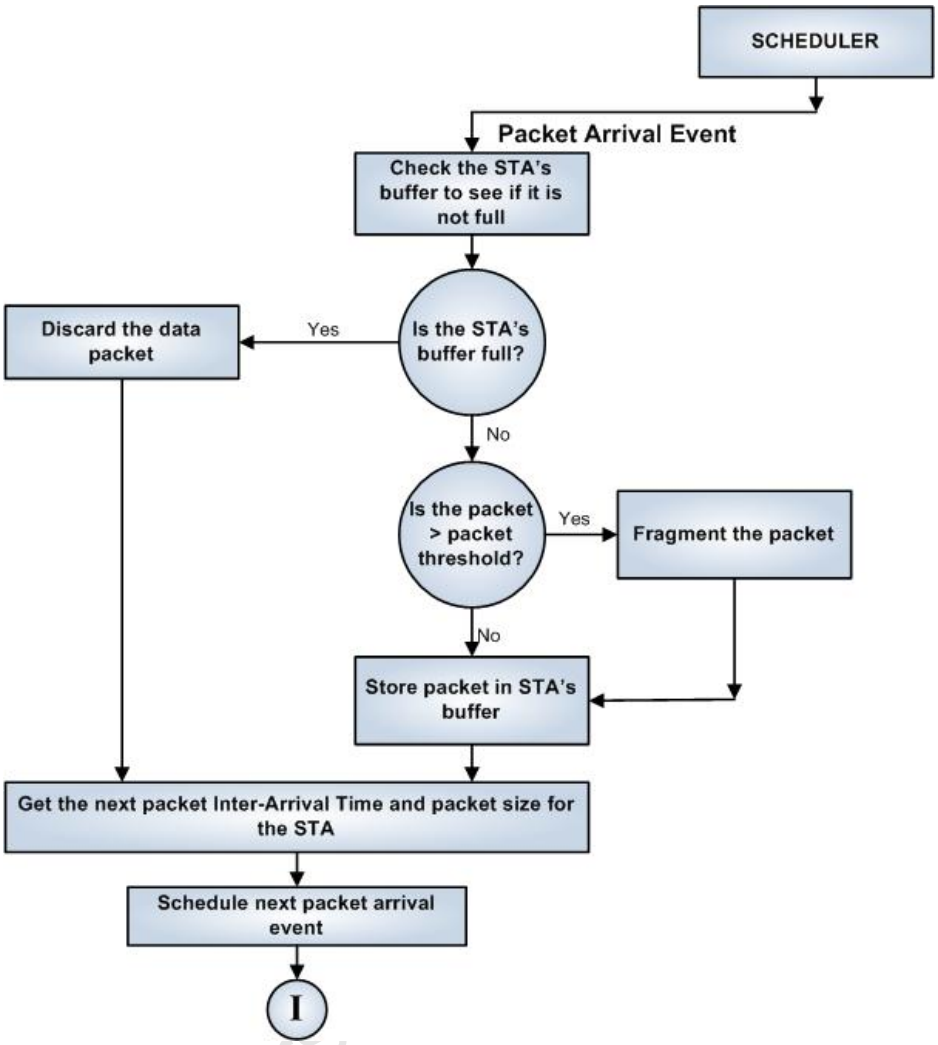

Figure 5.2: DCF Packet arrival event process flow

5.1.1 Packet Arrival

The Packet Arrival event, shown in Figure 5.2, facilitates the arrival of packets into STAs buffers. Packet arrivals are independent for each STA; therefore multiple STAs can simultaneously receive data packets.

At initialisation, the simulator schedules a packet arrival for each STA. The Packet Arrival event is always activated when a data packet is ready to enter the memory buffer of a STA. First of all, it checks to see all the STAs Pending Packet next arrival time to see which STA is supposed to have a packet arrival at that time. Once the STA is acquired, it the then checks to see if that particular STAs memory buffer can accommodate the pending packet. If the memory buffer is full, the packet is discarded. If there is space in the buffer, the event checks the pending packet size to verify whether it is greater than the packet size threshold and if it is, the packet is fragmented before being stored in the buffer; otherwise, it is stored at the end of the buffer.

The event then gets the next packet inter-arrival time and packet size for the STA, updates the Pending Packet next arrival time and packet size for the STA, and then schedules the next Packet Arrival event. The next packet arrival time for a STA is determined by adding the inter-arrival time to the current system time. The next Packet

Arrival event is scheduled by going through all STAs pending packet arrival times to see which has the least occurring time.

5.1.2 End of DIFS

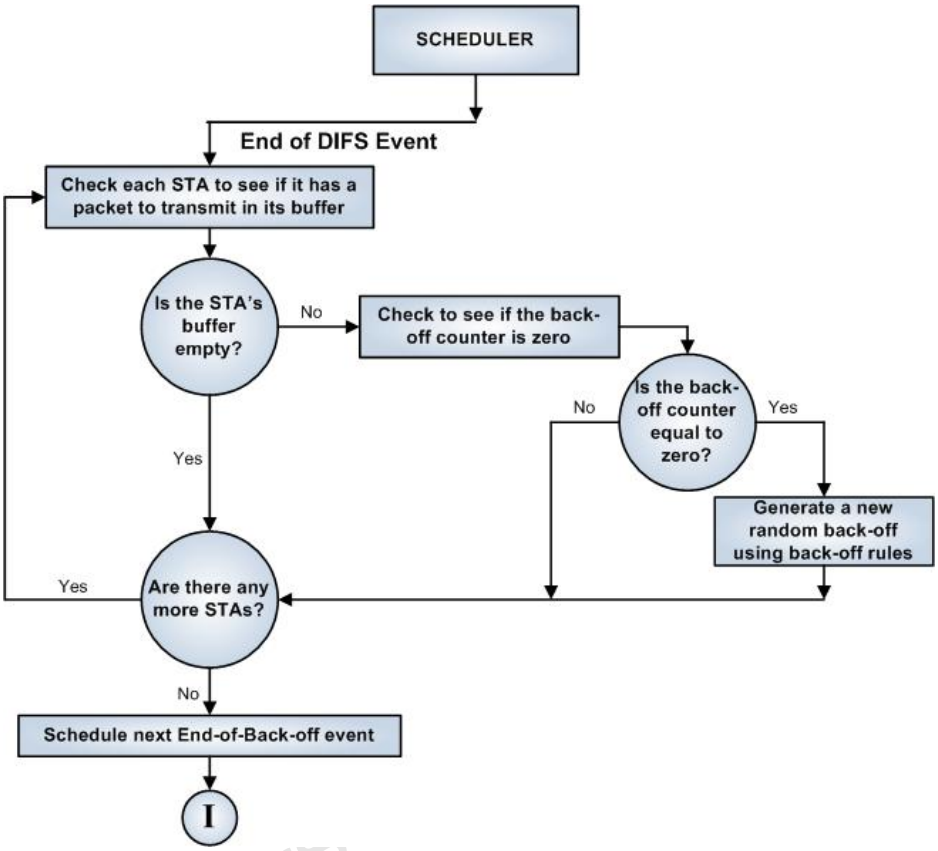

Figure 5.3: DCF End of DIFS event process flow

The End-of-DIFS event, shown in Figure 5.3, is activated when the channel has been idle for a DIFS interval. This event identifies all the STAs that have data packets to transmit in their buffers. The event examines each STAs buffer in turn and if the STA is ready to transmit, it checks its back-off counter. If the back-off counter is zero, a new random back-off is generated; otherwise the current back-off value is used. If a STA is not ready to transmit, it is skipped and the next STA is examined.

After going through all the STAs buffers, the event schedules the End-of-Back-off event. This is achieved by identifying the smallest back-off value from the back-off counters of STAs that have a packet to transmit and then scheduling the End-of-Back-off event to the current system clock plus the smallest back-off value identified.

5.1.3 End of EIFS

The End-of-EIFS event, shown in Figure 5.4, is activated when the channel has been idle for an EIFS interval. It is quite similar to the End of DIFS event described in Section 5.1.2. This event identifies all the STAs that have data packets to transmit in their buffers. The event examines each STAs buffer in turn and if the STA is ready to transmit, it checks its back-off counter. If the back-off counter is zero, a new random back-off is generated; otherwise the current back-off value is used. If a STA is not ready to transmit, it is skipped and the next STA is

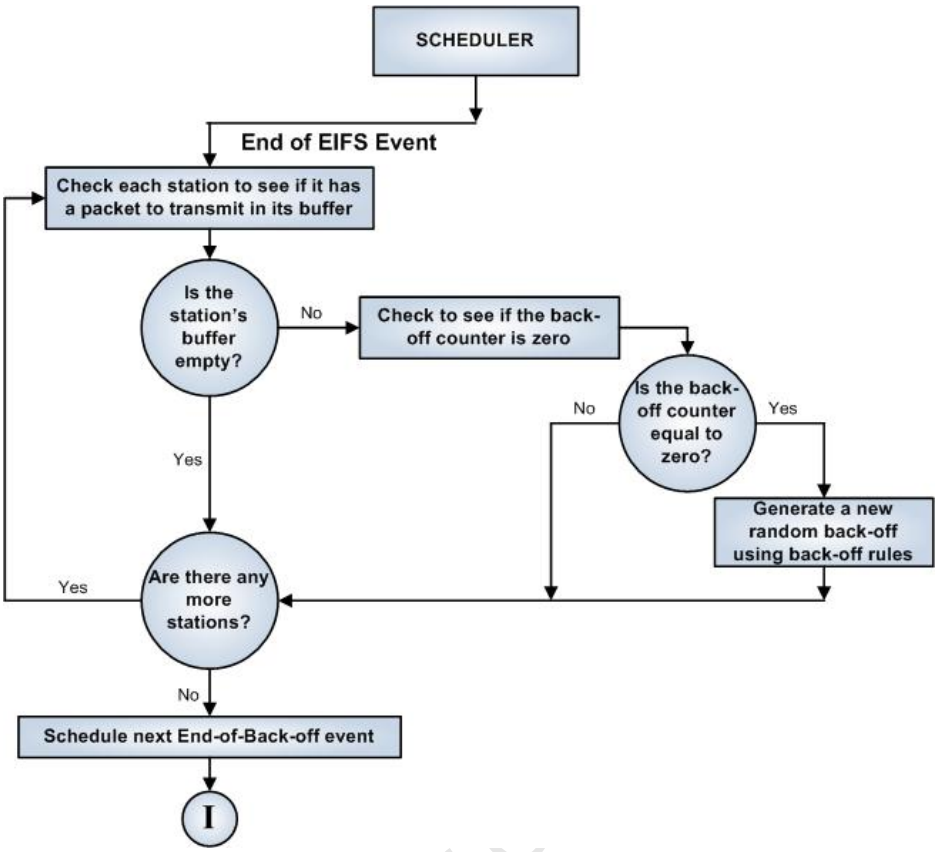

Figure 5.4: DCF End of EIFS event process flow

examined.

After going through all the STAs buffers, the event schedules the End-of-Back-off event. This is achieved by identifying the smallest back-off value from the back-off counters of STAs that have a packet to transmit and then scheduling the End-of-Back-off event to the current system clock plus the smallest back-off value identified.

5.1.4 **End of Back-off**

Figure 5.5 illustrates the End of Back-off event.

When the End of Back-off event starts executing, it first checks to see if it is resolving a packet collision. It then checks all STAs back-off counters to see if there are colliding STAs. When STAs collide, they start their back-off procedures to contend for the transmission channel again. If there are colliding STAs contending for the network, it only deals with these STAs; otherwise, it deals with all the STAs.

When dealing with colliding STAs, the event loops through all the colliding STAs to find the one with the smallest back-off value. It then checks to see if there are other colliding STAs with the same back-off value. The same procedure is followed when it is not resolving collisions. In this case though, it loops through ALL STAs to find the smallest back-off value. It then checks to see if there are other STAs with the same back-off value. If there are at least two STAs with the smallest back-off value, they will acquire the channel and try to transmit at the same time, resulting in packet collision.

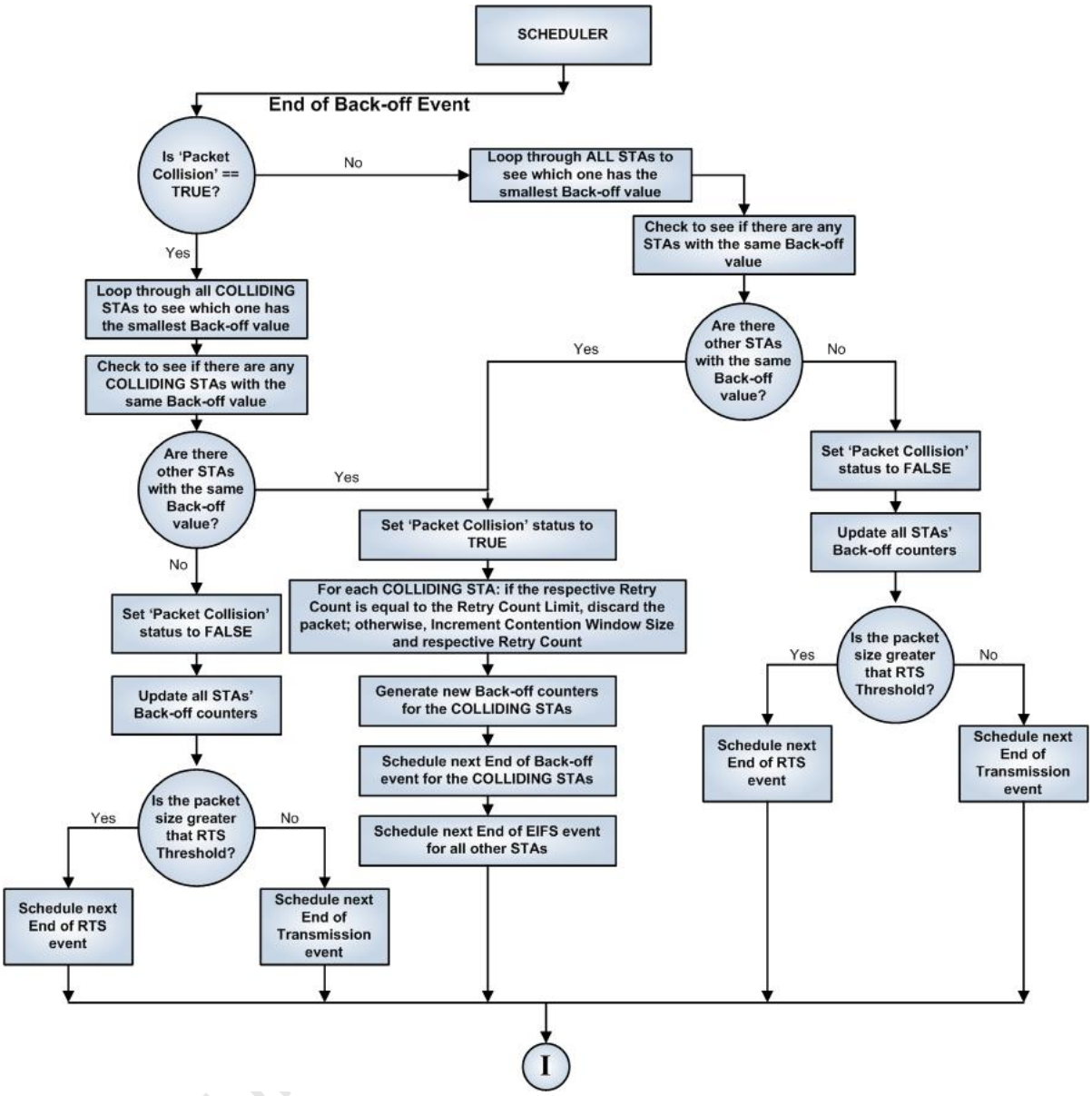

Figure 5.5: DCF End of Back-off event process flow

If there are other STAs with the same back-off value, it sets the Packet Collision status to TRUE. This is a status variable that shows whether collision has occurred or not. It then loops through all STAs to update their retry counts and contention window sizes. Each STA has two retry counts; Short Retry Count and Long Retry Count. The Short Retry Count is incremented whenever the size of the packet to be transmitted is less than or equal to the RTS Threshold and the Long Retry Count is incremented when the packet size is greater than the RTS Threshold. There are retry count limits, which, if reached, the packet will be discarded, and the respective retry count and the contention window size are reset. The contention window size is doubled using the exponential back-off rules, and, if it reaches the maximum window size, it will not change until it is reset (through successful packet transmission or retry count limit reached). After updating the retry counts and contention window sizes, new back-off values for the respective colliding STAs are generated. The event the schedules the next End of Back-off event for the colliding STAs, and the End of EIFS event for the other STAs as well.

If there are no other STAs with the smallest back-of value, it means that only one STA is able to transmit; therefore, this event sets the Packet Collision status to FALSE and updates all the STAs back-off counters. It then checks to see whether the basic access or the RTS/CTS scheme is used. This is achieved by comparing the respective packet size to the RTS Threshold. If the packet size is less than or equal to this threshold, the basic access scheme is used, and the event schedules the next End of Transmission event to the time of system clock plus the time it takes to transmit the packet plus the End of ACK Timeout event; otherwise, it uses the RTS/CTS scheme and schedules the next End of RTS event, as well as the End of CTS Timeout event.

5.1.5 **End of Transmission**

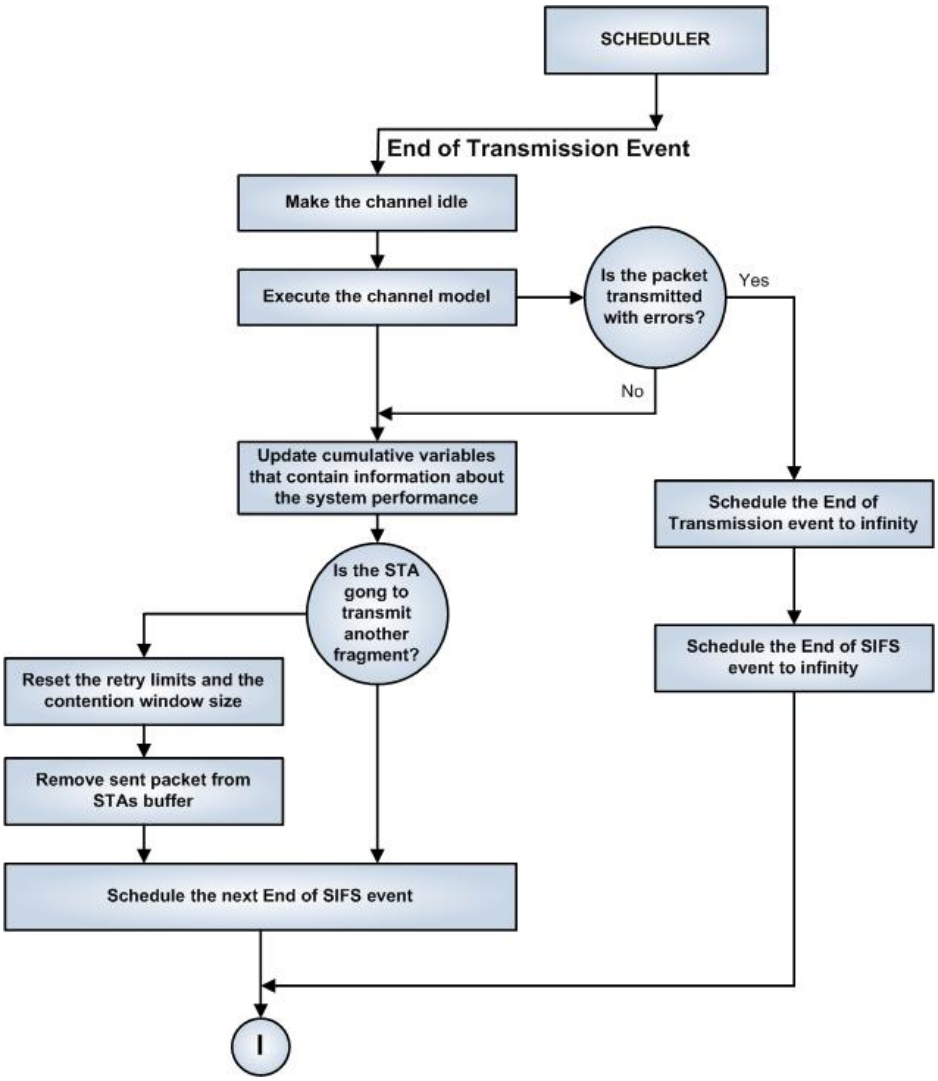

Figure 5.6: DCF End of Transmission event process flow

This event, shown in Figure 5.6, is activated when transmission of a data packet is complete. The event first changes the channel status of the transmission medium to idle. It then runs the channel model, passing the packet size as a parameter. It executes the channel model to determine whether the packet was transmitted with error or not. If the packet was transmitted with error, it schedules itself and the End of SIFS event to infinity. Otherwise, it updates the system statistics in the current observation window, such as the number of packets transmitted so far for the STA and the number of bits that have been transmitted in the network.

The event then schedules the next End of SIFS event to the current system clock time plus the SIFS interval.

5.1.6 End of SIFS

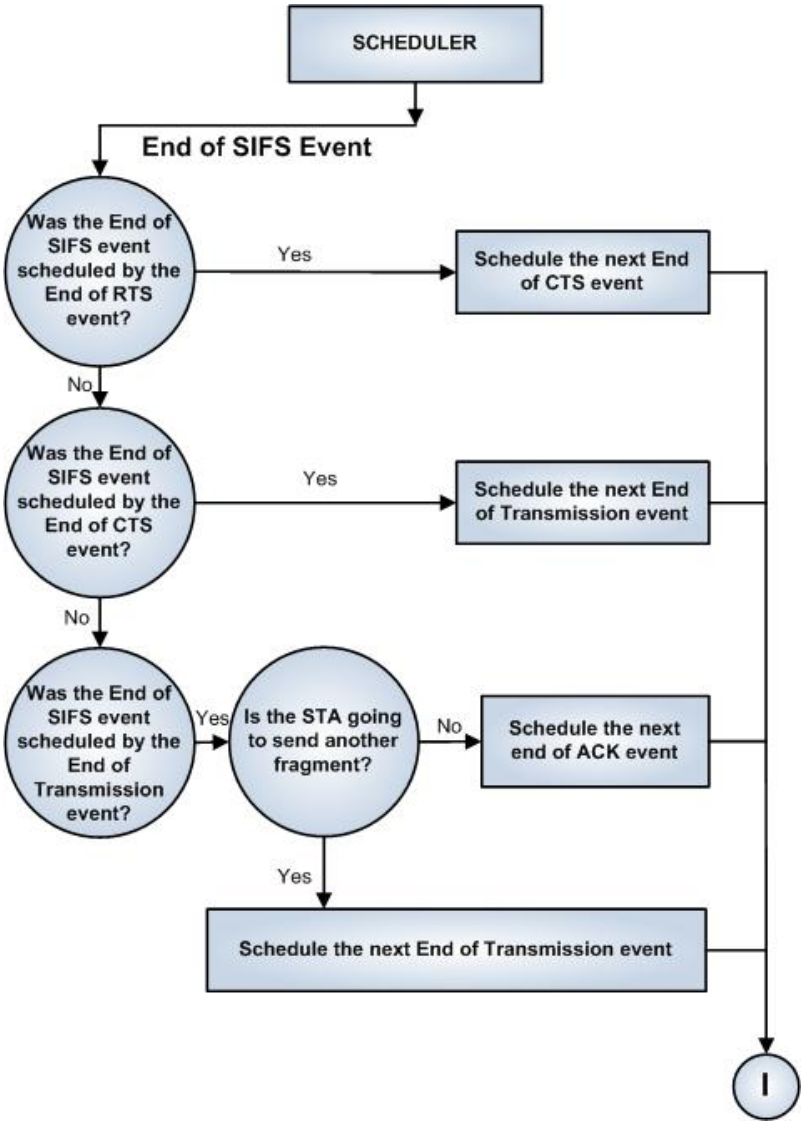

Figure 5.7: DCF End of SIFS event process flow

The End of SIFS event, shown in Figure 5.7, can be scheduled by three events; End of RTS, End of CTS, and End of Transmission.

If it is scheduled by the End of RTS event, it schedules the next End of CTS event, if it is scheduled by the End of CTS event; it schedules the next End of Transmission event, and if it is scheduled by the End of Transmission event, first checks to see if there is a multiple fragment to transmit. If so, it schedules the next End of Transmission event using the next packet fragment size as a parameter to determine the transmission duration, which means that the transmission of a fragment takes place. Otherwise, it schedules the next End of ACK event.

The channel status is then set to TRUE to show that a packet or frame is being transmitted.

5.1.7 End of RTS

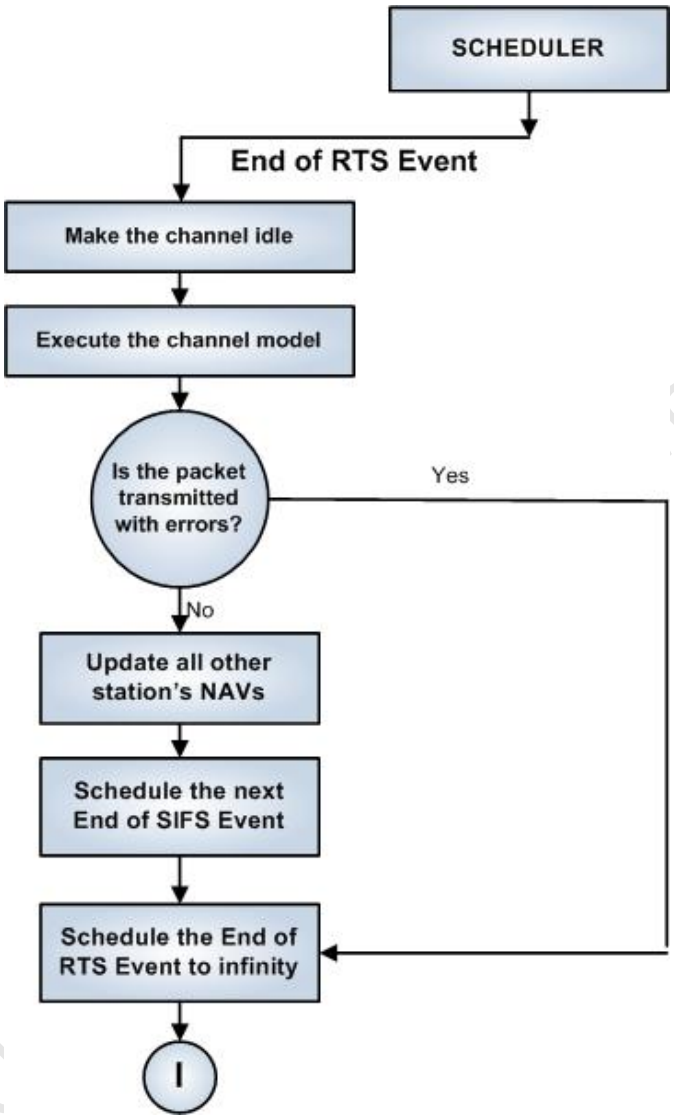

Figure 5.8: DCF End of RTS event process flow

The End of RTS event, shown in Figure 5.8, signals the end of an RTS frame transmission. It is only scheduled when the size of the packet to be transmitted is greater than the RTS Threshold.

The event first runs the channel model, passing the RTS frame size as the parameter. If the frame is received with error, the event simply schedules itself to infinity.

The event updates all the other STAs NAVs to highlight the duration for which the channel will be busy. The duration will be the time it takes to transmit the RTS frame plus the time it takes to transmit the data packet. The NAVs will contain the time at which transmission ends. This corresponds to the current system time plus the transmission duration. It then schedules the next End of SIFS event and then schedules itself to infinity.

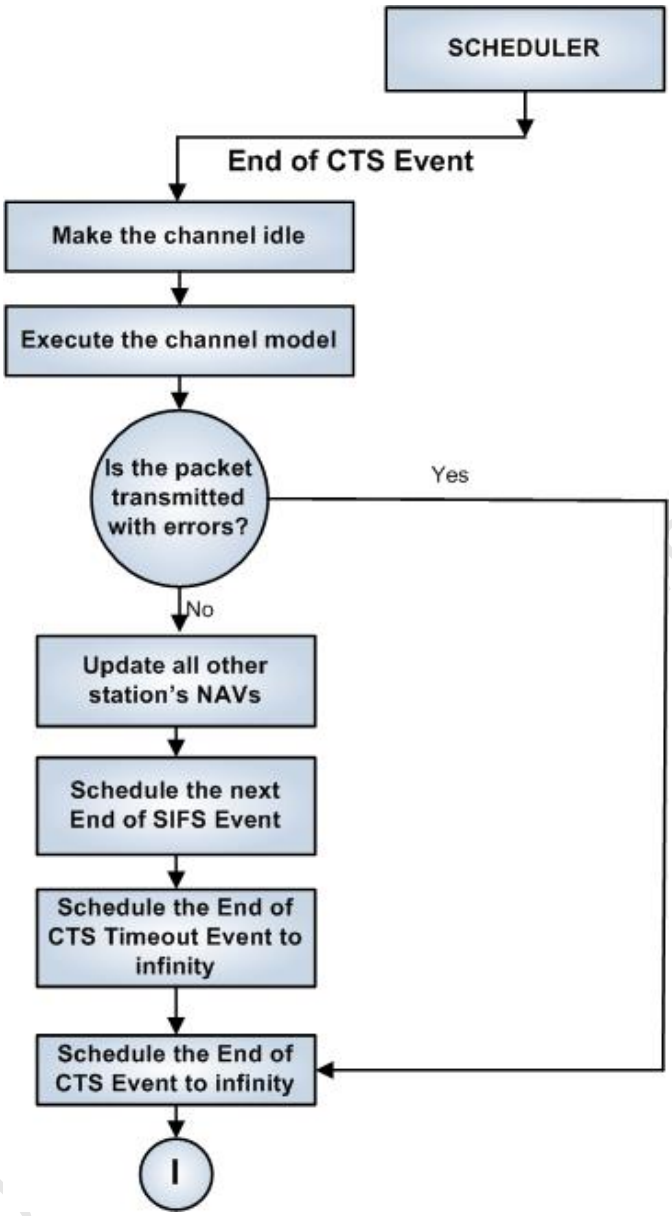

Figure 5.9: DCF End of CTS event process flow

5.1.8 **End of CTS**

The End of CTS event, illustrated in Figure 5.9, is similar to the End of RTS event.

The event first runs the channel model, passing the CTS frame size as the parameter. If the frame is received with error, the event simply schedules itself to infinity.

It is responsible for updating the NAVs of all the other STAs to show the duration for which the channel will be busy transmitting a data packet. The duration will be the time it takes to transmit a data packet. It calculates the time at which transmission ends, compares it to the current value in the NAV, and if this new value is greater, the NAV is updated; otherwise the NAV is left unchanged.

It then schedules the next End of SIFS event and then schedules itself and the End of CTS Timeout event to infinity. The End of CTS Timeout event cannot occur if the CTS frame is received successfully.

5.1.9 End of ACK

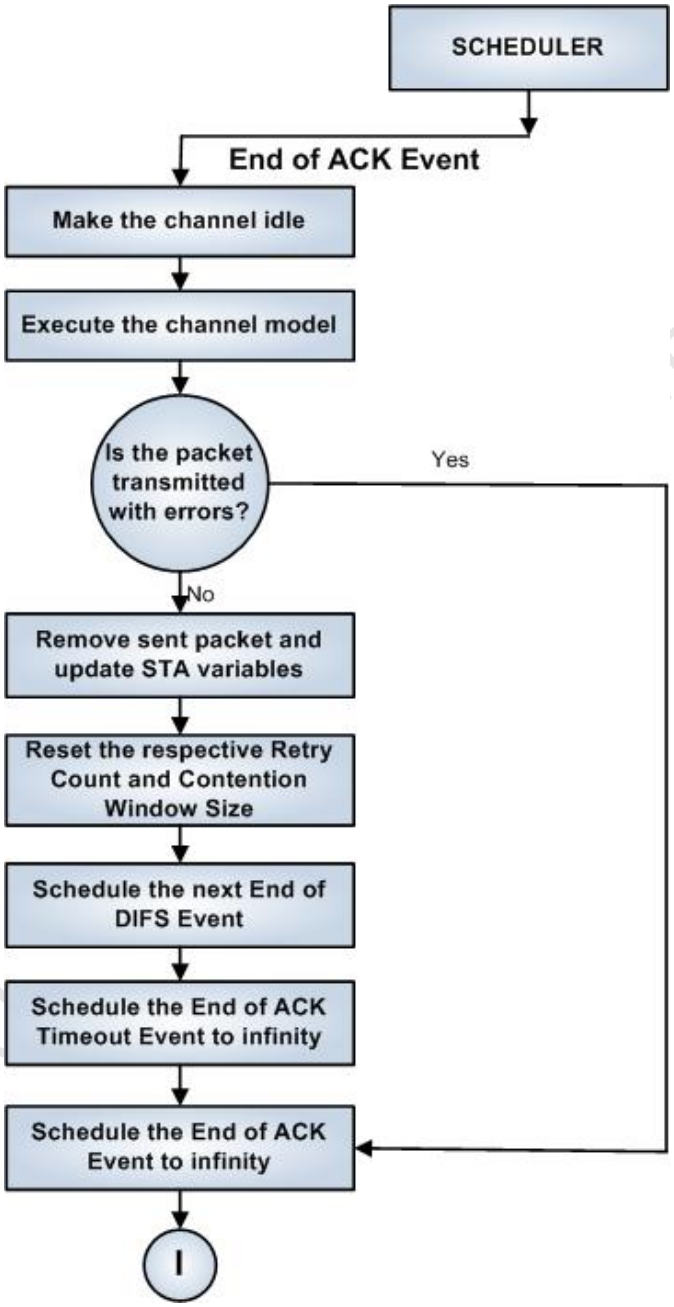

Figure 5.10: DCF End of ACK event process flow

The End of ACK event, shown in Figure 5.10, signals the end of a successful transmission through the receipt of an ACK frame by the sending STA.

The event first runs the channel model, passing the ACK frame size as the parameter. If the frame is received with error, the event simply schedules itself to infinity.

If the frame is received successfully, it resets the respective retry count as well as the contention window size. This is because when a STA fails to transmit a packet due to collision or channel failure, it doubles its contention window size and increments its retry counts, depending on the access scheme used (if basic access scheme is used, it increments the short retry count; otherwise it increments the long retry count). If packet transmission is successful, these variables have to be reset to their original values; that is, the contention window size is reset to the minimum contention window size, and the retry counts are reset to zero.

This event then schedules the next End of DIFS event before scheduling itself to infinity. It also sets the End of ACK Timeout event to infinity since it cannot occur upon successful reception of the ACK frame.

5.1.10 **End of ACK Timeout**

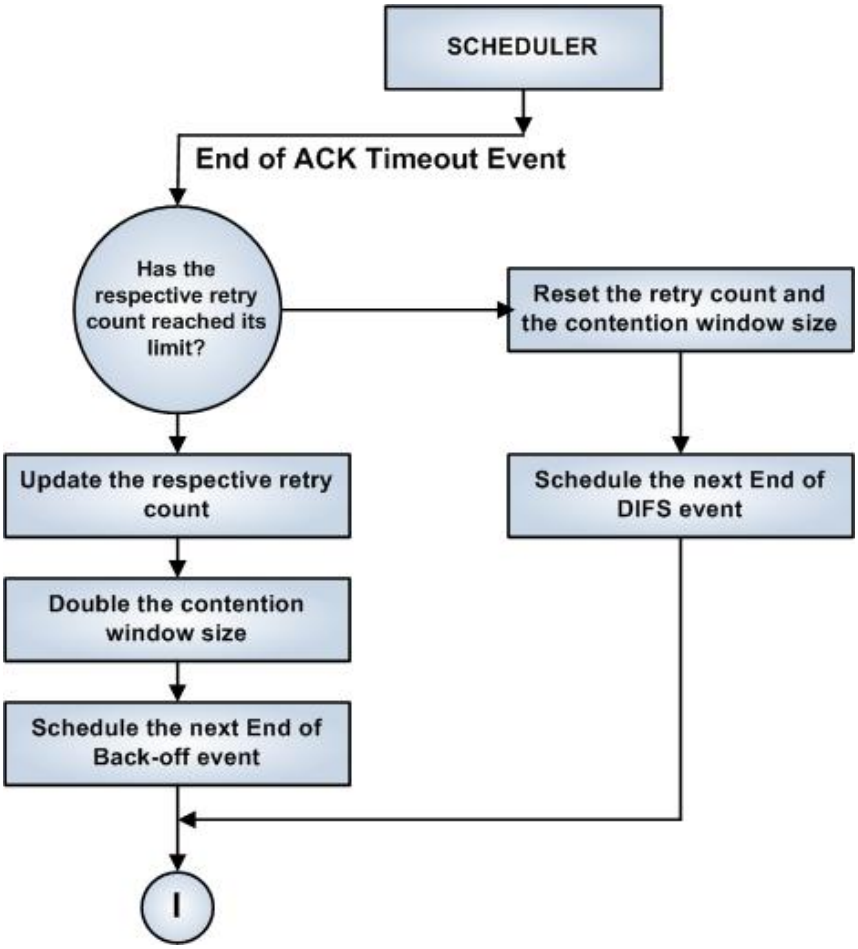

Figure 5.11: DCF End of ACK timeout event process flow

The End of ACK Timeout event, illustrated in Figure 5.11, signals the failure of the sending STA receiving an ACK frame. This may mean that there was an error in packet transmission. The sending STA will have to re-transmit its packet again. This event updates the short retry count (for basic access) or the long retry count (for RTS/CTS), as well as doubling the contention window size.

If the short retry count or long retry count are less than or equal to their respective limits after the update, it schedules the next End of Back-off event to allow the STA to re-transmit its packet, and then schedules the next

End of EIFS for all other STAs. However, if the short retry count and long retry count limits have been reached prior to the update, the packet is discarded, and the next End of DIFS event is scheduled.

5.1.11 End of CTS Timeout

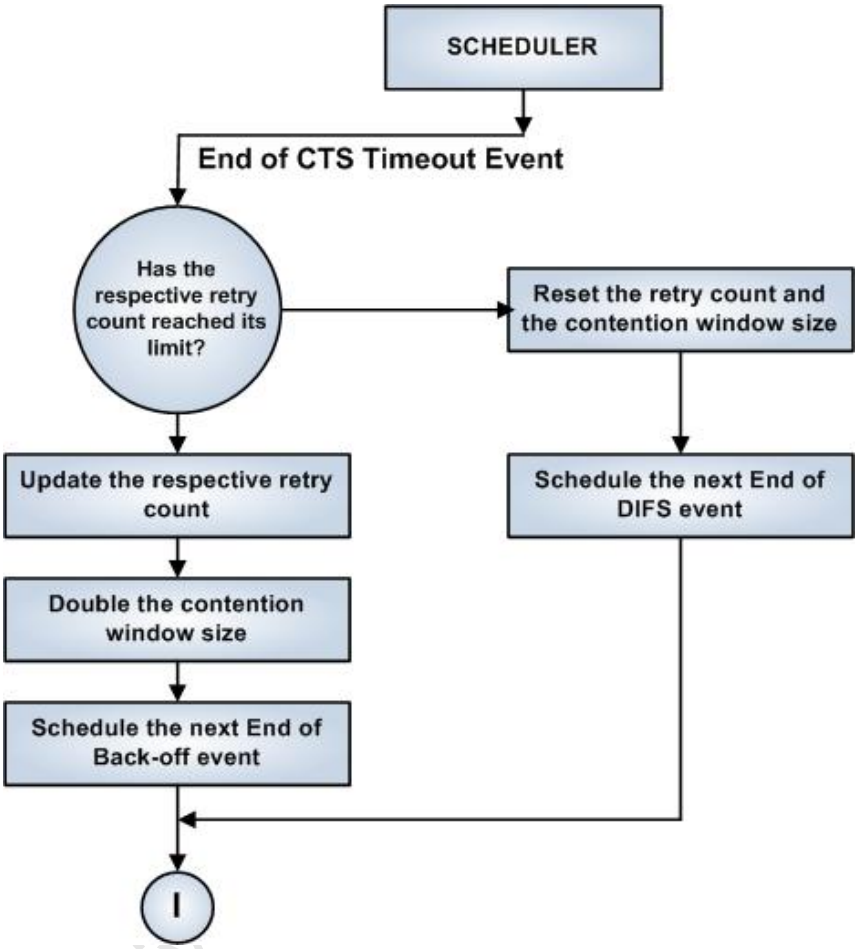

Figure 5.12: DCF End of CTS timeout event process flow

The End of CTS Timeout event, shown in Figure 5.12, is similar to the End of ACK Timeout event, and it signals the failure of the sending STA receiving a CTS frame. This may mean that there was an error in the RTS frame transmission or CTS frame transmission. The sending STA will have to re-transmit its packet again. This event updates the short retry count (for basic access) or the long retry count (for RTS/CTS), as well as doubling the contention window size.

If the short retry count or long retry count are less than or equal to their respective limits after the update, it schedules the next End of Back-off event to allow the STA to re-transmit its packet, and then schedules the next End of EIFS for all other STAs. However, if the short retry count and long retry count limits have been reached prior to the update, the packet is discarded, and the next End of DIFS event is scheduled.

5.1.12 Observation

The Observation event is responsible for recording the throughput for a fixed time interval. It simply calculates the throughput, adds it to a cumulative variable, and then increments the number of observation windows that have

occurred thus far.

5.1.13 End of Simulation

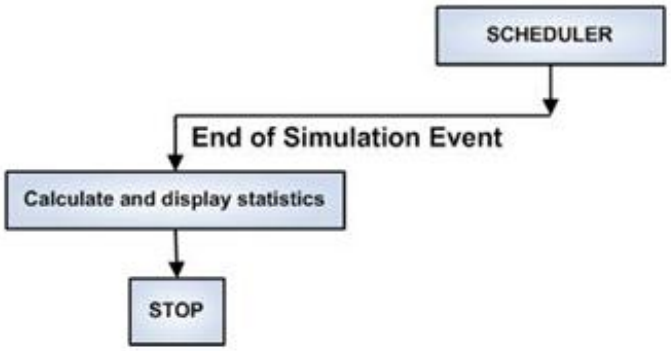

Figure 5.13: DCF End of simulation event process flow

The End-of-Simulation event, shown in Figure 5.13, is activated when the system reaches the specified time for the end of simulation. For each STA, the number of packets sent is printed. The simulator then calculates the average throughput from the observations and prints it out to screen, along with the total number of packets transmitted in the network. The average throughput is calculated by dividing the cumulative throughput sum by the number of observation windows that occurred throughout the simulation run.

5.1.14 Testing

The testing of the machine model is included in this work for completeness. It is carried out in order to verify that the model is functioning as it should. Path testing is carried out in order to see if the simulator is following the correct paths, and whether the scheduler is selecting the next event to be executed correctly.

The focus of these tests is to check that the correct execution is carried out as outlined in the design. For each event type, major execution paths are selected and the state of the system before and after the execution of each event type analysed to show that the correct paths are taken. Each event type test is only passed if all the major paths are traversed correctly. Some exceptions are made where paths identified to be highly unlikely to occur are left out of the testing as conditions necessary for the paths to execute are difficult to obtain at run time.

The first step of the test is to check that the system clock is holding the correct time and advancing the time correctly as well. It also checks to see if the event list is holding the correct times and whether the correct event is being picked from the list. The observation variables are also checked to see if the simulation results are being recorded. The testing then checks the execution paths to see if the events are being executed. Table 5.1 presents the testing results. The output from the test runs is provided in the Appendix Section in Appendix A.1.1.

Test Path	Details	Status
System Clock	Keeps track of the current simulation time and is update by the scheduler based on the Events List.	PASS
Event List	Keeps track of all events and their associated time of activation. This variable is updated by event handlers.	PASS
Observation Variables	These are a set of container variables to keep track of successful packets sent, ACKS received, collisions, throughput, bit error rate, dropped packets, and access delay.	PASS
Observation	All observations were recorded periodically and results presented at the end of each simulation run.	PASS
Packet Arrival	Detailed flow diagram for this test is provided in Figure 5.2 and the output can be found in Figure A.1.	PASS
End of DIFS	Detailed flow diagram for this test is provided in Figure 5.3 and the output can be found in Figure **??**.	PASS
End of Back-off	Detailed flow diagram for this test is provided in Figure 5.5 and the output can be found in Figure A.3.	PASS
End of Transmission	Detailed flow diagram for this test is provided in Figure 5.6 and the output can be found in Figure A.4: Event Number 12.	PASS
End of SIFS	Detailed flow diagram for this test is provided in Figure 5.7 and the output can be found in Figure A.5.	PASS
End of ACK	Detailed flow diagram for this test is provided in Figure 5.10 and the output can be found in Figure A.6.	PASS
End of RTS	Detailed flow diagram for this test is provided in Figure 5.8 and the output can be found in Figure A.7.	PASS
End of CTS	Detailed flow diagram for this test is provided in Figure 5.9 and the output can be found in Figure A.8.	PASS
End of EIFS	This was not tested as it is a rare occurrence, but a walkthrough of the code was done to check for logical correctness of the implementation. Also observations of when this events occurrence time e was updated was carried out. A detailed flow diagram is provided in Figure 5.4.	N/A
End of ACK Timeout	Detailed flow diagram for this test is provided in Figure 5.11 and the output can be found in Figure A.10	PASS
End of CTS Timeout	Detailed flow diagram for this test is provided in Figure 5.12 and the output can be found in Figure A.11	PASS
End of Simulation	Detailed flow diagram for this test is provided in Figure 5.13 and the output from this event can be found in Figure A.9	PASS

Table 5.1: Path testing results for DCF

5.2 802.11E EDCA

The 802.11e EDCA protocol, like the DCF, is implemented using the event-driven paradigm, with events discussed in Section 4.3.1. Figure 5-4 is an illustration of the EDCA simulator, showing how all the components are associated.

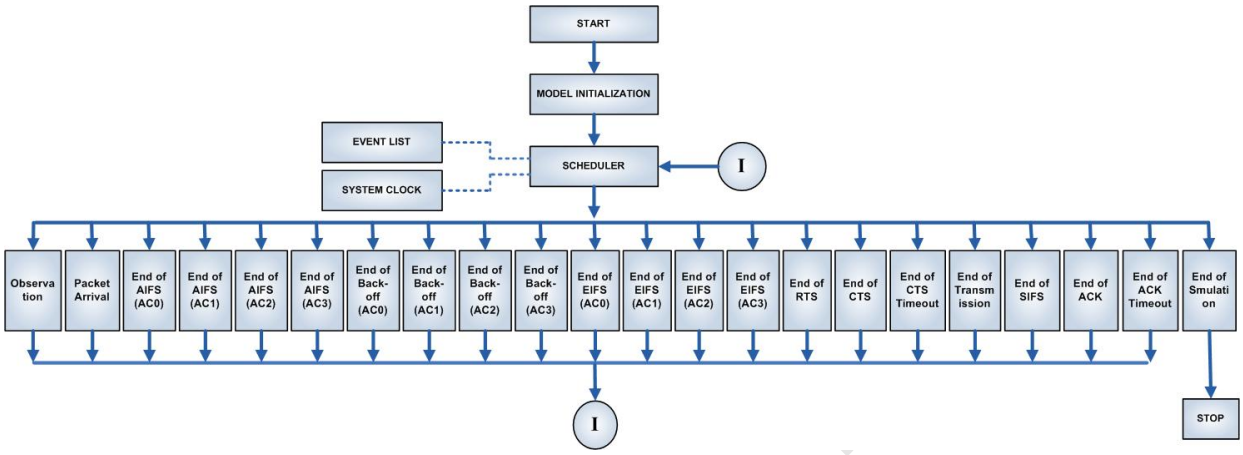

Figure 5.14: EDCA process flow chart

5.2.1 Packet Arrival

The Packet Arrival event, shown in Figure 5.15, just like the DCF Packet Arrival event, facilitates the arrival of packets into STAs buffers. The difference is that an EDCA STA has four packet buffers, one for each AC. Packet arrivals are independent for each STA; therefore multiple STAs can simultaneously receive data packets.

At initialisation, the simulator schedules a packet arrival for each buffer in each STA. The Packet Arrival event is always activated when a data packet is ready to enter one of the memory buffers of a STA. First of all, it checks to see all the buffers Pending Packet next arrival time to see which buffer and which STA is supposed to have a packet arrival at that time. Once the buffer is acquired, it the then checks to see if that particular buffers memory buffer can accommodate the pending packet. If the memory buffer is full, the packet is discarded. If there is space in the buffer, the event checks the pending packet size to see if it is greater than the packet size threshold and if it is, the packet is fragmented before being storing each fragment at the end of the buffer; otherwise, it is stored at the end of the buffer as one frame.

The event then gets the next packet inter-arrival time and packet size for the buffer, updates the Pending Packet next arrival time and packet size for the buffer, and then schedules the next Packet Arrival event. The next packet arrival time for a buffer is determined by adding the inter-arrival time to the current system time. The next Packet Arrival event is scheduled by going through all buffers in all STAs pending packet arrival times to see which has the least occurrence time.

5.2.2 End of AIFS

Since an EDCA specifies four ACs, each AC is treated independently, and therefore each ACs channel contention is oblivious to the other ACs in the STA. Each AC has its own AIFS time, and in order to model this as correctly and as efficient as possible, there is need to have an End-of-AIFS event for each AC.

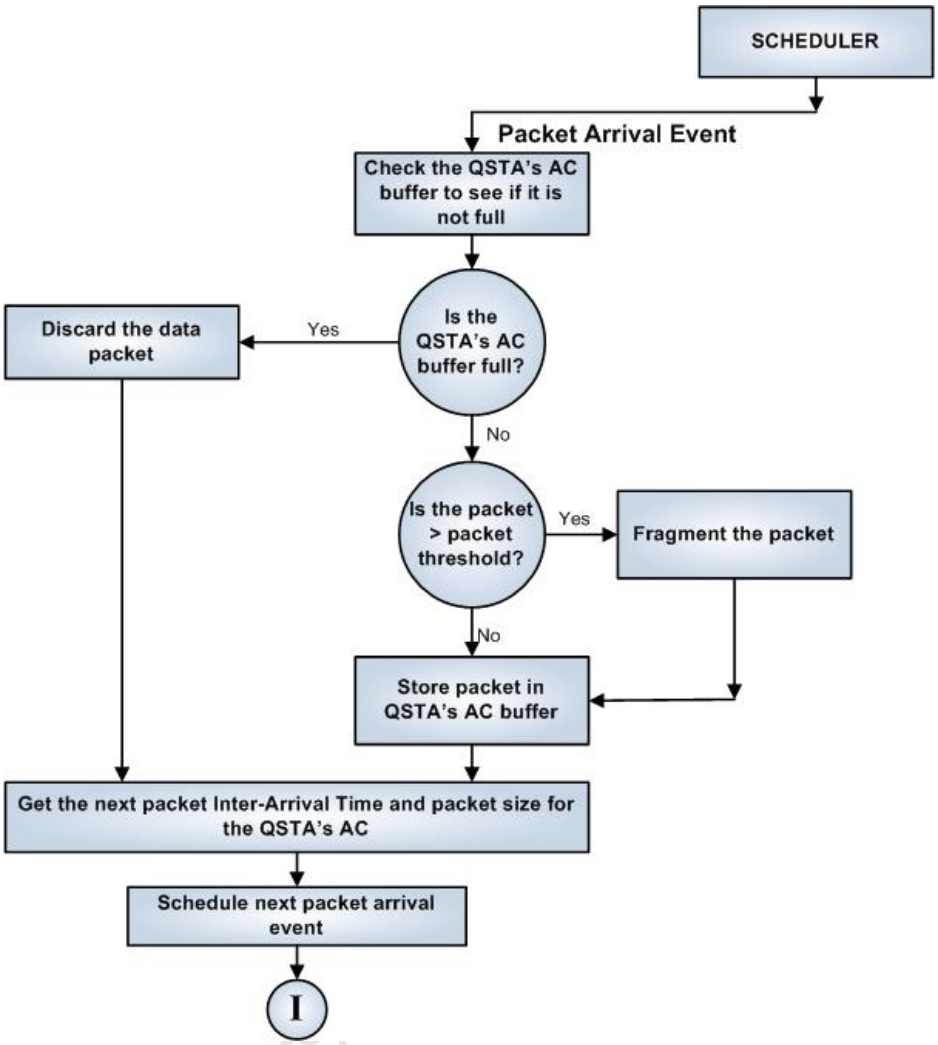

Figure 5.15: EDCA Packet arrival event process flow chart

In the system, there are four End-of-AIFS events, *End-of-AIFS-AC0, End-of-AIFS-AC1, End-of-AIFS-AC2*, and *End-of-AIFS-AC3*. These events are similar; therefore in this discussion a general discussion of the End-of-AIFS procedure is presented.

The End-of-AIFS event, shown in Figure 5.16, is activated when the channel has been idle for an AIFS interval, depending on the AC. This event identifies all the STAs buffer that has data packets to transmit for the particular AC. It then marks that buffer to show that it is ready to transmit.

After going through all the STAs buffers for the AC, the event goes through all the STAs that can transmit to find the smallest back-off counter, and then schedules the End-of-Back-off event for the particular AC using the smallest back-off value. The next End-of-Back-off event is set as the current system time plus the smallest back-off value identified.

The event then schedules itself to infinity to avoid looping in the same event infinitely.

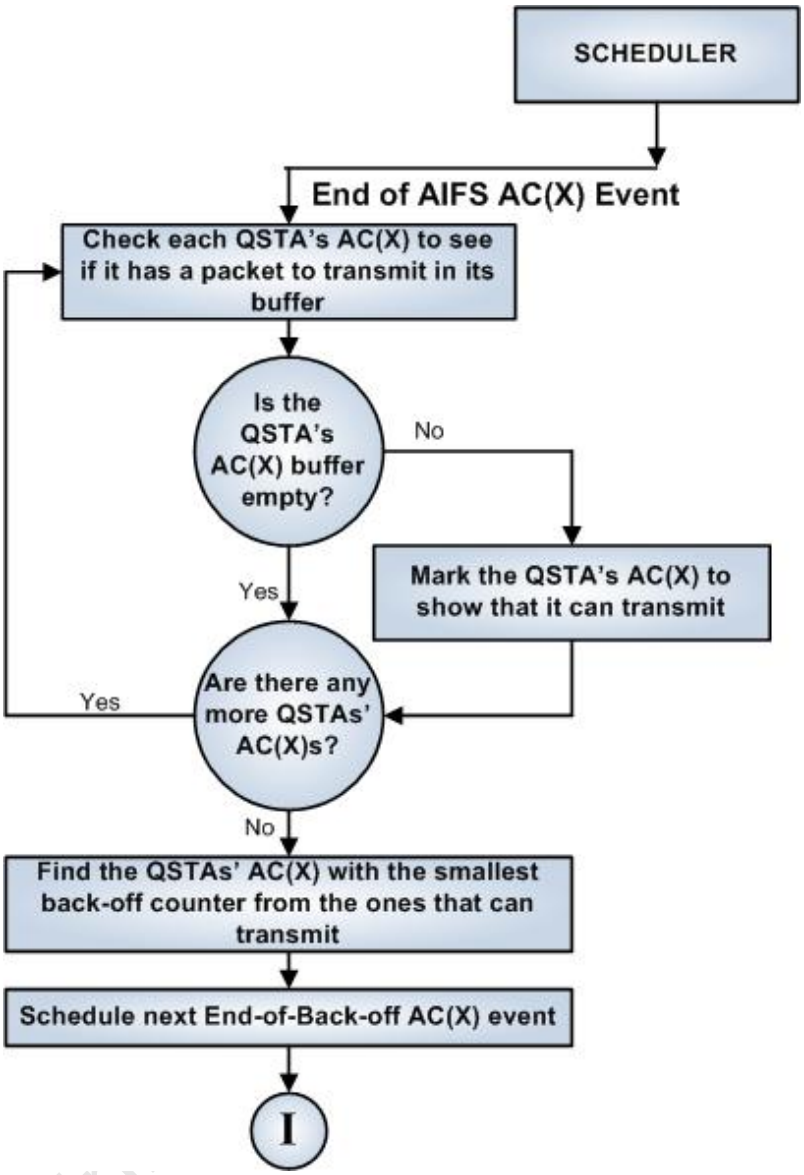

Figure 5.16: EDCA End of AIFS event process flow chart

5.2.3 **End of EIFS**

The End-of-EIFS event, shown in Figure 5.17, works exactly the same as the End-of-AIFS event. The difference is in how it is scheduled. When collision is being resolved, all other STAs ACs that were not involved in the collision have to listen to the channel for an EIFS interval before they can contend for the channel.

The End-of-EIFS event is activated when the channel has been idle for an EIFS interval, depending on the AC. This event identifies all the STAs buffer that has data packets to transmit for the particular AC. It then marks that buffer to show that it is ready to transmit.

After going through all the STAs buffers for the AC, the event goes through all the STAs that can transmit to find the smallest back-off counter, and then schedules the End-of-Back-off event for the particular AC using the smallest back-off value. The next End-of-Back-off event is set as the current system time plus the smallest back-off

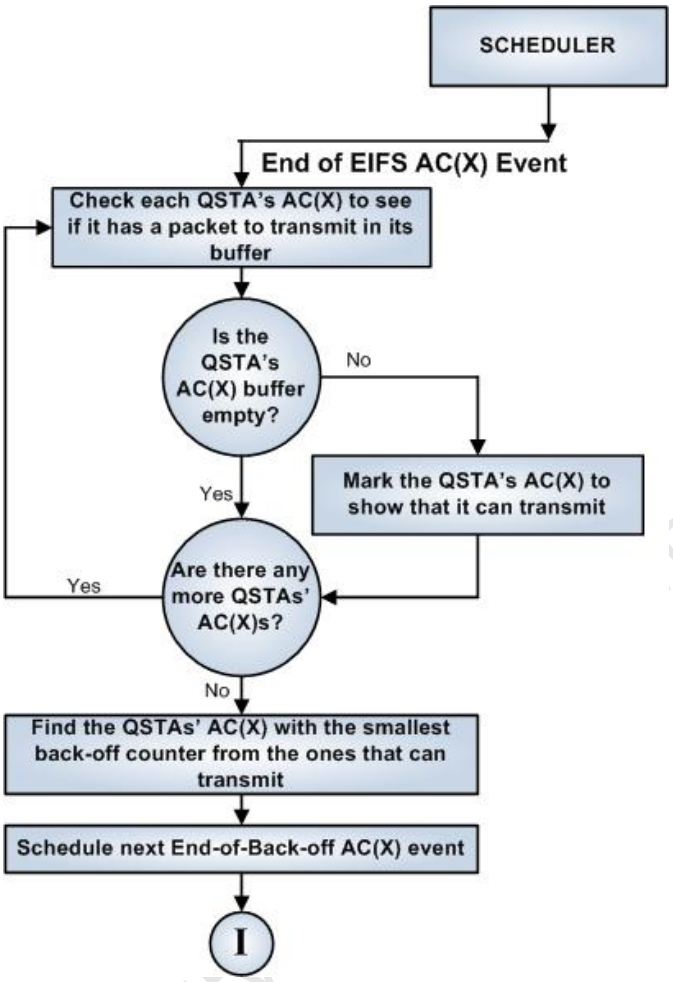

Figure 5.17: EDCA End of EIFS event process flow chart

value identified.

The event then schedules itself to infinity to avoid looping in the same event infinitely.

5.2.4 **End of Back-off**

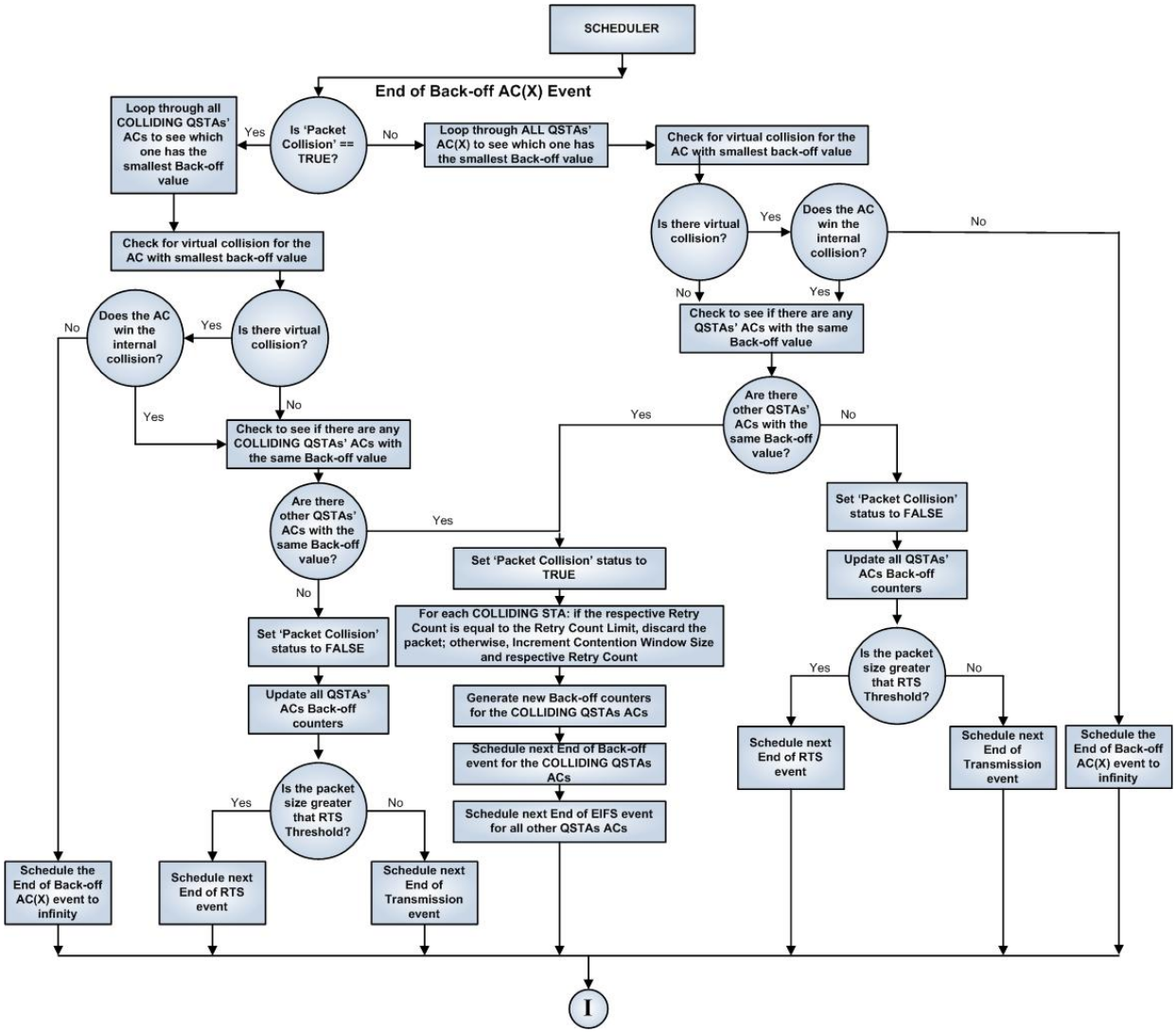

Figure 5.18: EDCA End of Back-off event process flow chart

As mentioned in Section 5.2.2, each AC is treaded independently, therefore, four End-of-Back-off events are specified, one for each AC. Since the events are similar, a general discussion of this event is presented. The End of Back-off event is illustrated in Figure 5.18.

When the End of Back-off event starts executing, it first checks to see if it is resolving a packet collision. It then checks all STAs ACs back-off counters to see if there are colliding STAs. When STAs collide, they double their contention window sizes and generate new back-off counters to contend for the transmission channel again. If there are colliding STAs contending for the network, it only deals with these STAs; otherwise, it deals with all the STAs.

When dealing with colliding STAs, the event loops through all the colliding STAs ACs to find the one with the smallest back-off value. When the AC is identified, virtual or internal collision is handled. This checks to see if there are ACs of higher priority that are also going to transmit at the same time within the STA. The AC can only transmit if there are no ACs of higher priority in the STA that are going to transmit at the same time as the AC in question.

If the AC loses the virtual collision, nothing else can be done, and the End-of-Back-off event for the AC is set to infinity, otherwise, the event checks for internal collision.

It checks to see if there are other colliding STAs ACs with the same back-off value, and within each STA, virtual collision is handled so that there can only be ACs that have won their own virtual collision. The same procedure is followed when it is not resolving collisions. In this case though, it loops through ALL STAs ACs to find the smallest back-off value. It then checks to see if there are other STAs ACs with the same back-off value. If at least two STAs ACs have this back-off value, they will acquire the channel and try to transmit at the same time, resulting in packet collision.

If there are other STAs with the same back-off value, it sets the Packet Collision status to TRUE. This is a status variable that shows whether collision has occurred or not. It then loops through all the involved STAs ACs to update their retry counts and contention window sizes. Each STAs AC has two retry counts; Short Retry Count and Long Retry Count. The Short Retry Count is incremented whenever the size of the packet to be transmitted is less than or equal to the RTS Threshold and the Long Retry Count is incremented when the packet size is greater than the RTS Threshold. There are retry count limits, which, if reached, the packet will be discarded, and the respective retry count and the contention window size are reset. The contention window size is doubled using the exponential back-off rules, and, if it reaches the maximum window size, it will not change until it is reset (through successful packet transmission or retry count limit reached). After updating the retry counts and contention window sizes, new back-off values for the respective colliding STAs are generated. The event the schedules the next End of Back-off event for the colliding STAs, and the End of EIFS event for the other STAs as well.

If there are no other STAs ACs with the smallest back-of value, it means that only one STA is able to transmit; therefore, this event sets the Packet Collision status to FALSE and updates all the STAs AC back-off counters. It then checks to see whether the basic access or the RTS/CTS scheme is used. This is achieved by comparing the respective packet size to the RTS Threshold. If the packet size is less than or equal to this threshold, the basic access scheme is used, and the event schedules the next End of Transmission event to the time of system clock plus the time it takes to transmit the packet, and t also schedules the End of ACK Timeout event; otherwise, it uses the RTS/CTS scheme and schedules the next End of RTS event, as well as the End of CTS Timeout event.

5.2.5 End of Transmission

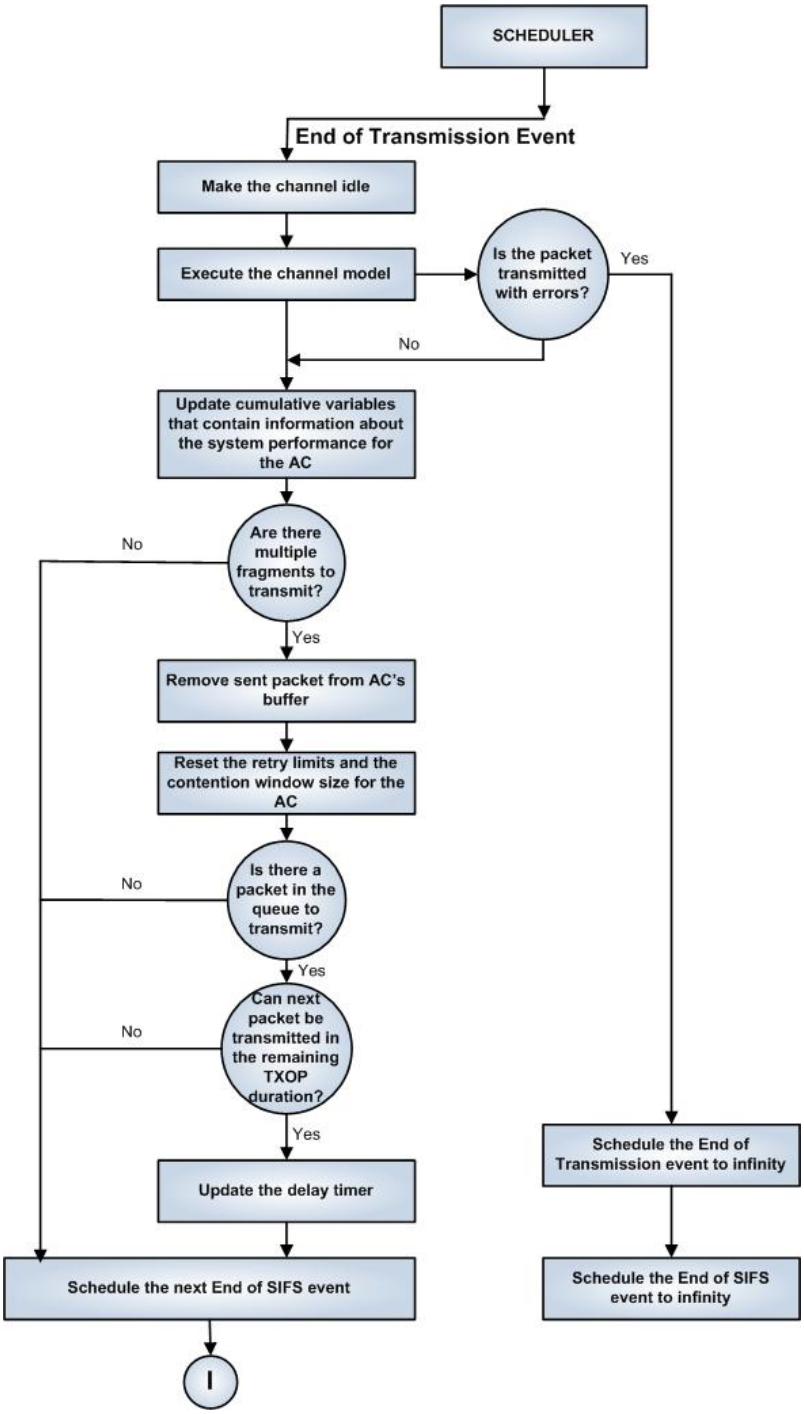

Figure 5.19: EDCA End of Transmission event process flow chart

This event, shown in Figure 5.19, is activated when transmission of a data packet is complete. The event first changes the channel status of the transmission medium to *idle*. It then runs the channel model, passing the packet size as a parameter. It executes the channel model to determine whether the packet was transmitted with error or not. If the packet was transmitted with error, it schedules itself and the End-of-SIFS event to infinity. Otherwise, it updates the system statistics in the current observation window, such as the number of packets transmitted so far

for the STAs AC and the number of bits that have been transmitted in the network.

It then checks to see if multiple fragments are being sent. If the system is sending multiple fragments, it removes the packet that has just been transmitted from the Acs buffer and resets the respective retry count and contention window size. It then checks to see if there is another packet to be transmitted and if so, it also checks to see if the packet can be transmitted in the remaining TXOP duration. If this is the case, the delay timer is updated.

The event then schedules the next End of SIFS event to the current system clock time plus the SIFS interval.

5.2.6 **End of SIFS**

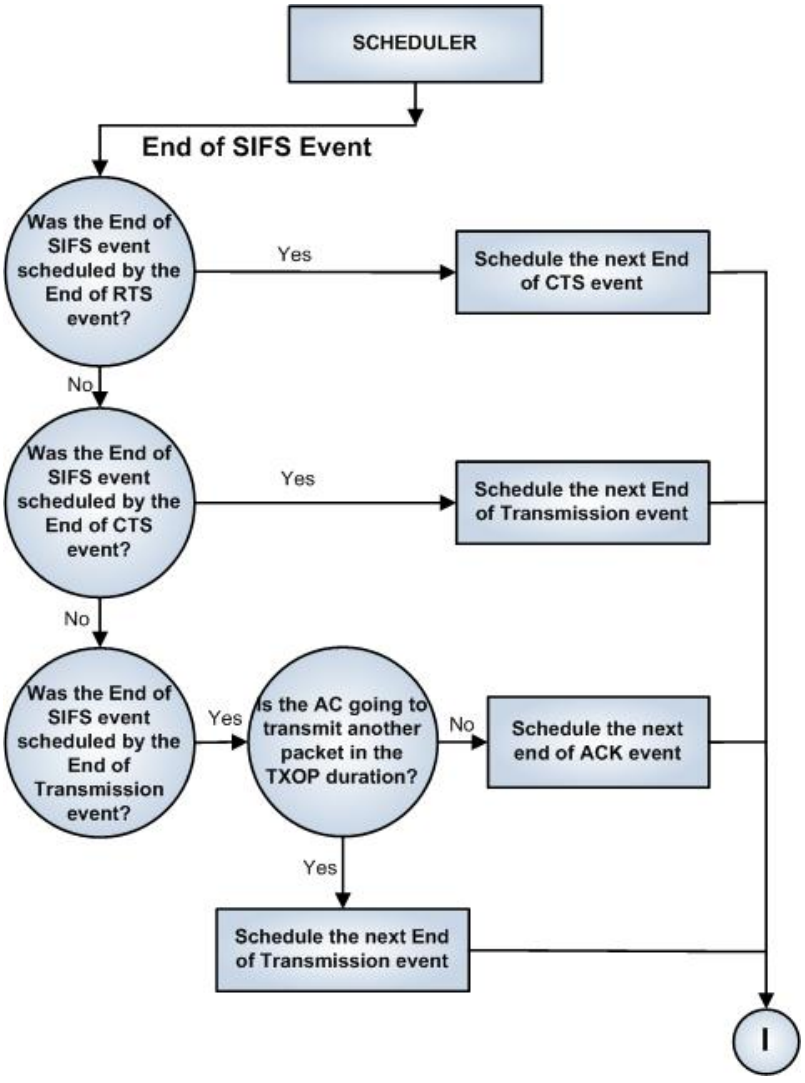

Figure 5.20: EDCA End of SIFS event process flow chart

The End of SIFS event, shown in Figure 5.20, can be scheduled by three events; *End of RTS, End of CTS*, and *End of Transmission*.

If it is scheduled by the End of RTS event, it schedules the next End of CTS event, if it is scheduled by the End of CTS event; it schedules the next End of Transmission event, and if it is scheduled by the End of Transmission event,

first checks to see if there is another packet to transmit. If so, it then checks to see if the packet can be transmitted in the remaining TXOP duration. If a packet can be sent in the remaining TOP duration, it schedules the next End of Transmission event using the next packet size as a parameter to determine the transmission duration, which means that the transmission of a fragment takes place. Otherwise, it schedules the next End of ACK event.

The channel status is then set to TRUE to signal packet or frame transmission.

5.2.7 **End of RTS**

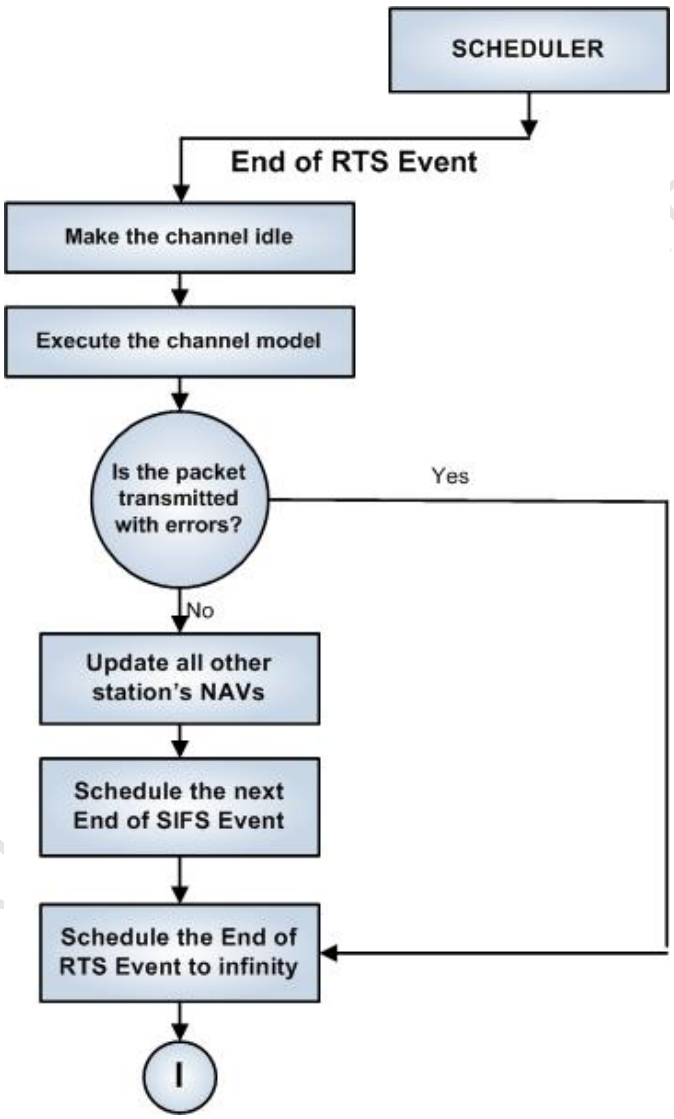

Figure 5.21: EDCA End of RTS event process flow chart

The End of RTS event, shown in Figure 5.21, is similar to the DCF End-of-RTS event, discussed in Section 5.1.7, and it signals the end of an RTS frame transmission. It is only scheduled when the size of the packet to be transmitted is greater than the RTS Threshold.

The event first runs the channel model, passing the RTS frame size as the parameter. If the frame is received with error, the event simply schedules itself to infinity.

The event updates all the other STAs NAVs to highlight the duration for which the channel will be busy. The duration will be the time it takes to transmit the RTS frame plus the time it takes to transmit the data packet. The NAVs will contain the time at which transmission ends. This corresponds to the current system time plus the transmission duration. It then schedules the next End of SIFS event and then schedules itself to infinity.

5.2.8 **End of CTS**

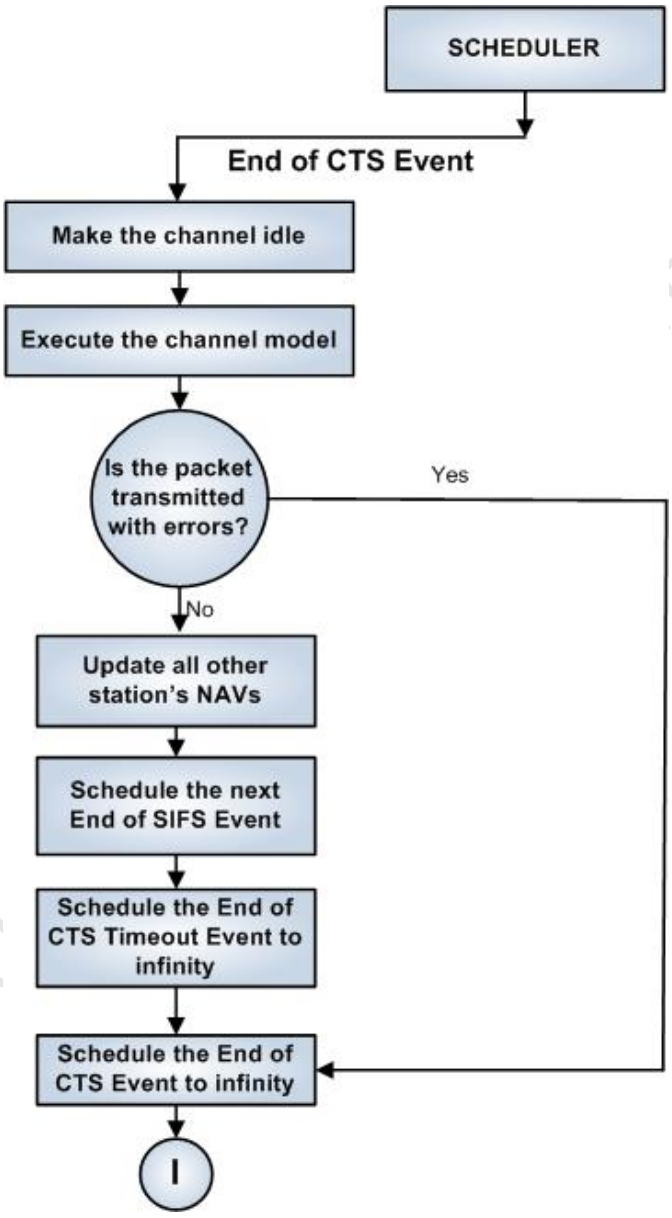

Figure 5.22: EDCA End of CTS event process flow chart

The End-of-CTS event, illustrated in Figure 5.22, is similar to the End-of-RTS event.

The event first runs the channel model, passing the CTS frame size as the parameter. If the frame is received with error, the event simply schedules itself to infinity.

It is responsible for updating the NAVs of all the other STAs to show the duration for which the channel will be

busy transmitting a data packet. The duration will be the time it takes to transmit a data packet. It calculates the time, at which transmission ends, compares it to the current value in the NAV, and if this new value is greater, the NAV is updated; otherwise the NAV is left unchanged.

It then schedules the next End of SIFS event and then schedules itself and the End-of-CTS-Timeout event to infinity. The End-of-CTS-Timeout event cannot occur if the CTS frame is received successfully.

5.2.9 End of ACK

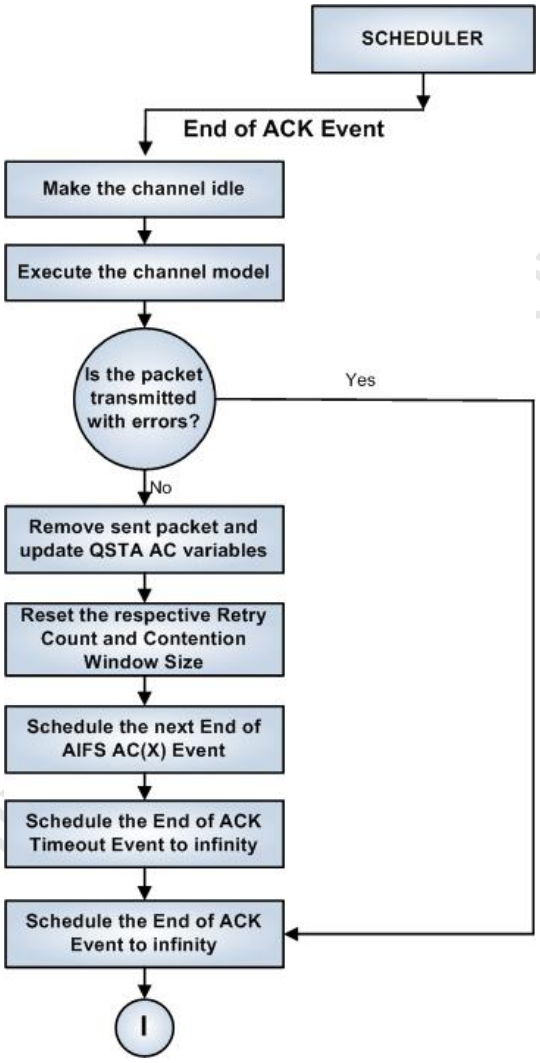

Figure 5.23: EDCA End of ACK event process flow chart

The End-of-ACK event, shown in Figure 5.23, signals the end of a successful transmission through the receipt of an ACK frame by the sending STA.

It first runs the channel model, passing the ACK frame size as the parameter. If the frame was received with error, the event simply schedules itself to infinity.

If the ACK frame was received successfully, it resets the respective retry count as well as the contention window size for the concerned AC. This is because when a STAs AC fails to transmit a packet due to collision or channel

failure, it doubles its contention window size and increments its retry counts, depending on the access scheme used (if basic access scheme is used, it increments the short retry count; otherwise it increments the long retry count). If packet transmission is successful, these variables have to be reset to their original values; that is, the contention window size is reset to the minimum contention window size, and the retry counts are reset to zero.

A new back-off counter value for the AC is generated.

The event then schedules the next End-of-AIFS events for all the ACs before scheduling itself to infinity. It also sets the End-of-ACK-Timeout event to infinity since it cannot occur upon successful reception of the ACK frame.

5.2.10 End of ACK Timeout

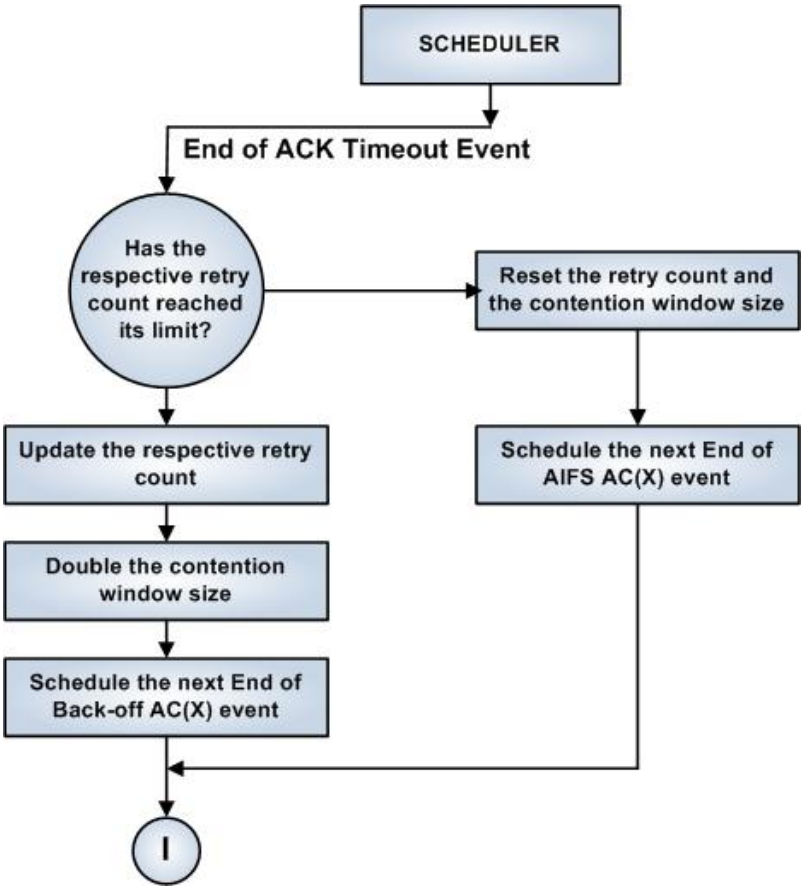

Figure 5.24: EDCA End of ACK timeout event process flow chart

The End of ACK Timeout event, illustrated in Figure 5.24, signals the failure of the sending STA receiving an ACK frame. This may mean that there was an error in packet transmission. The sending STAs AC will have to re-transmit its packet again. This event updates the short retry count (for basic access) or the long retry count (for RTS/CTS), as well as doubling the contention window size.

If the short retry count or long retry count are less than or equal to their respective limits after the update, it schedules the next End of Back-off event for the AC to allow it to re-transmit its packet, and then schedules the next End-of-EIFS events for all other STAs. However, if the short retry count and long retry count limits have been reached prior to the update, the packet is discarded, and the next End-of-AIFS events are scheduled.

5.2.11 End of CTS Timeout

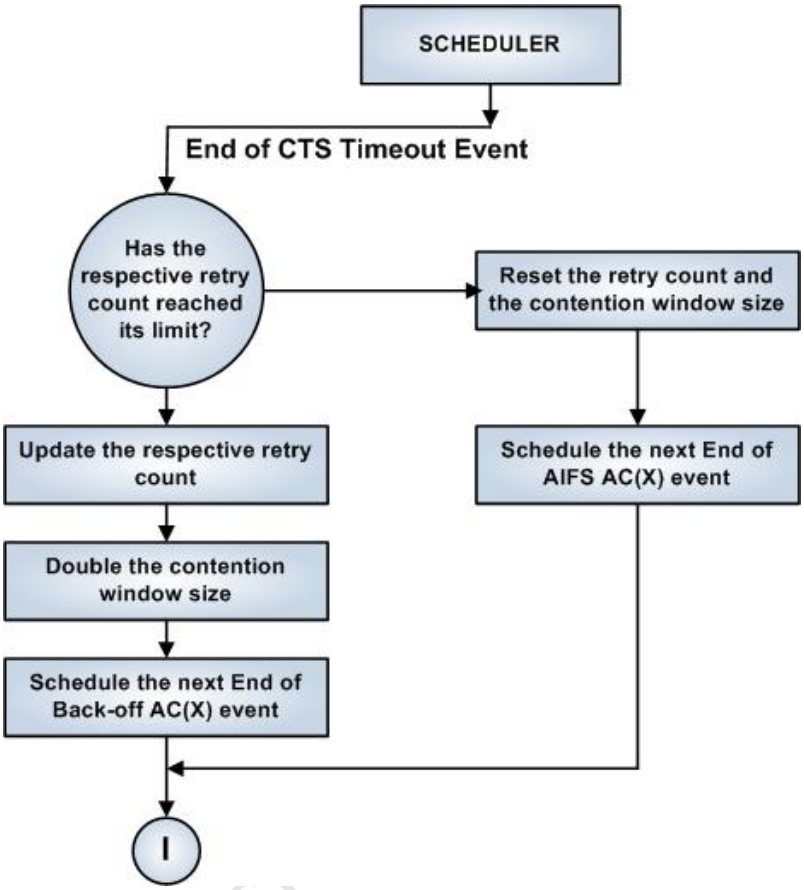

Figure 5.25: EDCA End of CTS timeout event process flow chart

The End of CTS Timeout event, shown in Figure 5.25, is similar to the End of ACK Timeout event, and it signals the failure of the sending STA receiving a CTS frame. This may mean that there was an error in the RTS frame transmission or CTS frame transmission. The sending STAs AC will have to re-transmit its packet again. This event updates the short retry count (for basic access) or the long retry count (for RTS/CTS), as well as doubling the contention window size.

If the short retry count or long retry count are less than or equal to their respective limits after the update, it schedules the next End-of-Back-off event for the AC to allow it to re-transmit its packet, and then schedules the next End-of-EIFS events for all other STAs. However, if the short retry count and long retry count limits have been reached prior to the update, the packet is discarded, and the next End-of-AIFS events are scheduled.

5.2.12 Observation

The Observation event is responsible for recording the throughput for a fixed time interval. It simply calculates the throughput, adds it to a cumulative variable, and then increments the number of observation windows that have occurred thus far.

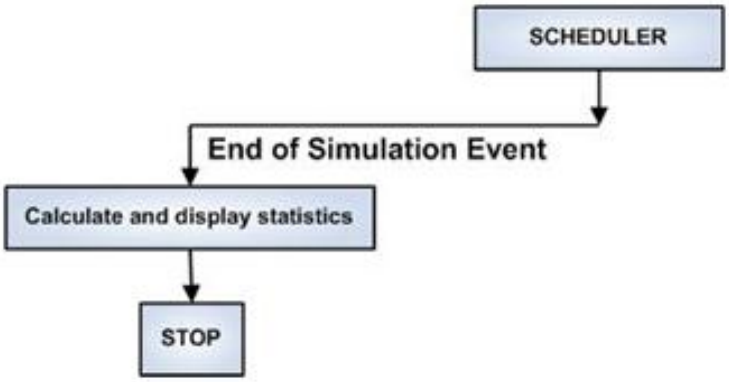

Figure 5.26: EDCA End of Simulation timeout event process flow chart

5.2.13 **End of Simulation**

The End-of-Simulation event, shown in Figure 5.26, is activated when the system reaches the specified time for the end of simulation. For each STA, the number of packets sent is printed. The simulator then calculates the average throughput from the observations and prints it out to screen, along with the total number of packets transmitted in the network. The average throughput is calculated by dividing the cumulative throughput sum by the number of observation windows that occurred throughout the simulation run.

5.2.14 **Testing**

Testing for the EDCA protocol follows the same pattern as that of DCF. Testing results are provided in Table 5.2. The output from the test runs is provided in the Appendix Section in Appendix A.1.2.

Test Path	Details	Status
System Clock	Keeps track of the current simulation time and is update by the scheduler based on the Events List.	PASS
Event List	Keeps track of all events and their associated time of activation. This variable is updated by event handlers.	PASS
Observation Variables	These are a set of container variables to keep track of successful packets sent, ACKS received, collisions, throughput, bit error rate, dropped packets, and access delay.	PASS
Observation	All observations were recorded periodically and results presented at the end of each simulation run.	PASS
Packet Arrival	Detailed flow diagram for this test is provided in Figure 5.15 and the output can be found in A.12.	PASS
End of AIFS (AC0)	Detailed flow diagram for this test is provided in Figure 5.16 and the output can be found in Figure A.13.	PASS
End of AIFS (AC1)	Detailed flow diagram for this test is provided in Figure 5.16 and the output can be found in Figure A.14.	PASS
End of AIFS (AC2)	Detailed flow diagram for this test is provided in Figure 5.16 and the output can be found in Figure A.15.	PASS
End of AIFS (AC3)	Detailed flow diagram for this test is provided in Figure 5.16 and the output can be found in Figure A.16.	PASS
End of Back-off (AC0)	Detailed flow diagram for this test is provided in Figure 5.18 and the output can be found in Figure A.17.	PASS
End of Back-off (AC1)	Detailed flow diagram for this test is provided in Figure 5.18 and the output can be found in Figure A.18.	PASS
End of Back-off (AC2)	Detailed flow diagram for this test is provided in Figure 5.18 and the output can be found in Figure A.19.	PASS
End of Back-off (AC3)	Detailed flow diagram for this test is provided in Figure 5.18 and the output can be found in Figure A.20.	PASS
End of Transmission	Detailed flow diagram for this test is provided in Figure 5.6 and the output can be found in Figure A.24.	PASS
End of SIFS	Detailed flow diagram for this test is provided in Figure 5.7 and the output can be found in Figure A.23.	PASS
End of ACK	Detailed flow diagram for this test is provided in Figure 5.10 and the output can be found in Figure A.25.	PASS
End of RTS	Detailed flow diagram for this test is provided in Figure 5.8 and the output can be found in Figure A.21.	PASS
End of CTS	Detailed flow diagram for this test is provided in Figure 5.9 and the output can be found in Figure A.22.	PASS
End of EIFS	This was not tested as it is a rare occurrence, but a walkthrough of the code was done to check for logical correctness of the implementation. Also observations of when this events occurrence time e was updated was carried out. A detailed flow diagram is provided in Figure 5.4.	N/A
End of ACK Timeout	Detailed flow diagram for this test is provided in Figure 5.11 and the output can be found in Figure A.26.	PASS
End of CTS Timeout	Detailed flow diagram for this test is provided in Figure 5.12 and the output can be found in Figure A.27.	PASS
End of Simulation	Detailed flow diagram for this test is provided in Figure 5.13 and the output from this event can be found in Figure A.28.	PASS

Table 5.2: Path testing results for EDCA

5.3 CHANNEL MODEL

5.3.1 Channel Model Implementation

The channel model determines whether a packet or frame has been transmitted successfully or not over the wireless medium. In this section, a detailed discussion of its implementation is given, following the steps presented in Section 4.4.4.

In the simulator, the channel model is executed at every packet or frame transmission. This means that it is called during the End-of-Transmission, End-of-RTS, End-of-CTS, and End-of-ACK events. The size of the packet or frame is passed as a parameter, and it also makes use of the bit energy to noise ratio (Eb/NO), which is a user-specified integer.

Calculation of the Signal-to-Noise Ratio

The Es/NO is determined by the calculation [52]:

$$EbN0dB + 10\log\frac{nDSC}{nFFT} + 10\log\frac{DataSymbolDuration}{TotalSymbolDuration} \tag{5.1}$$

where
nDSC is the number of sub-carriers, and
nFFT is the FFT size

Generation of a Binary Number Sequence

A random generator is used to generate random bits (0s and 1s) and the bits are stored in an array. The number of bits equals the packet or frame size, so it a packet is 1000 bytes long (which equates to 8000 bits, then 8000 bits are generates to form the binary representation of the packet. The random number generator generates a number between 0 and 1 and if the value is less than 0.5, it is stored as a 0; otherwise, it is stored as a 1. This attempts to resemble the transmitted packet.

BPSK Modulation

The BPSK modulation changes all the 0s in the binary sequence to -1s and all the 1s remain as 1s in the array, using the following calculation:

$$(2 * array[x]) - 1 \tag{5.2}$$

where
array[x] denotes a bit in the array at position x

This results in a sequence of 1s and -1s.

Assigning Modulated Symbols to the Sub-Carriers

There are 52 sub-carriers specified for OFDM at 54Mbps, but the FFT size is set to 64. Therefore, all the BPSK modulated bits are assigned to each of the sub-carriers, and the remaining symbols are assigned to 0. A Reshape function is used to assign these symbols. Each sub-carrier will contain a certain number of symbols. The number of symbols is calculated by dividing the number of bits by the number of bits per symbol. This data is then stored in a 2-dimensional array with the number of sub-carriers and number of symbols as the dimensions. The symbols are assigned to subcarriers [-26 to -1, +1 to 26].

Orthogonality of Sub-Carriers using IFFT and Normalising the Transmit Power of Symbols to 1

The next step is to take the IFFT [52] of the modulated symbols in the sub-carriers in order for them to be orthogonal. The data are then normalised by multiplying each resultant symbol by the value (nFFT/sqrt(nDSC)) [52]. Normalisation brings all the data characteristics to a common scale. The IFFT result is a dataset of complex numbers.

Appending Cyclic Prefix (CP)

The CP is appended to the beginning of the data set. This is done by taking the last 16 columns in the array and appending them to the beginning of the array. This ensures that the waveform is continuous [52].

Convolving each OFDM Symbol to a 10-Tap Rayleigh Multi-path Fading Channel

The multi-path channel is executed in this step. A set of complex values corresponding to the number of symbols is created. The fading on each symbol is independent. The frequency response of the fading channel on each symbol is computed and stored. This is done by applying the FFT algorithm.

The next step is the convolution of each of each symbol with the random channel. This is a mathematical operation on two functions, a and b, that produces a third function that is viewed as a modified version of one of the original functions [52].

Concatenation of Multiple Symbols to Form a Long Transmission Sequence

The multiple symbols are then concatenated, one after the other to form a long sequence of data that will be transmitted over the medium.

Adding White Gaussian Noise

The AWGN is modelled as Gaussian complex numbers, and randomly generated values are added to the long transmission sequence. The Es/NO is used as a function to determine the severity of the noise on the transmission sequence.

Grouping the Received Vector Into Multiple Symbols

The received transmission sequence is grouped into multiple symbols.

Removing CP

The CP is removed from the data set to leave only the relevant symbols.

Converting the Symbol Received in the Same Time Domain Into Frequency Domain

This achieved by using the FFT algorithm.

Dividing the Received Symbol With The Known Frequency Response of the Channel

Each element in the received symbol is divided by its corresponding element in the frequency response that was calculated previously.

Extracting the Required Sub-Carriers

The 52 relevant sub-carriers are extracted from the data and then stored as a long sequence.

BPSK Demodulation

All -1s are converted to 0s and all 1s remain as 1s. This leaves a binary sequence of the received packet. This resembles the received packet or frame.

Counting the Number of Bit Errors

The corresponding bits from the transmitted and received packets are compared. A counter is incremented whenever there is a mismatch. If the counter is greater than 0 after the comparison, it means that the packet was received with an error, otherwise a count of zeros signals successful transmission.

5.3.2 Testing

The testing for the channel model was carried out in order to see whether the steps are followed, and whether the results are produced correctly. Figures A.29 to A.45 in Appendix A.2 show snippets of the output from the channel model. The test was done for a packet size of 52 bits. The output from the test was too large to display, so the results have been truncated. The results only show the kind of output that is produced by the channel model and there is a snippet for each step in the channel model as discussed in Section 4.4.4.

The test followed the correct execution steps and produced the correct results, therefore the test passed.

5.4 WORKLOAD MODEL

5.4.1 Workload Model Implementation

The workload model was implemented using the design discussed in Section 4.5.

- VoIP Traffic: implemented a two-state MMPP
- Video Traffic: implemented a Log-normal distributed traffic generator
- Best-Effort Traffic: implemented a Pareto IAT process

- Background Traffic: implemented a Weibull IAT process

Each of these traffic types generates its own independent traffic. Each traffic type is mapped to its corresponding AC in EDCA, but the DCF selects a traffic type at random for the next packet arrival.

5.4.2 **Testing**

Testing was carried out in order to see whether the model generates traffic as designed, and a well-detailed discussion of the testing is provided in Appendix A.3.

CHAPTER 6

EXPERIMENTATION

6.1 OVERVIEW

The purpose of this research and the experiments is to understand the performance of the IEEE 802.11e and 802.11g protocols under realistic channel and workload conditions, and to determine which of the two protocols best supports QoS-sensitive traffic.

We want to understand how networks operate under various channel conditions, and how these tend to affect overall network performance. We also want to understand how QoS-sensitive traffic is handled in such networks, and which protocol best supports this traffic.

6.2 PERFORMANCE METRICS

In order to fully understand overall network performance, we need to assess the performance using various performance metrics. We need to evaluate performance from a systems perspective, and also evaluate the performance from a users experience perspective. This is because we need to see how the systems resources are being utilised, as well as to determine how the performance from a users perspective measures up.

6.2.1 System Performance Metrics

The system performance metrics of interest are:

- **Normalised Aggregate Throughput** - this is the ratio of the actual network throughput to the net bit rate of the channel in bits/s, where the actual network throughput is the average amount of the data transmitted in the channel in bits/s.

- **Collision Probability** - this is the probability that a packet collides, and is measured as the number of packets collided against the total number of packets transmitted. The value is expressed as a probability (in the range 0-1).

6.2.2 User Performance Metric

The user experience performance metric of interest is:

- **Access Delay** this is measured as the duration from which a packet becomes available for transmission to the time it is transmitted successfully across a channel

6.3 EXPERIMENTAL DESIGN

Two sets of experiments were conducted, scalability experiments and comparison experiments. The scalability experiments show each performance metric against an increasing number of stations for a given Bit-Energy-To-Noise Ratio (Eb/NO) value, and the comparison experiments show each performance metric against an increasing number of Eb/NO values.

The following conditions are considered for the experiments:

1. The network is fully connected and all STAs are in range.
2. The network operates in realistic channel conditions; therefore, there are interference and noise from external sources and channel fading.
3. The experiments consider the IEEE 802.11g and 802.11e protocols:
   - Pure mode operation without IEEE 802.11 backward compatibility.
   - Fixed rates of 54 Mbps and 24 Mbps for data and control frames, respectively.
4. Only the basic access scheme is used in the experiments. This is because this work does not cover the comparison of the basic access and RTS/CTS schemes, but only wants to discuss the QoS enhancements as well as the effects of a realistic channel. The basic access scheme is thus sufficient for this purpose.

	Basic Access
Scalability	EXPERIMENT SET 1
Comparison	EXPERIMENT SET 2

Table 6.1: Experimentation breakdown overview

Table 6.1 shows the grouping of the experiments.

Experiment set 1 contains 4 experiments, with two EDCA experiments and two DCF experiments. The four EDCA experiments are for Bit Energy-To-Noise ratios (Eb/NO) of 30 and 50 and the same applies to the DCF experiments. Each experiment is run for an increasing number of STAs, from 1 to 16.

Experiment set 2 contains 16 experiments, 8 for EDCA and 8 for DCF. Each experiment is run for 10 STAs in the network, and for Eb/NO values of 15, 20, 25, 30, 35, 40, 45, and 50. Results are reported for Voice and Video traffic since these are QoS-sensitive traffic types.

The breakdown of the experiments is highlighted in Tables 6.2 (scalability) and 6.3 (comparison). Each experiment produces results for all the performance metrics considered.

	EDCA	DCF
Number of STAs	[1, 2, 3, 4, 5,..., 16]	[1, 2, 3, 4, 5,..., 16]
Eb/NO Values	30, 50	30, 50

Table 6.2: Scalability experiments breakdown

	EDCA	DCF
Number of STAs	10	10
Eb/NO Values	20, 25, 30, 35, 40, 45, 50	20, 25, 30, 35, 40, 45, 50

Table 6.3: Comparison experiments breakdown

6.4 MODEL PARAMETERISATION

The system requires several parameters to be specified in order to run the experiments. Table 6.4 specifies the parameter values that are used for the experiments. For the workload model, values that provided the most packets were chosen, so that there are a lot of packets in the system in order to evaluate the system performance under heavy loads.

Variables	**Parameters**
Simulation Duration	30 seconds
Number of Runs	30
Channel Model	Eb/NO: User-specified [20, 25, 30, 35, 40, 45, 50] Packet Size: Size of packet transmitted the packet sizes are generated by the workload model
VoIP Workload Model	λ_1: 9.87 λ_2: 0.55 ω_1: 0.001 ω_2: 0.034 Packet Size: 100 - 1500 bytes
Video Workload Model	D_1: 9.87 D_2: 0.55 E_1: 0.001 E_2: 0.034 Packet Size: (100 - 1500 bytes for small packets, 1500 - 8015 bytes for big packets)
Best-Effort Workload Model	λ_1: 9.87 λ_2: 0.55 ω_1: 0.001 ω_2: 0.034 Packet Size: 100 - 1500 bytes
Background Workload Model	λ_1: 9.87 λ_2: 0.55 ω_1: 0.001 ω_2: 0.034 Packet Size: 100 - 1500 bytes
Random Seed	23456789

Table 6.4: Model parameterisation for the experiments

All the simulation runs execute for 30 seconds to ensure that a lot of packets are transmitted, and for each number of STAs, 30 runs are executed in order to have normally distributed data sets. The average of the 30 runs

is calculated for each performance metric, and the confidence intervals are calculated for each of these.

In order to ensure accuracy of the results, two bias removal experiments were run; one for each protocol.

Figures 6.1 and 6.2 contain graphs of the two tests. Figure 6.1 shows the variation of the throughput in time for the EDCA protocol for Eb/NO of 45 and using the realistic workload model. Each experiment was a single-run experiment with duration of 30 seconds for 10 contending STAs. Throughput was sampled using consecutive observation windows with duration of 100 microseconds. The variation of the throughput remains almost constant with time, showing that the system is in a stable state from very early on in the experiment.

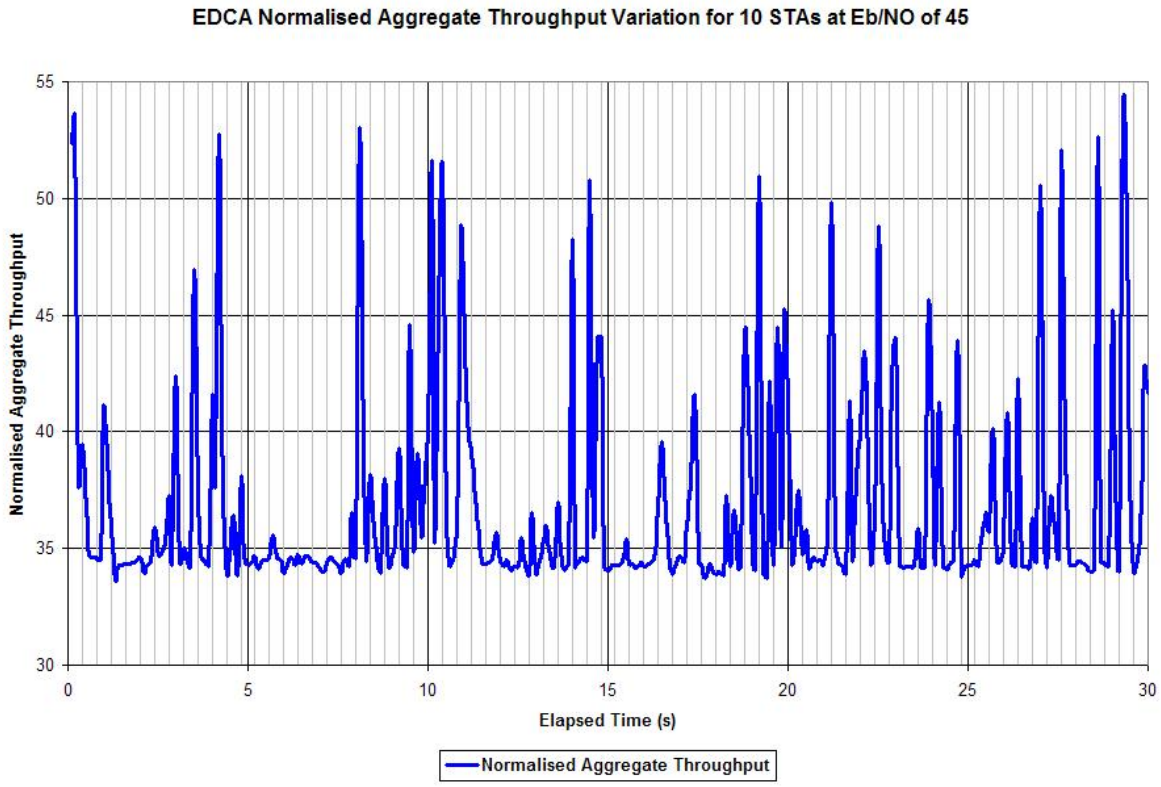

Figure 6.1: Variation of normalised aggregate throughput of 10 contending STAs with time for EDCA at Eb/NO of 45

The same is observed in the case of the DCF protocol. Figure 6.2 shows the throughput of the realistic workload for a 30-second run of 10 contending STAs, at Eb/NO of 45, and with 100-microsecond observation windows. Throughput behaviour suggests that the system also reaches a stable state very early on in the run.

Hence, for the experiments, it is not necessary to perform bias removal since the bias introduced by the transition period will have a negligible impact on the simulation results - provided the run durations are relatively long enough, as is in this case.

6.5 Results and Findings

In this section, experimental results are presented and discussed.

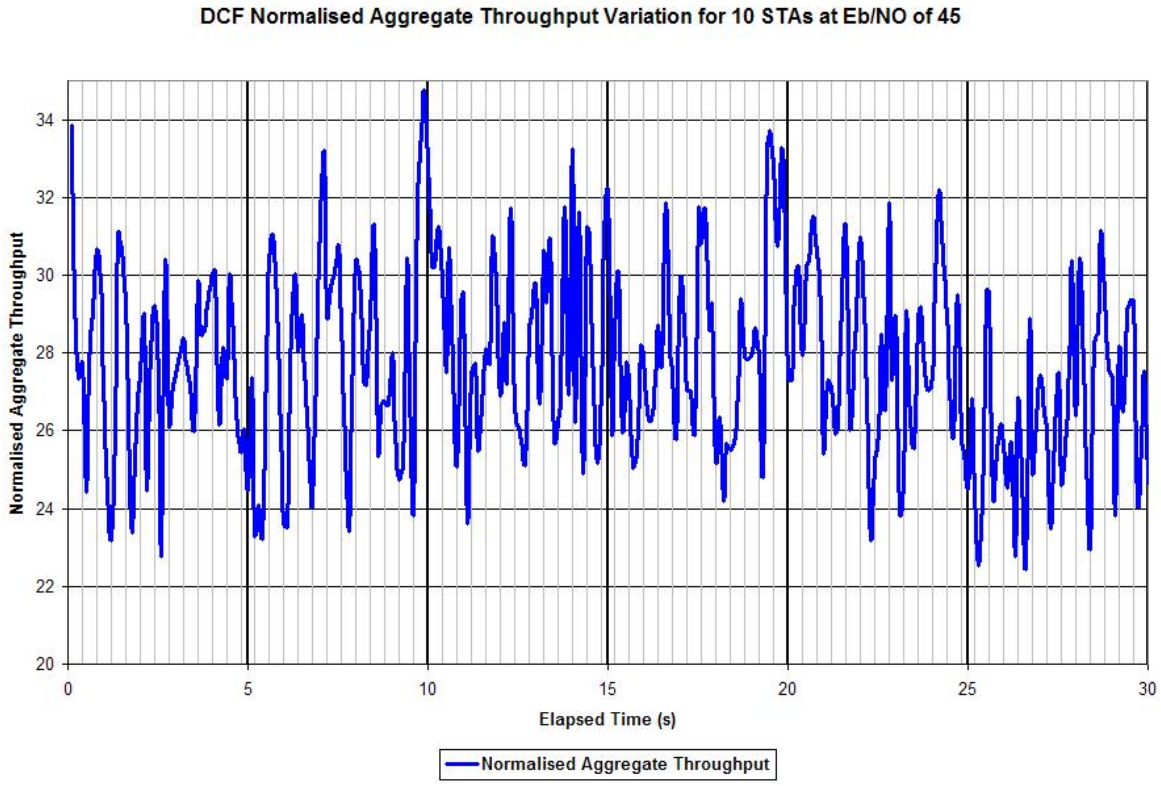

Figure 6.2: Variation of normalised aggregate throughput of 10 contending STAs with time for DCF at Eb/NO of 45

We first verify the effect of channel modelling by comparing results from thte simulator to the DNA hardware testbed [67]. We compare the simulation results for both ideal and realistic channel conditions to the results from [67] for the DCF protocol. Figure 6.3. The results show us that the channel model has an effect on network performance results. From the figure, we see that the performance with channel modelling is lower than that with ideal conditions, and it better matches the DNA test bed results as the number of STAs increases, which makes the simulation model provide a better approximation of network performance. This is because the realistic channel model attempts to mimic the environment of the test bed, incorporating channel errors and the effects of interference and noise, resulting in a better approximation of network performance.

We then discuss scalability experiments, which are experiments that that show an increasing number of STAs, thereby measuring the scalability of the network. We then discuss the comparison experiments, which compare the performance of each protocol for the different traffic types, and the overall performance for each protocol, for an increasing Eb/NO. These results consider all the performance metrics discussed in Section 6.2.1. For scalability experiments, we first look at the system performance, and then go on to discuss the performance from a user experience perspective. The same is done for the comparison experiments.

In Figure 6.4, we look at both EDCA and DCF. In this graph, we compare the normalised aggregate throughput of ideal channel conditions against realistic channel conditions with Eb/NO of 40. The comparison is done for an increasing number of STAs, form 1 to 16. Results show that there are differences in the results for both EDCA and DCF. EDCA shows a difference for 15 and 16 STAs, and this is mainly due to the errors in packet transmission in some of the packets that are transmitted in realistic channel conditions. Results show that EDCA normalised

Figure 6.3: Simulator normalised aggregate throughput for the simulator using both ideal and realistic channel conditions vs. the DNA hardware test bed [67] for DCF

aggregate throughput gradually increases with an increasing number of STAs. This is because the architecture of the EDCA protocol allows it to perform better for a larger number of STAs compared to DCF, and this can be attributed to the smaller contention windows and TXOP durations for high priority traffic. Since voice and video traffic have smaller contention windows, they can quickly gain access to the channel, reducing idle time, and since multiple packets are sent one after the other in one TXOP duration, idle times and collisions are further reduced and the network is able to transmit more packets than DCF.

DCF graphs show that the performance of a network is lower for Eb/NO of 40 compared to ideal conditions, meaning that a wireless medium suffers from external interference, even for a high Eb/NO.

The DCF normalised aggregate throughput rapidly increases from 1 to 3 STAs, but then gradually decreases, relatively constantly, as the number of STAs continues to increase. This can be attributed to a gradual increase in the number of collisions in the network, with the relative increase of collisions being greater than the increase in number of packets transmitted in the network.

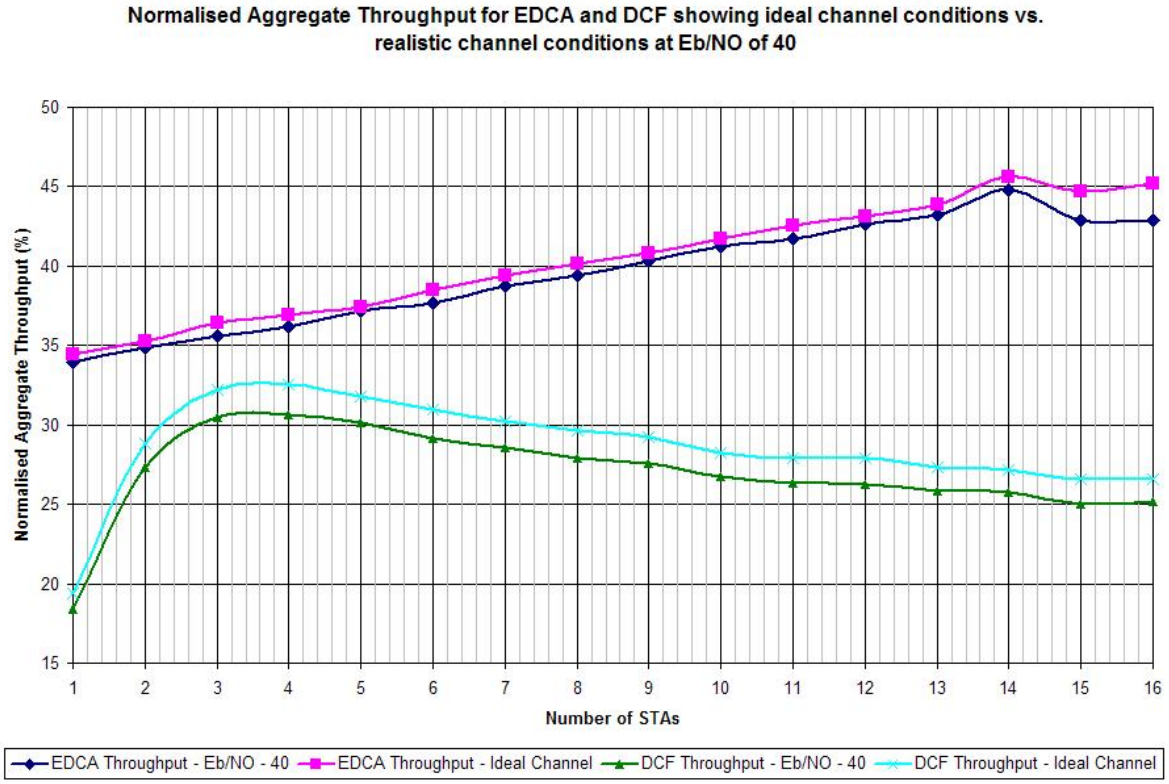

Figure 6.4: Graph of Normalised Aggregate Throughput against an increasing number of STAs for both EDCA and DCF showing ideal channel conditions vs. Realistic channel conditions at Eb/NO of 40

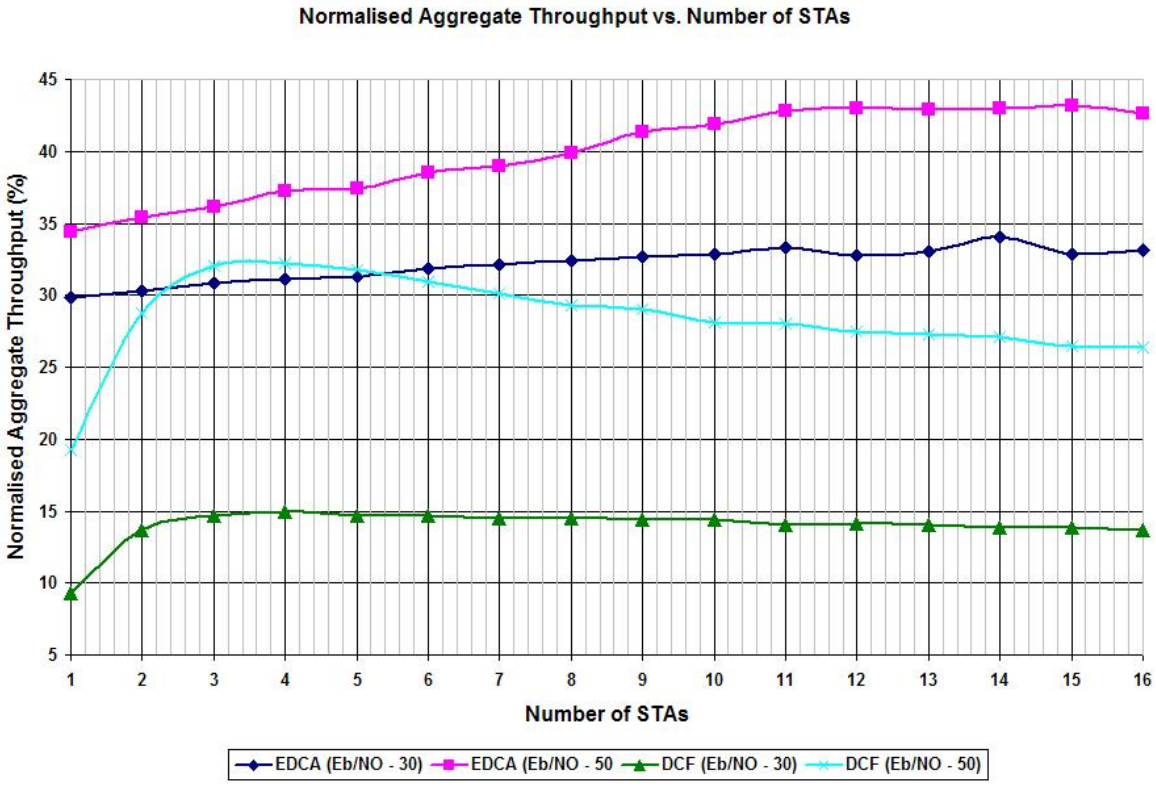

Figure 6.5: Graph of Normalised Aggregate Throughput percentage against an increasing number of STAs for EDCA and DCF for Eb/NO of 30 and 50

In Figure 6.5, we compare normalised aggregate throughput of both EDCA and DCF for Eb/NO values of 30 and 50. The graph shows the normalised aggregate throughput against an increasing number of STAs, from 1 to 16. Confidence intervals for the results are provided in Section B.1.

The results show us that the DCF protocol achieves lower throughput than EDCA, and Eb/NO of 50 achieves better network performance than and Eb/NO o30. From the analysis of these two protocols, it is apparent that the EDCA protocol achieves better network performance. The EDCA protocol allows a STA to send multiple packets for voice and video traffic in one window, without having to contend for the channel each time. This means that the waiting times are reduced, resulting in more packets being sent over the channel. This also means that the number of collisions is reduced as a STA 'exclusively' owns the transmission medium during its transmission for a TXOP duration (for voice and video traffic). The EDCA AIFS and CW sizes are much smaller for voice and video traffic compared to DCF, therefore EDCA STAs wait for shorter periods of time before contending for the channel.

The results in Figure 6.5 also tell us that the channel effect is significant. The achieved throughput for Eb/NO of 30 is lower than for Eb/NO of 50. This results from the amount of unwanted interference that affects the transmitted signal. at En/No of 30, these unwanted interferences have bigger influence than at Eb/NO of 50, and this results in more packets being transmitted with error, or even being dropped from the network after multiple transmissionretries.

Another observation is that DCF throughput increases when the number of STAs increases at first, but then gradually decreases. EDCA throughput, on the other hand, gradually increases. This shows that the EDCA protocol

gives better performance as the number of STAs in a network increases.

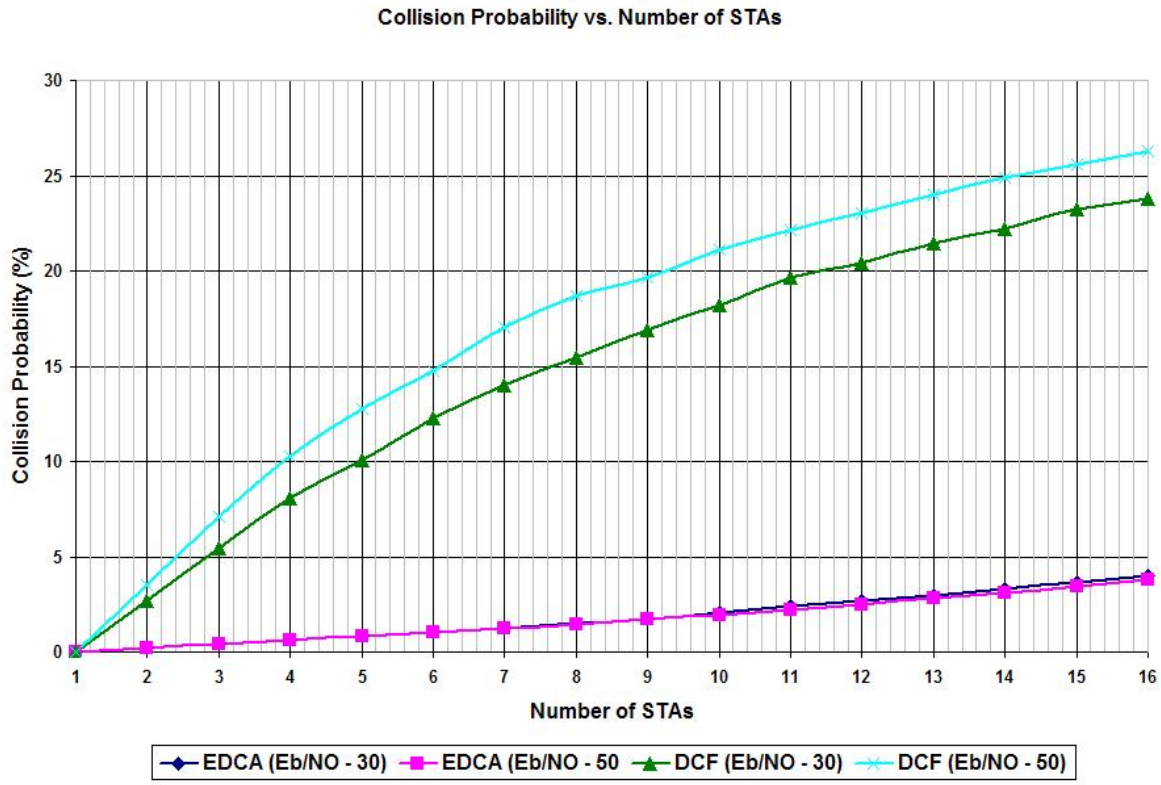

Figure 6.6: Graph of Collision Probability percentage against an increasing number of STAs for EDCA and DCF for Eb/NO of 30 and 50

Figure 6.6 shows the collision probability for both EDCA and DCF as the number of STAs increases, and for Eb/NO values of 30 and 50. The general indication from the results is that DCF has a higher collision probability compared to EDCA.

A higher collision probability for DCF means that a lot more packets are colliding, resulting in lower network throughput, and this best explains why the normalised aggregate throughput for DCF gradually decreases as the number of STAs increases from 3 STAs onwards. The collision probability is higher for Eb/NO of 50. This is because there are less errors in the network, resulting in more STAs attempting to transmit packets, which effectively results in more collisions.

The collision probability for EDCA is significantly lower compared to DCF, and it is similar for Eb/NO of 30 and 50. This is because the instances for collisions to occur in EDCA are much less compared to DCF, since EDCA STAs can transmit multiple packets in one window, i.e. the STAs do not have to contend for each packet for voice and video traffic. This results in the probability of collision being lower.

In Figure 6.7 we assess the system performance from the users experience perspective by looking at the access delay.

The access delay denotes how much a user waits before a packet is successfully transmitted.

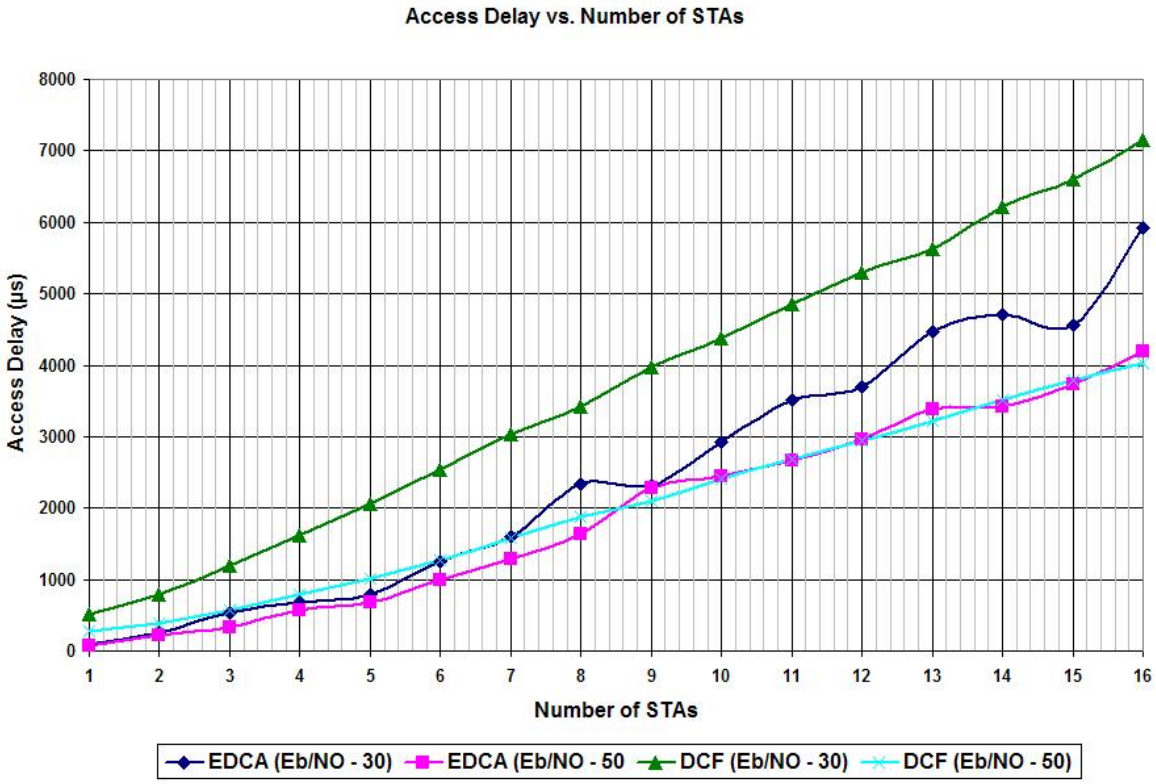

Figure 6.7: Graph of Access Delay in μs against an increasing number of STAs for EDCA and DCF for Eb/NO of 30 and 50

Figure 6.7 shows the access delay of both protocols, for an increasing number of STAs, from 1 to 16, and for Eb/NO values of 30 and 50.

The results show us that the access delay is higher at an Eb/NO of 30, with DCF having the highest access delay. This, again, is due to a lot of packet transmission errors in the network, and a lot of packets having to be re-transmitted. As the Eb/NO increases, the access delay reduces and eventually becomes relatively linear with an increasing number of STAs. This is also evident in the linearity of the normalised aggregate throughput and collision probability for high Eb/NO values.

The findings discussed for the scalability experiments show that EDCA performs better than DCF as the number of STAs increases. Looking at overall normalised aggregate throughput, EDCA performance gradually increases as the number of STAs increases, whereas DCFs rapidly increases until 3 STAs, and then gradually increases. The conclusion is that EDCA scales better than DCF, and therefore, will perform better in larger networks.

The findings also significantly emphasise how channel conditions affect network performance. For low Eb/NO values, the overall performance is lower, and, at times, fluctuating and erratic. This is because in such conditions, a lot of external factors influence the network, making it unpredictable. As shown in the graphs, the performance metrics for low Eb/NO values fluctuate more, on average, than those reported for higher Eb/NO values. This is a result of the unpredictable nature of the channel, i.e. the transmission errors introduced, which is greatly influenced by external sources.

Next, we discuss the comparison experiments, considering the three performance metrics discussed in Sections 6.2.1 and 6.2.2. The comparison experiments compare performance results of voice and video traffic (in order to compare results for QoS-sensitive traffic), and the overall protocol performance as the Eb/NO increases, for a network of 10 STAs. They help to understand how the network performs under various network conditions, thereby really showing the effects of channel modelling in performance evaluation of such networks. The confidence intervals for the results are provided in Section B.2.

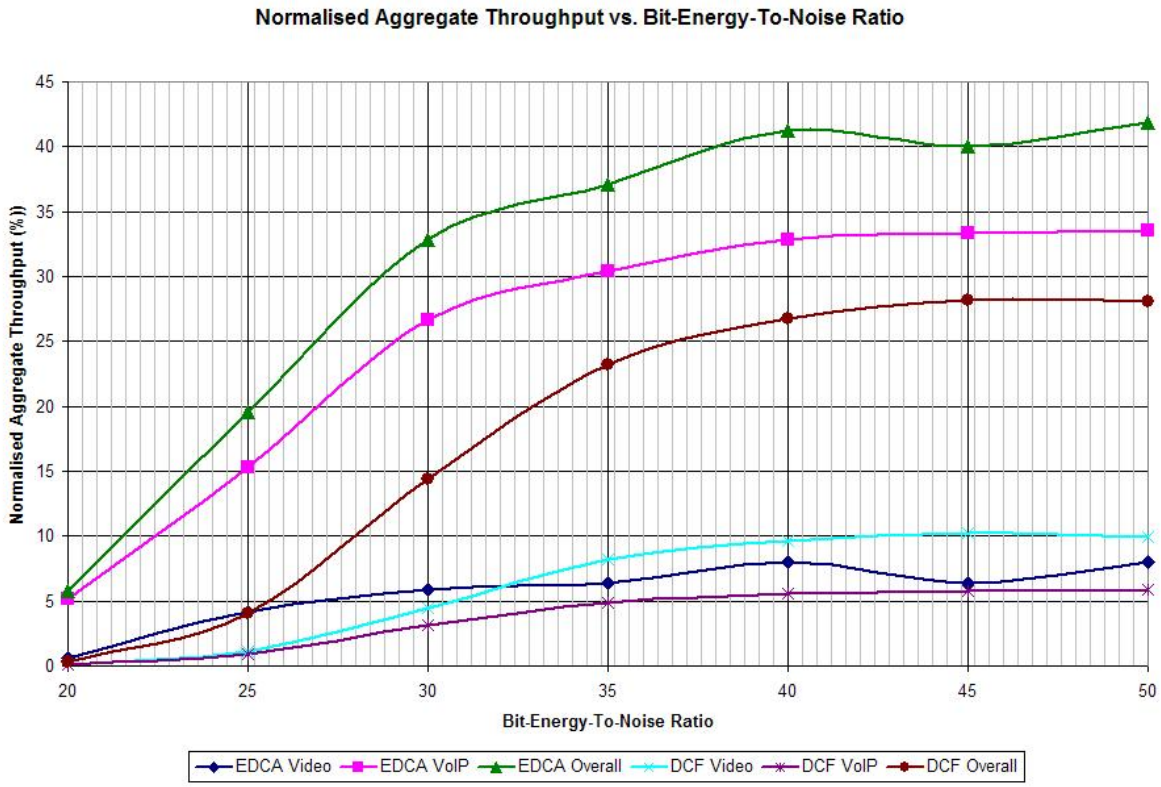

Figure 6.8: Graph of Normalised Aggregate Throughput against an increasing Signal-To-Noise ratio for voice and video traffic types and the integrated traffic in both EDCA and DCF

The discussion starts off with the normalised aggregate throughput, shown in Figure 6.8. From the graph, we see that the network throughput increases as the Eb/NO increases. This is expected since higher Eb/NO values mean that there are fewer errors in the network, resulting in significantly more packets being transmitted successfully.

For EDCA, VoIP traffic achieves the highest throughput, followed by video traffic. Throughput for DCF is evenly distributed, meaning that all four traffic types achieve relatively similar throughputs. This can be seen from Figure 6.8 where DCF voice and video are about a quarter of the overall throughput. The throughput for voice traffic in EDCA makes up the majority of traffic serviced, and since voice traffic is QoS-sensitive, it means that the EDCA protocol better services QoS traffic compared to DCF. With DCF the traffic evenly distributed meaning that the protocol does not differentiate the traffic types. This means that the DCF protocol is not suited to QoS-sensitive applications.

In EDCA, the ACs are prioritised in order to service QoS-sensitive applications. The AC for voice traffic has the smallest AIFS and contention window, which means that, on average, it accesses the channel more than the other ACs. Also, when it acquires the channel, it can send as many packets as can fit in the TXOP duration.

The same applies for video traffic, but it does not access the channel nearly as much as voice traffic. Best-effort and background traffic ACs access the channel the least, and when they successfully do, they are only allowed to transmit one packet at a time. Since they are of lower priority, if internal collision occurs, i.e. at least 2 ACs from the same STA try to access the channel at the same time, they give chance to the higher priority queues. The results for best-effort and background traffic are not illustrated here since the focus of the discussion is on QoS-sensitive traffic, i.e. voice and video traffic.

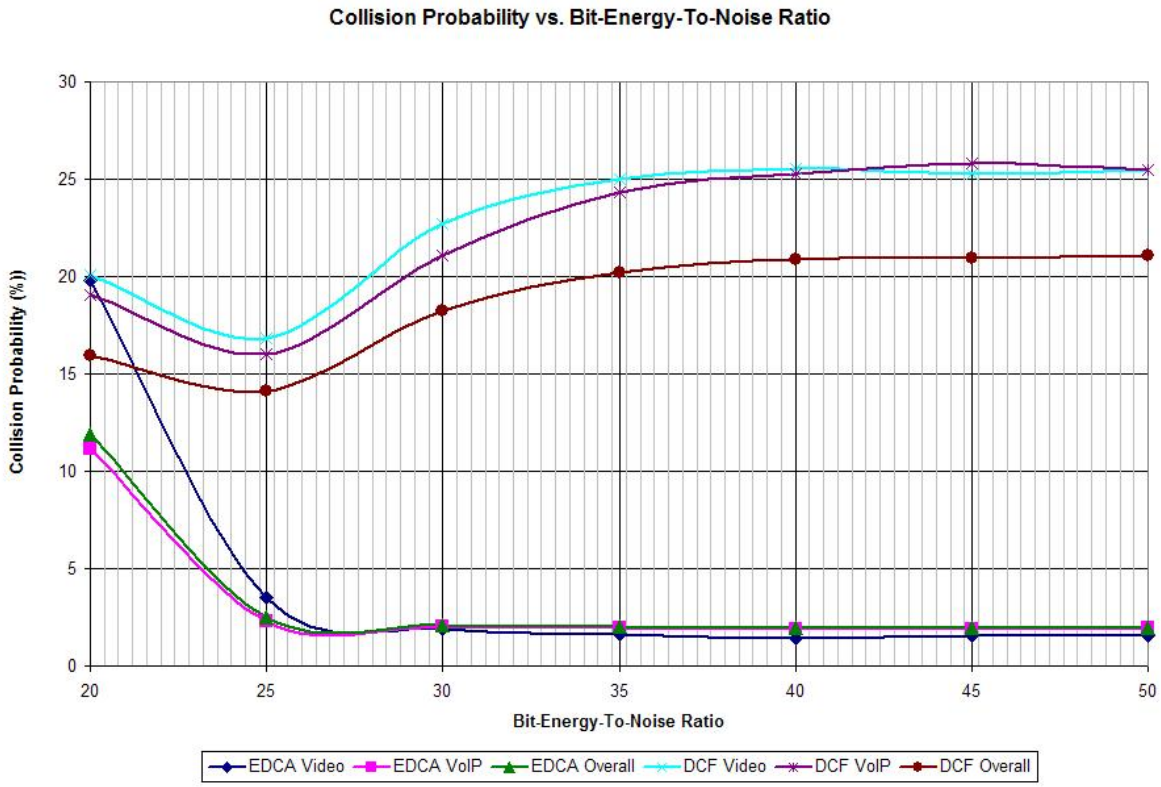

Figure 6.9: Graph of Collision Probability against an increasing Signal-To-Noise ratio for voice and video traffic types and the integrated traffic in both EDCA and DCF

The collision probability, shown in Figure 6.9, also decreases as the Eb/NO increases. Both EDCA voice and video traffic achieve the lowest collision probability. The collision probability is similar for voice and video traffic types for both protocols. The EDCA collision probability is initially high because the ratio of successfully transmitted packets to collisions is small because of the greater influence of interference and noise at a lower Eb/NO (at an Eb/NO of 20), but it rapidly decreases as the Eb/NO reaches 25. The EDCA collision probability is lower because the protocol allows for multiple packet transmission in one TXOP window, which means that not every packet is subject to channel contention (this is where collisions occur). The DCF protocol, however, has to contend for the channel for each packet transmission, therefore the rate of collisions inreases, resulting in a higher collision probability.

In Figure 6.10, the access delay is reported. As illustrated, the access delay decreases with an increasing Eb/NO until it remains constant. The access delay is initially higher for DCF.

On average, the overall EDCA access delay is lower than that of DCF, as illustrated in Figure 6.10. This is because the average waiting times of packets in the network is lower for EDCA than for DCF. It is lower because

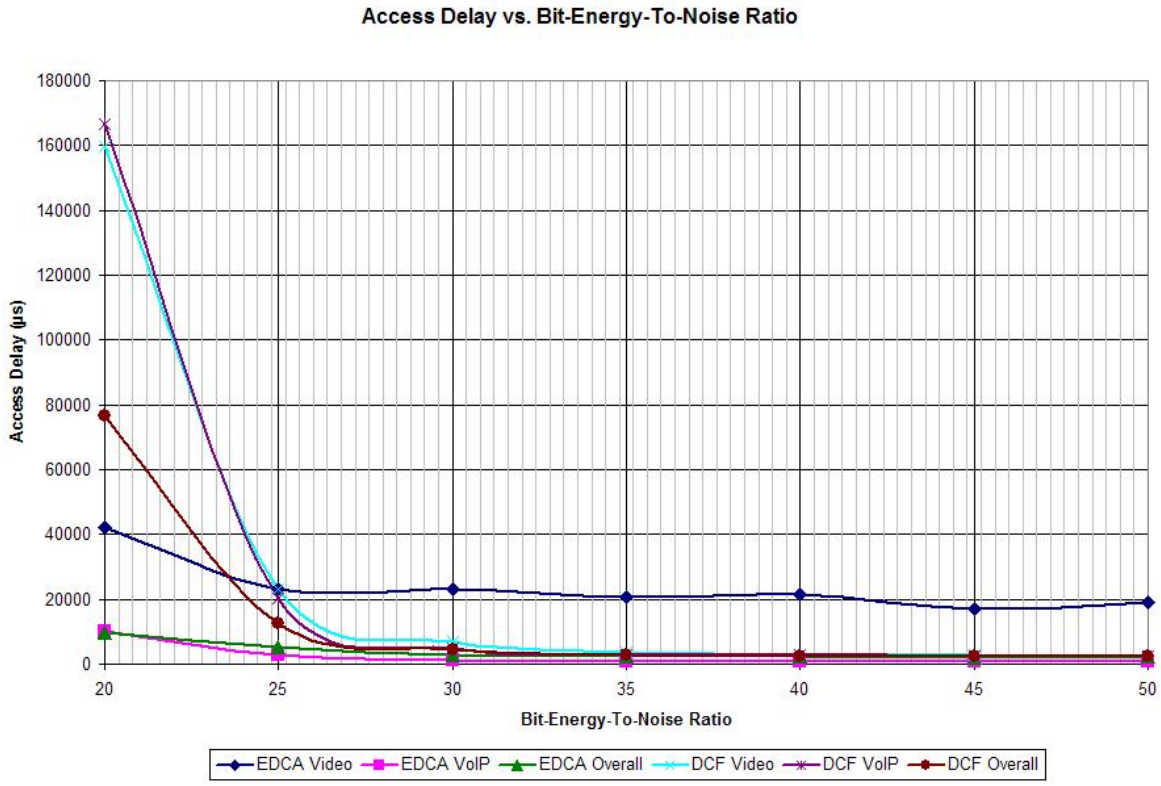

Figure 6.10: Graph of Access Delay against an increasing Signal-To-Noise ratio for voice and video traffic types and the integrated traffic in both EDCA and DCF

for voice traffic many packets can be transmitted one after the other in one TXOP, and the access delay of these packets will be low. Since the majority of EDCA packets transmitted are voice, the overall access delay is smaller. EDCA video traffic has the highest average access delay as the Eb/NO increases, and this is attributed to the longer waiting times that the video packets wait. Voice traffic has the smallest AIFS, so the voice Access Category (AC) can contend for and acquire the channel multiple times before a video AC does, thereby increasing its access delay.

From the findings discussed above, the performance of the network significantly improves as the Eb/NO increases for both EDCA and DCF. EDCA achieves better overall performance than DCF. This is so because EDCA, on average, transmits more packets in the same time frame than DCF, since EDCA can send many packets per contention (for voice and video packets), whereas DCF only transmits one packet.

Figure 6.8 shows that the normalised aggregate throughput achieved for EDCA voice alone is greater than that of DCF. EDCA video also achieves a relatively high throughput. The results from the DCF protocol show that the throughput of all traffic types is somewhat equally distributed. This shows that the EDCA protocol is better suited to QoS-sensitive applications than DCF. However, lower priority traffic is significantly starved in the EDCA protocol, compared to DCF.

It is very important to understand the type of applications that will utilise a networks resources before choosing which protocol to implement. EDCA is better suited for VoIP applications, since they require real-time delivery, whereas these applications will perform poorly with DCF.

CHAPTER 7

CONCLUSION

In this work, we took great care to accurately represent the 802.11g and 802.11e protocols in a simulation environment. These protocols were accompanied by a channel model and a workload model. The channel model was a combination of the OFDM modulation scheme, Rayleigh multi-path fading, and Additive White Gaussian Noise (AWGN). The choice of OFDM was motivated by the fact that it is used in both 802.11g and 802.11e. The nature of Rayleigh multi-path fading is that which is experienced in heavily built environments, such as urban metropolitans, which was the focus of our work. AWGN introduces the effect of noise in the wireless medium. A suitable workload model was developed, which generated four types of traffic: voice, video, P2P and HTTP. The density of the IATs was increased in order to introduce more packets into the system, but without distorting the IAT distributions.

Many factors affect network performance, and in different ways. There is much overhead, such as packet headers, which use up bandwidth; although, they are necessary for efficient operation of a network. Channel conditions seem to have the most appreciable impact on network performance.

Channel contention also affects the overall network performance. When two or more STAs acquire the channel simultaneously, packets are transmitted with error and, as the number of contending STAs increases, the probability of this happening also increases, resulting in a decrease in network performance. This is because the data packets transmitted do not contribute to throughput, since they are corrupt; therefore, the bandwidth used for the transmission is wasted.

The effects of channel contention can also be seen through the better EDCA performance illustrated in Section 6.5. First of all, the AIFS intervals for voice and video traffic in EDCA are much smaller than the DCFs DIFS interval, meaning that, on average, an EDCA STA contending for the channel waits for shorter periods before backing off.

When the voice and video ACs acquire the channel, they send as many packets as can fit into their respective TXOP intervals, meaning that a lot of these packets are sent sequentially, without having to contend for the channel as DCF does. This results in less contention times for EDCA and higher network performance. This also results in better scalability in the EDCA protocol, where the overall networks performance increases with an increasing number of STAs.

The size of data packets also has an impact on the performance, with an increasing length of the packets resulting in increased throughput. However, increasing the packet length can only work for a few STAs since the cost of packet collisions also increases.

Simplifying assumptions can be made in order to make the development of a model easier and quicker. They also ensure that performance data is obtained faster. This potentially results in an inaccurate representation of network performance. The DNA hardware testbed factors in realistic channel conditions and its performance decreases faster than the other models with an increasing number of STAs, since this is the actual system.

It is evident that the channel is not always perfect, especially since wireless is known for being unreliable in nature. It is best to make as little simplifying assumptions as possible to acquire a better network performance estimate. However, it requires a great deal more investment in terms of time, effort, and resources; for example, most existing testbeds have a limited number of STAs.

In future, it would be interesting to see how the IEEE 802.11n protocol performs compared to the DCF and EDCA, and to see if there is any improvement in the delivery of QoS-sensitive traffic. The IEEE 802.11n standard was implemented in order to improve service delivery of QoS applications and to increase channel capacity. It is important to verify whether the percentage of QoS traffic delivered has improved compared to 80.11e and 802.11g. It is also important to see how non-QoS-sensitive applications are affected.

BIBLIOGRAPHY

[1] A. Aguiar and J. Gross, "Wireless Channel Models, Technical Report TKN-03-007," in *Telecommunication Networks Group, Technische Universitaet Berlin*. [Online]. Available: http://paginas.fe.up.pt/~anaa/publications.html

[2] A. Baiocchi, N. B. Melazzi, M. Listanti, A. Roveri, and R. Winkler, "Loss Performance Analysis of an ATM Multiplexer Loaded with High-speed ON-OFF Sources," *IEEE J. Select. Areas Commun.*, vol. 9, no. 3, pp. 388–393, 1991.

[3] B. Bellalta, M. Oliver, and D. Rincon, "Capacity and Traffic Analysis of Voice Services over GPRS Mobile Networks," *Technical University of Catalonia, University Pompeu Fabra, Spain*, 2002.

[4] G. Bianchi, "Performance analysis of the 802.11 Distributed Coordination Function," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 535–547, 2000.

[5] J. Bicket, D. Aguayo, S. Biswas, and R. Morris, "Architecture and evaluation of an unplanned 802.11b mesh network," in *MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM, 2005, pp. 31–42. [Online]. Available: http://dx.doi.org/10.1145/1080829.1080833

[6] L. Blasi, G. Boggia, P. Camarda, L. Carbone, and L. A. Grieco, "Extended EDCA for providing bounded delay services in 802.11e WLANs," in *MSWiM '06: Proceedings of the 9th ACM international symposium on Modeling analysis and simulation of wireless and mobile systems*. New York, NY, USA: ACM, 2006, pp. 273–276.

[7] J. Cao, W. S. Cleveland, Y. Gao, K. Jeffay, F. D. Smith, and M. Weigle, "Stochastic Models for Generating Synthetic HTTP Source Traffic," *IEEE INFOCOM*, 2004.

[8] Q. Chen, F. Schmidt-Eisenlohr, D. Jiang, M. Torrent-Moreno, L. Delgrossi, and H. Hartenstein, "Overhaul of IEEE 802.11 Modelling and Simulation in NS-2," *DaimlerChrysler Research*, 2007.

[9] T. P.-C. Chen and T. Chen, "Markov Modulated Punctured Autoregressive Processes for Traffic and Channel Modeling," *Proc. Of Packet Video*, 2002.

[10] S. Choi, Del, Sai, and S. Mangold, "IEEE 802.11 e contention-based channel access (EDCF) performance evaluation," in *Communications, 2003. ICC '03. IEEE International Conference on*, vol. 2, 2003, pp. 1151–1156 vol.2. [Online]. Available: http://dx.doi.org/10.1109/ICC.2003.1204546

[11] S. Cocorada, "An ieee 802.11 g simulation model with extended debug capabilities," in *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, ser. Simutools '08. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008, pp. 81:1–81:3. [Online]. Available: http://portal.acm.org/citation.cfm?id=1416222.1416314

[12] D. Covarrubias and M. A. Gmez, "Determining the optimum length of voice packet for transmission in packet switching networks," *Journal of the Mexican Society of Instrumentation, Revista de la Sociedad Mexicana de Instrumentacin, A. C.*, vol. 3, no. 5, pp. 8–13, 1995.

[13] T. D. Dang, B. Sonkoly, and S. Molnr, "Fractal Analysis and Modeling of VoIP Traffic," *High Speed Networks Lab, Dept. of Telecommunications and Media Informatics, Budapest Univ. of Technology and Economics, H1117, Magyar tudsok krtja 2*.

[14] F. S. Fading, "Retrieved November 14, 2010, howpublished = "http://sna.csie.ndhu.edu.tw/~cnyang/MCCDMA/tsld036.htm",."

[15] O. S. T. R. . A. . from the Opnet website, "Heterogeneous concurrent Modeling in Java, howpublished = "http://www.opnet.com",."

[16] V. Frost and B. Melamed, "Traffic Modelling for Telecommunications Networks," *IEEE Communications Magazine*, pp. 70–81, 1994.

[17] D. Gu and J. Zhang, "Evaluation of EDCF mechanism for QoS in IEEE 802.11 wireless networks," in *IWCMC '06: Proceedings of the 2006 international conference on Wireless communications and mobile computing*, Mitsubishi Electric Information Technology Center America, 2003.

[18] B. G. Haskell, "Buffer and channel sharing by several interframe picture phone coders," *Bell Sys. Tech. J.*, pp. 261–289, 1972.

[19] D. He and C. Q. Shen, "Simulation Study of the IEEE 802.11e EDCF," *Institute of Infocomm Research*, vol. 1, pp. 685–689, 2003.

[20] H. Heffes and D. M. Lucantoni, "A Markov Modulated characterization of Packetized Voice and Data Traffic and Related Statistical Multiplexer Performance," *IEEE J. Select. Areas Commun.*, vol. SAC-4, no. 6, pp. 856–868, 1986.

[21] J. Herre, H. Purnhagen, J. Breebaart, C. Faller, S. Disch, K. Kjrling, E. Schuijers, J. Hilpert, and F. Myburg, "The Reference Model Architecture for MPEG Spatial Audio Coding," in *Preprint 118th Conv. Aud. Eng. Soc.*, 2005. [Online]. Available: www.aes.org

[22] J. Huang, T. Le-Ngoc, and J. F. Hayes, "Broadband SATCOM system for multimedia services," *Proc. ICC96*, pp. 906–909, 1996.

[23] P. II, "Heterogeneous concurrent Modeling in Java, howpublished = "http://www.eecs.berkeley.edu",."

[24] C. M. G. D. Institute, "PPLive Traffic Modelling."

[25] Intel and 802.11, "Helping Define 802.11n and other Wireless LAN Standards," http://www.intel.com/standards/case/case_802_11.htm.

[26] D. L. J. Cao, W. S. Cleveland and D. X. Sun, "On the Nonstationarity of Internet Traffic," *ACM SIGMETRICS*, vol. 29, no. 1, pp. 102–112, 2001.

[27] R. Jain and S. A. Routhier, "Packet Trains: Measurements and a New Model for Computer Network Traffic," *IEEE Journal on Selected Areas in Communications*, vol. SAC-4, no. 6, pp. 986–995, 1986.

[28] P. Ji, B. Liu, D. Towsley, Z. Ge, and J. Kurose, "Modeling frame-level errors in gsm wireless channels," *Perform. Eval.*, vol. 55, pp. 165–181, January 2004. [Online]. Available: http://dx.doi.org/10.1016/S0166-5316(03)00104-4

[29] W. Jiang and H. Schulzrinne, "Analysis of On-Off Patterns in VoIP and Their Effect on Voice Traffic Agregation," 2000.

[30] D. Johnson and A. Lysko, "Overview of the Meraka wireless grid test bed for evaluation of ad-hoc routing protocols," Meraka Institute, CSIR, 2007.

[31] S. Kim, M. Lee, and M. Kim, "E-Matching Technique for MMPP Modeling of Hetrogeneous ONOFF Sources," *Proc. Globecom 94*, pp. 1090–1094, 1994.

[32] A. Konrad, B. Y. Zhao, A. D. Joseph, and R. Ludwig, "A Markov-based channel model algorithm for wireless networks," *Wirel. Netw.*, vol. 9, no. 3, pp. 189–199, 2003.

[33] I. Labs, "VoIP and 802.11," 2006. [Online]. Available: www.opus1.com/voip/2006ilabs80211voip.pdf

[34] J. Landman, "Analytical Models of IP Traffic on UMTS Mobile Networks," *Dissertation submitted to the Department of Computer Science, Faculty of Science at the University of Cape Town*, 2005.

[35] A. M. Law and W. D. Kelton, "Simulation Modeling and Analysis," *Third Edition, Mcgraw-Hill Higher Education*, 2000.

[36] C.-W. Lee, C.-S. Yang, and Y.-C. Su, "Adaptive UEP and packet size assignment for scalable video transmission over burst-error channels," *EURASIP J. Appl. Signal Process.*, no. 1, pp. 256–256, 2006.

[37] T. Li, Q. Ni, D. Malone, and D. Leith, "A new MAC scheme for very high-speed WLANs," in *Proceedings of the 2006 International Syposium on a World of Wireless, Mobile and Multiedia Networks (WoWMoM06)*, 2006.

[38] M. Lucas, B. J. Dempsey, D. E. Wrege, and A. C. Weaver, "(M , P, S) - An Efficient Background Traffic Model for Wide-Area Network Simulation," *Department of Computer Science, University of Virginia, Charlottesville*, 1997. [Online]. Available: ieeexplore.ieee.org/iel4/5002/14054/00644399.pdf

[39] M. Lucas, D. Wrege, B. Dempsey, and A. Weaver, "Statistical Characterization of Wide-Area IP Traffic," *In Proceedings qf Sixth International Conference on Computer Communications and Networks (IC3N'97)*, 1997.

[40] H. Lundgren, I. Sheriff, P. A. Kumar, A. Lam, J. Miller, K. Sanzgiri, and E. Belding-Royer, "Video Performance Analysis in the UCSB MeshNet Testbed," 2005.

[41] B. Maglaris, "Performance models of statistical multiplexing in packet video communications," *IEEE Tran. Comm.*, pp. 834–844, 1998.

[42] D. Malone, P. Clifford, and D. J. Leith, "Mac layer channel quality measurement in 802.11." *IEEE Communications Letters.*, vol. 11, no. 2, pp. 143–145, 2007.

[43] S. Manitpornsut and B. Landfeldt, "On the performance of IEEE 802.11 QoS mechanisms under spectrum competition," in *IWCMC '06: Proceedings of the 2006 international conference on Wireless communications and mobile computing*. New York, NY, USA: ACM, 2006, pp. 719–724.

[44] J. McDougall and S. Miller, "Sensitivity of Wireless Network Simulations to a two-state Markov Model Channel Approximation," *Department of Electrical Engineering*, vol. 2001.

[45] NS-2, "The Network Simulator - NS-2," http://www.isi.edu/nsnam/ns.

[46] OMNET++, "A Discrete Event Simulation System," http://www.omnetpp.org.

[47] K. P, Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan, "Measurement, Modeling, and Analysis of a Peer-to-Peer File-Sharing Workload."

[48] P802.11, "Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, ANSI/IEEE Std. 802.11," IEEE, Tech. Rep., 2007.

[49] V. Paxson and S. Floyd, "Wide-area traffic: The failure of Poisson modelling," *IEEE/ACM Trans. On Networking*, vol. 3, 1995.

[50] M. Pernyi, T. D. Dang, A. Gefferth, and S. Molnri, "Identification and analysis of Peer-to-Peer Traffic," *JOURNAL OF COMMUNICATIONS*, vol. 1, no. 7, 2006.

[51] M. R. Philipp Svoboda, Wolfgang Karner, "MODELING E-MAIL TRAFFIC FOR 3G MOBILE NETWORKS."

[52] K. Pillay, "BER for BPSK in OFDM with Rayleigh multipath channel (2010), Retrieved 10 January 2010," http://www.dsplog.com/2008/08/26/ofdm-rayleigh-channel-ber-bpsk/.

[53] PPLive, "Retrieved 27 January, 2010, howpublished = "http://www.pplive.com/",."

[54] M. Project, "Retrieved January 27, 2010, howpublished = "http://www.ftw.at/ftw/research/projects/",."

[55] W.-F. A. P. Release, "Wi-Fi Industry to Ship 300 Million Chipsets in 2007," 2007.

[56] ——, "802.11n Equipment Sales Show Continued Momentum One Year After Wi-Fi Alliance Certification Program Begins," 2008.

[57] O. Saleh and M. Hefeeda, "Modeling and Caching of Peer-to-Peer Traffic," in *Institute of Electrical and Electronic Engineers, IEEE*. IEEE, 2006.

[58] J. H. Sarker, "Models for Video Traffic," *Communications Laboratory, Institute of radio Communications*. [Online]. Available: www.netlab.tkk.fi/opetus/s38149/s99/reports/1108jahangir.doc

[59] H. Schulzrinne, "Voice communication across the Internet: A network voice terminal," *Technical Report TR 92-50*, 1992.

[60] J. Seger, "Modelling Approach for VoIP Traffic Aggregations for Transferring Tele-traffic Trunks in a QoS-enabled IP-Backbone Environment."

[61] P. Sen, "Models for packet switching of variable-bit-rate video sources," *IEEE Sel. Area on Comm.*, pp. 865–869, 1989.

[62] M. S. Sendil, "An Optimized Method for Analyzing the Peer to Peer Traffic," *European Journal of Scientific Research*, vol. 34, no. 4, pp. 535–541, 2009.

[63] S. Shah-Heydari and T. Le-Ngoc, "MMPP Modeling of Aggregated ATM Traffic."

[64] V. Siris and C. Courcoubetis, "Resource Control for the EDCA and HCCA Mechanisms in IEEE 802.11e Networks," *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, 2006 4th International Symposium on Modeling and Optimization in Mobile, Ad-Hoc and Wireless Networks*, pp. 1–6, April 2006.

[65] V. A. Siris and C. Courcoubetis, "Resource Control for the EDCA Mechanism in Multi-Rate IEEE 802.11e Networks," in *WOWMOM '06: Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 419–428.

[66] P. size, "Retrieved November 07, 2010, howpublished = "http://publib.boulder.ibm.com/infocenter/iwedhelp/v6r0/index.jsp?topic=%2Fcom.ibm.mqe.doc%2Fcom10009.html",."

[67] A. Symington and P. Kritzinger, "A hardware test bed for measuring IEEE 802.11g distribution coordination function performance," *Modelling, Analysis and Simulation of Computer and Telecommunications Systems, MASCOTS*, pp. 1–7, 2009.

[68] H. tailed distributions, "Retrieved November 14, 2010, howpublished = "http://www.ee.ust.hk/~heixj/publication/comp660f/node4.html",."

[69] K.-S. Test, "Retrieved April 25, 2010, howpublished = "http://www.physics.csbsju.edu/stats/KS-test",."

[70] M. Thottan and M. C. Weigle, "Impact of 802.11e EDCA on mixed TCP-based applications," in *WICON '06: Proceedings of the 2nd annual international workshop on Wireless internet*. New York, NY, USA: ACM, 2006, p. 26.

[71] D. Tse and P. Viswanath, "Fundamentals of Wireless Communication," *WIRELESS-COMMUNICATION*.

[72] I. T. Union, "Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excite linear-predictioon anex b, A silence compression scheme for g.792 optimized for terminals conforming to recommendation v.70," 1996.

[73] J. Walsh and D. Koconis, "Background Traffic and Network IPS Testing," *Cybertrust*, 2006.

[74] J. Weinmiller, M. Schlager, A. Festag, and A. Wolisz, "Performance study of access control in wireless LANs IEEE 802.11 DFWMAC and ETSI RES 10 Hiperlan," *Mobile Networks and Applications*, vol. 2, 1997.

[75] Y. Xiao, "Orthogonal Frequency Division Multiplexing Modulation and Inter-Carrier Interference Cancellation," *A Thesis submitted to the Graduate Faculty of the Louisiana State University and Agricultural and Mechanical College*, 2009.

APPENDIX A

SYSTEM TESTING

This section contains a well-detailed discussion of the system testing. Testing was carried out for the Machine Model (DCF and EDCA protocols), the channel model, and the workload model.

A.1 MACHINE MODEL TESTING

In this section, the testing for the machine model is presented. The first sub-section presents the testing carried out on the DCF protocol, and the second section presents the EDCA protocol testing.

A.1.1 DCF Testing

This section shows output from DCF testing.

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          8730.0
Observation:             100000.0
End Of DIFS:             8751.222222222223
End Of Back-off:         1.0E9
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      1.0E9
End Of CTS Timeout:      1.0E9
End Of Simulation:       10000.0

Event Number: 1
Event Type:   PACKET ARRIVAL
System Clock: 8730.0
```

Figure A.1: DCF Packet Arrival event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:         8763.0
Observation:            100000.0
End Of DIFS:            8751.222222222223
End Of Back-off:        1.0E9
End Of Transmission:    1.0E9
End Of SIFS:            1.0E9
End Of RTS:             1.0E9
End Of CTS:             1.0E9
End Of ACK:             1.0E9
End Of ACK Timeout:     1.0E9
End Of CTS Timeout:     1.0E9
End Of Simulation:      10000.0

Event Number: 4
Event Type:   END OF DIFS
System Clock: 8751.222222222223
```

Figure A.2: DCF End of DIFS event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:         8763.0
Observation:            100000.0
End Of DIFS:            1.0E9
End Of Back-off:        8760.222222222223
End Of Transmission:    1.0E9
End Of SIFS:            1.0E9
End Of RTS:             1.0E9
End Of CTS:             1.0E9
End Of ACK:             1.0E9
End Of ACK Timeout:     1.0E9
End Of CTS Timeout:     1.0E9
End Of Simulation:      10000.0

Event Number: 5
Event Type:   END OF BACK-OFF
System Clock: 8760.222222222223
```

Figure A.3: DCF End of Back-off event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:         9062.0
Observation:            100000.0
End Of DIFS:            1.0E9
End Of Back-off:        1.0E9
End Of Transmission:    9051.296296296296
End Of SIFS:            1.0E9
End Of RTS:             1.0E9
End Of CTS:             1.0E9
End Of ACK:             1.0E9
End Of ACK Timeout:     9100.296296296296
End Of CTS Timeout:     1.0E9
End Of Simulation:      10000.0

Event Number: 12
Event Type:   END OF TRANSMISSION
System Clock: 9051.296296296296
```

Figure A.4: DCF End of Transmission event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          9073.0
Observation:             100000.0
End Of DIFS:             1.0E9
End Of Back-off:         1.0E9
End Of Transmission:     1.0E9
End Of SIFS:             9067.296296296296
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      1.0E9
End Of CTS Timeout:      1.0E9
End Of Simulation:       10000.0

Event Number: 13
Event Type:   END OF SIFS
System Clock: 9067.296296296296
```

Figure A.5: DCF End of SIFS event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          9096.0
Observation:             100000.0
End Of DIFS:             1.0E9
End Of Back-off:         1.0E9
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              9092.962962962962
End Of ACK Timeout:      9143.962962962962
End Of CTS Timeout:      1.0E9
End Of Simulation:       10000.0

Event Number: 14
Event Type:   END OF ACK
System Clock: 9092.962962962962
```

Figure A.6: DCF End of ACK event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          9447.0
Observation:             100000.0
End Of DIFS:             1.0E9
End Of Back-off:         1.0E9
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              9444.148148148151
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      1.0E9
End Of CTS Timeout:      9494.148148148151
End Of Simulation:       10000.0

Event Number: 15
Event Type:   END OF RTS
System Clock: 9444.148148148151
```

Figure A.7: DCF End of RTS event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          9467.0
Observation:             100000.0
End Of DIFS:             1.0E9
End Of Back-off:         1.0E9
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              9462.63703703704
End Of ACK:              1.0E9
End Of ACK Timeout:      1.0E9
End Of CTS Timeout:      9536.814814814818
End Of Simulation:       10000.0

Event Number: 16
Event Type:   END OF CTS
System Clock: 9462.63703703704
```

Figure A.8: DCF End of CTS event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          10009.0
Observation:             100000.0
End Of DIFS:             1.0E9
End Of Back-off:         1.0E9
End Of Transmission:     10040.555555555555
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      1.0E9
End Of CTS Timeout:      1.0E9
End Of Simulation:       10000.0

Event Number: 17
Event Type:   END OF SIMULATION
System Clock: 10000.0
```

Figure A.9: DCF End of Simulation event output snippet

```
Packet Arrival:          292.0
Observation:             100000.0
End Of DIFS:             1.0E9
End Of Back-off:         1.0E9
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      288.0740740740741
End Of CTS Timeout:      1.0E9
End Of Simulation:       10000.0

Event Number: 18
Event Type:   END OF ACK TIMEOUT
System Clock: 288.0740740740741
```

Figure A.10: DCF End of ACK Timeout event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          18201.624881917527
Observation:             100000.0
End Of DIFS:             1.0E9
End Of Back-off:         1.0E9
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      1.0E9
End Of CTS Timeout:      18181.22962962963
End Of Simulation:       20000.0

Event Number: 19
Event Type:   END OF CTS TIMEOUT
System Clock: 18181.22962962963
```

Figure A.11: DCF End of CTS Timeout event output snippet

A.1.2 **EDCA Testing**

This section shows the testing output for EDCA.

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          0.0
Observation:             100000.0
End Of AIFS - AC0:       79.0
End Of AIFS - AC1:       43.0
End Of AIFS - AC2:       34.0
End Of AIFS - AC3:       34.0
End Of Back off - AC0:   1.0E9
End Of Back off - AC1:   1.0E9
End Of Back off - AC2:   1.0E9
End Of Back off - AC3:   1.0E9
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      1.0E9
End Of CTS Timeout:      1.0E9
End Of Simulation:       10000.0

Event Number: 1
Event Type:   PACKET ARRIVAL
System Clock: 0.0
```

Figure A.12: EDCA Packet Arrival event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          3594.0
Observation:             100000.0
End Of AIFS - AC0:       3593.6296296296305
End Of AIFS - AC1:       1.0E9
End Of AIFS - AC2:       1.0E9
End Of AIFS - AC3:       1.0E9
End Of Back off - AC0:   1.0E9
End Of Back off - AC1:   3661.6296296296305
End Of Back off - AC2:   3678.6296296296305
End Of Back off - AC3:   3606.6296296296305
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      3634.6296296296305
End Of CTS Timeout:      1.0E9
End Of Simulation:       10000.0

Event Number: 6
Event Type:   END OF AIFS (AC0)
System Clock: 3593.6296296296305
```

Figure A.13: EDCA End of AIFS (AC0) event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          44.0
Observation:             100000.0
End Of AIFS - AC0:       79.0
End Of AIFS - AC1:       43.0
End Of AIFS - AC2:       69.0
End Of AIFS - AC3:       1.0E9
End Of Back off - AC0:   1.0E9
End Of Back off - AC1:   1.0E9
End Of Back off - AC2:   1.0E9
End Of Back off - AC3:   52.0
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      1.0E9
End Of CTS Timeout:      1.0E9
End Of Simulation:       10000.0

Event Number: 7
Event Type:   END OF AIFS (AC1)
System Clock: 43.0
```

Figure A.14: EDCA End of AIFS (AC1) event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          35.0
Observation:             100000.0
End Of AIFS - AC0:       79.0
End Of AIFS - AC1:       43.0
End Of AIFS - AC2:       34.0
End Of AIFS - AC3:       1.0E9
End Of Back off - AC0:   1.0E9
End Of Back off - AC1:   1.0E9
End Of Back off - AC2:   1.0E9
End Of Back off - AC3:   52.0
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      1.0E9
End Of CTS Timeout:      1.0E9
End Of Simulation:       10000.0

Event Number: 8
Event Type:   END OF AIFS (AC2)
System Clock: 34.0
```

Figure A.15: EDCA End of AIFS (AC2) event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          35.0
Observation:             100000.0
End Of AIFS - AC0:       79.0
End Of AIFS - AC1:       43.0
End Of AIFS - AC2:       34.0
End Of AIFS - AC3:       34.0
End Of Back off - AC0:   1.0E9
End Of Back off - AC1:   1.0E9
End Of Back off - AC2:   1.0E9
End Of Back off - AC3:   1.0E9
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      1.0E9
End Of CTS Timeout:      1.0E9
End Of Simulation:       10000.0

Event Number: 9
Event Type:   END OF AIFS (AC3)
System Clock: 34.0
```

Figure A.16: EDCA End of AIFS (AC3) event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          3594.0
Observation:             100000.0
End Of AIFS - AC0:       1.0E9
End Of AIFS - AC1:       1.0E9
End Of AIFS - AC2:       1.0E9
End Of AIFS - AC3:       1.0E9
End Of Back off - AC0:   3593.6296296296305
End Of Back off - AC1:   3661.6296296296305
End Of Back off - AC2:   3678.6296296296305
End Of Back off - AC3:   3606.6296296296305
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      3634.6296296296305
End Of CTS Timeout:      1.0E9
End Of Simulation:       1000000.0

Event Number: 10
Event Type:   END OF BACK-OFF (AC0)
System Clock: 3593.6296296296305
```

Figure A.17: EDCA End of Back-off (AC0) event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          4645.0
Observation:             100000.0
End Of AIFS - AC0:       1.0E9
End Of AIFS - AC1:       1.0E9
End Of AIFS - AC2:       1.0E9
End Of AIFS - AC3:       1.0E9
End Of Back off - AC0:   1.0E9
End Of Back off - AC1:   4644.9629629629635
End Of Back off - AC2:   1.0E9
End Of Back off - AC3:   4918.222222222223
End Of Transmission:     4882.555555555556
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      4931.555555555556
End Of CTS Timeout:      1.0E9
End Of Simulation:       1000000.0

Event Number: 11
Event Type:   END OF BACK-OFF (AC1)
System Clock: 4644.9629629629635
```

Figure A.18: EDCA End of Back-off (AC1) event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          3909.0
Observation:             100000.0
End Of AIFS - AC0:       1.0E9
End Of AIFS - AC1:       1.0E9
End Of AIFS - AC2:       1.0E9
End Of AIFS - AC3:       1.0E9
End Of Back off - AC0:   3974.8148148148143
End Of Back off - AC1:   3970.8148148148143
End Of Back off - AC2:   3906.8148148148143
End Of Back off - AC3:   3915.8148148148143
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      1.0E9
End Of CTS Timeout:      1.0E9
End Of Simulation:       10000.0

Event Number: 12
Event Type:   END OF BACK-OFF (AC2)
System Clock: 3906.8148148148143
```

Figure A.19: EDCA End of Back-off (AC2) event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          53.0
Observation:             100000.0
End Of AIFS - AC0:       79.0
End Of AIFS - AC1:       87.0
End Of AIFS - AC2:       69.0
End Of AIFS - AC3:       1.0E9
End Of Back off - AC0:   1.0E9
End Of Back off - AC1:   1.0E9
End Of Back off - AC2:   1.0E9
End Of Back off - AC3:   52.0
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      1.0E9
End Of CTS Timeout:      1.0E9
End Of Simulation:       10000.0

Event Number: 13
Event Type:   END OF BACK-OFF (AC3)
System Clock: 52.0
```

Figure A.20: EDCA End of Back-off (AC3) event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          79.0
Observation:             100000.0
End Of AIFS - AC0:       1.0E9
End Of AIFS - AC1:       1.0E9
End Of AIFS - AC2:       1.0E9
End Of AIFS - AC3:       1.0E9
End Of Back off - AC0:   1.0E9
End Of Back off - AC1:   1.0E9
End Of Back off - AC2:   1.0E9
End Of Back off - AC3:   1.0E9
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              78.66666666666667
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      1.0E9
End Of CTS Timeout:      128.66666666666666
End Of Simulation:       1000000.0

Event Number: 14
Event Type:   END OF RTS
System Clock: 78.66666666666667
```

Figure A.21: EDCA End of RTS event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          98.0
Observation:             100000.0
End Of AIFS - AC0:       1.0E9
End Of AIFS - AC1:       1.0E9
End Of AIFS - AC2:       1.0E9
End Of AIFS - AC3:       1.0E9
End Of Back off - AC0:   1.0E9
End Of Back off - AC1:   1.0E9
End Of Back off - AC2:   1.0E9
End Of Back off - AC3:   1.0E9
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              97.15555555555557
End Of ACK:              1.0E9
End Of ACK Timeout:      1.0E9
End Of CTS Timeout:      169.33333333333334
End Of Simulation:       1000000.0

Event Number: 15
Event Type:   END OF CTS
System Clock: 97.15555555555557
```

Figure A.22: EDCA End of CTS event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          95.0
Observation:             100000.0
End Of AIFS - AC0:       1.0E9
End Of AIFS - AC1:       1.0E9
End Of AIFS - AC2:       1.0E9
End Of AIFS - AC3:       1.0E9
End Of Back off - AC0:   1.0E9
End Of Back off - AC1:   1.0E9
End Of Back off - AC2:   1.0E9
End Of Back off - AC3:   1.0E9
End Of Transmission:     1.0E9
End Of SIFS:             94.66666666666667
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      1.0E9
End Of CTS Timeout:      128.66666666666666
End Of Simulation:       1000000.0

Event Number: 16
Event Type:   END OF SIFS
System Clock: 94.66666666666667
```

Figure A.23: EDCA End of SIFS event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          177.0
Observation:             100000.0
End Of AIFS - AC0:       1.0E9
End Of AIFS - AC1:       1.0E9
End Of AIFS - AC2:       1.0E9
End Of AIFS - AC3:       1.0E9
End Of Back off - AC0:   1.0E9
End Of Back off - AC1:   1.0E9
End Of Back off - AC2:   1.0E9
End Of Back off - AC3:   1.0E9
End Of Transmission:     176.30370370370372
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              194.0
End Of ACK:              1.0E9
End Of ACK Timeout:      225.30370370370372
End Of CTS Timeout:      244.0
End Of Simulation:       1000000.0

Event Number: 17
Event Type:   END OF TRANSMISSION
System Clock: 176.30370370370372
```

Figure A.24: EDCA End of Transmission event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          3586.0
Observation:             100000.0
End Of AIFS - AC0:       1.0E9
End Of AIFS - AC1:       1.0E9
End Of AIFS - AC2:       1.0E9
End Of AIFS - AC3:       1.0E9
End Of Back off - AC0:   1.0E9
End Of Back off - AC1:   1.0E9
End Of Back off - AC2:   1.0E9
End Of Back off - AC3:   1.0E9
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              3585.6296296296305
End Of ACK Timeout:      3634.6296296296305
End Of CTS Timeout:      1.0E9
End Of Simulation:       1000000.0

Event Number: 18
Event Type:   END OF ACK
System Clock: 3585.6296296296305
```

Figure A.25: EDCA End of ACK event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          3755.0
Observation:             100000.0
End Of AIFS - AC0:       3754.8888888888896
End Of AIFS - AC1:       1.0E9
End Of AIFS - AC2:       1.0E9
End Of AIFS - AC3:       1.0E9
End Of Back off - AC0:   1.0E9
End Of Back off - AC1:   3822.8888888888896
End Of Back off - AC2:   3839.8888888888896
End Of Back off - AC3:   3767.8888888888896
End Of Transmission:     1.0E9
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      3754.222222222223
End Of CTS Timeout:      1.0E9
End Of Simulation:       1000000.0

Event Number: 19
Event Type:   END OF ACK TIMEOUT
System Clock: 3754.222222222223
```

Figure A.26: EDCA End of ACK Timeout event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          170.0
Observation:             100000.0
End Of AIFS - AC0:       1.0E9
End Of AIFS - AC1:       1.0E9
End Of AIFS - AC2:       1.0E9
End Of AIFS - AC3:       1.0E9
End Of Back off - AC0:   1.0E9
End Of Back off - AC1:   1.0E9
End Of Back off - AC2:   1.0E9
End Of Back off - AC3:   1.0E9
End Of Transmission:     249.48888888888888
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      298.4888888888889
End Of CTS Timeout:      169.33333333333334
End Of Simulation:       1000000.0

Event Number: 20
Event Type:   END OF CTS TIMEOUT
System Clock: 169.33333333333334
```

Figure A.27: EDCA End of CTS Timeout event output snippet

```
..............<EVENTS CALENDAR>.......................
Packet Arrival:          1000001.0
Observation:             1100000.0
End Of AIFS - AC0:       1.0E9
End Of AIFS - AC1:       1.0E9
End Of AIFS - AC2:       1.0E9
End Of AIFS - AC3:       1.0E9
End Of Back off - AC0:   1.0E9
End Of Back off - AC1:   1.0E9
End Of Back off - AC2:   1.0E9
End Of Back off - AC3:   1.0E9
End Of Transmission:     1000038.0370370299
End Of SIFS:             1.0E9
End Of RTS:              1.0E9
End Of CTS:              1.0E9
End Of ACK:              1.0E9
End Of ACK Timeout:      1000087.0370370299
End Of CTS Timeout:      1.0E9
End Of Simulation:       1000000.0

Event Number: 21
Event Type:   END OF SIMULATION
System Clock: 1000000.0
```

Figure A.28: EDCA End of Simulation event output snippet

A.2 CHANNEL MODEL TESTING

This section shows the output from the channel model.

```
Bit Energy-To-Noise Ratio: 45
Signal-To-Noise Ratio: 43.12913356642856
```

Figure A.29: Channel Model Step 1

```
Packet Bits...
0     0     1     1     0     1     1     1     1     1     0     1     0     1
      1     1     0     0     0     1     0     0     1     1     0     0     1
      1     0     0     0     1     1     1     1     0     1     0     1     0
      0     1     0     0     0     0     0     1     1     1     0     1     1
      1     0     1     1     1     0     1     1     1     0     0     0     0
      1     1     1     1     0     1     0     1     1     0     0     1     0
      1     1     0     0     1     1     0     1     1     0     0     1     1
      0     1     0     1     0     1     1     1     1     1     1     1
```

Figure A.30: Channel Model Step 2

```
BPSK Modulation...
-1    -1    1     1     -1    1     1     1     1     1     -1    1     -1    1
      1     1     -1    -1    -1    1     -1    -1    1     1     -1    -1    1
      1     -1    -1    -1    1     1     1     1     -1    1     -1    1     -1
      -1    1     -1    -1    -1    -1    -1    1     1     1     -1    1     1
      1     -1    1     1     1     -1    1     1     1     -1    -1    -1    -1
      1     1     1     1     -1    1     -1    1     1     -1    -1    1     -1
      1     1     -1    -1    1     1     -1    1     1     -1    -1    1     1
      -1    1     -1    1     -1    1     1     1     1     1     1     1
```

Figure A.31: Channel Model Step 3

```
Sub-carriers...
0     0     0     0     0     0     -1    -1    1     1     -1    1     1     1
      1     1     -1    1     -1    1     1     1     -1    -1    -1    1     -1
      -1    1     1     -1    -1    0     1     1     -1    -1    -1    1     1
      1     1     -1    1     -1    1     -1    -1    1     -1    -1    -1    -1
      -1    1     1     1     -1    1     0     0     0     0     0
0     0     0     0     0     0     1     1     -1    1     1     1     -1    1
      1     1     -1    -1    -1    -1    1     1     1     1     -1    1     -1
      1     1     -1    -1    1     0     -1    1     1     -1    -1    1     1
      -1    1     1     -1    -1    1     1     -1    1     -1    1     -1    1
      1     1     1     1     1     1     0     0     0     0     0
```

Figure A.32: Channel Model Step 4

```
IFFT...
1.941450686788302
-0.6387596283357257 + 0.24951357631795137i
-0.2444419313879624 - 0.39613983070609804i
0.7608369790264369 + 0.008490151146014643i
-1.5022098546074925 + 0.004953579239497318i
-0.06916212047776801 + 0.20291702778328286i
0.6527298637707248 + 0.8031866044957593i
0.32543077313315194 - 0.49414970313190204i
```

Figure A.33: Channel Model Step 5

```
Appending Cyclic Prefix...
-1.386750490563073 + 0.2773500981126146i
0.6092158941409059 + 0.5152083054873896i
0.1858755079009978 - 0.33421170461629285i
-0.30467220294061353 + 0.29209444860037914i
0.3951340098125756 + 0.48757283343438446i
0.18490984405309382 - 0.86310755636393i
-0.050121652789272164 - 1.4237039251123642i
-0.5712115166008485 - 0.6532242622508235i
0.11488217216375357 + 0.8656985035271666i
```

Figure A.34: Channel Model Step 6

```
Calculate Frequency Response...
-0.9062249386460717 - 1.8741022555386748i
-1.725490585709157 - 1.2481733088840871i
-2.0330871725851734 - 0.23349510487330938i
-1.7014019112974648 + 0.8238617127831804i
-0.804354558263267 + 1.5615533448408838i
0.39577335651380574 + 1.7021853877599007i
1.5059748438083576 + 1.1475563791119134i
```

Figure A.35: Channel Model Step 7

```
Convolution of each OFDM symbol...
-0.18145973574433533 - 1.4123358795295 22i
0.05496726350905767 + 1.2222420658049167i
0.3104558465469561 + 0.2757505192146983i
-0.7788475107755579 - 0.8712052207780809i
-0.7706165219540315 + 1.8050099198963685i
1.773953465914725 - 0.5861289637036461i
0.5167689585640363 + 0.807619230915806i
```

Figure A.36: Channel Model Step 8

```
Form a long transmission sequence...
-0.18145973574433533 - 1.4123358795295 22i
0.05496726350905767 + 1.2222420658049167i
0.3104558465469561 + 0.2757505192146983i
-0.7788475107755579 - 0.8712052207780809i
-0.7706165219540315 + 1.8050099198963685i
1.773953465914725 - 0.5861289637036461i
0.5167689585640363 + 0.807619230915806i
-0.4139430715556952 - 0.7921884589898532i
-1.3064879197139216 - 1.0820604250408545i
```

Figure A.37: Channel Model Step 9

```
Add White Gaussian Noise...
-0.2034906887296525 - 1.5812278064328926i
0.06684569209441052 + 1.3625419476127183i
0.34000290791736026 + 0.3057927601428751i
-0.8733744122888518 - 0.9686377267459092i
-0.8607902752975838 + 2.0225562921021503i
1.9730380595940418 - 0.658353201656987i
0.5913183240867741 + 0.90705283290490 58i
-0.46395863441838375 - 0.88873902351829 07i
-1.4636420321910342 - 1.2122498423047685i
```

Figure A.38: Channel Model Step 10

```
Removing Cyclic Prefix...
-2.2191283079116455 + 4.2892096000470685i
-0.8073230836189235 - 2.0177943109223153i
2.0207378034343897 + 1.1312841558622229i
-0.27457298337288755 + 0.5727635383578964i
0.9281010596615136 - 4.949017888736768i
-0.353085831359541 + 2.850548425721655i
-1.2637764365004103 - 0.8940076510895494i
1.816528721799689 - 0.025964778642544227i
-0.8721767396964311 + 3.591504656213106i
```

Figure A.39: Channel Model Step 11

```
Converting time domain to frequency domain...
0.0012719312289897627 - 0.004786833741897156i
-0.00920649332369666 + 5.015387830790865E-4i
-0.00804564598061586 + 0.0044329426187211 87i
-0.0055584125828503405 - 0.0012986125745928306i
0.0013894975588658294 - 0.0018528400057007174i
-0.005198267495299705 + 0.005745091138762 68i
-1.675681033774673 - 1.2879123970652624i
-2.367977329705621 - 0.042859011800868374i
2.1703778178201203 - 1.354731937014793i
```

Figure A.40: Channel Model Step 12

```
Equalising by the known channel frequency response...
0.001804166349135028 + 0.0015510956029485298i
0.0033646867384252224 - 0.002724588010812087i
0.003658682685862933 - 0.002600590465303038i
0.002347048463744297 + 0.0018997603802228407i
-0.001299964856849225 - 2.2020695086285278E-4i
0.0025283867941261326 + 0.0036417511676175712i
-1.1162224852581997 - 0.0046376362976952 02i
-1.1207646606270698 + 0.0038066918513457 76i
1.1204536311199593 + 9.442265601030675E-4i
```

Figure A.41: Channel Model Step 13

```
Extracting the required sub-carriers...
-1.1162224852581997 - 0.0046376362976952O2i
-1.1207646606270698 + 0.0038066918513457 76i
1.1204536311199593 + 9.442265601030675E-4i
1.1177443977143415 - 9.799816353010526E-4i
-1.1154510787026433 + 6.471298121794311E-4i
1.1255550204922722 - 4.508169285305508E-4i
1.1157510307800642 - 0.0019478964538824362i
1.1226301575755722 - 0.0014431042656824489i
1.1177951713846488 - 6.519600599385056E-4i
```

Figure A.42: Channel Model Step 14

```
Get Received BPSK...
-1    -1    1     1     -1    1     1     1     1     1     -1    1     -1    1
      1     1     -1    -1    -1    1     -1    -1    1     1     -1    -1    1
      1     -1    -1    -1    1     1     1     1     -1    1     -1    1     -1
      -1    1     -1    -1    -1    -1    -1    1     1     1     -1    1
1     1     -1    1     1     1     -1    1     1     1     -1    -1    -1    -1
      1     1     1     1     -1    1     -1    1     1     -1    -1    1     -1
      1     1     -1    -1    1     1     -1    1     1     -1    -1    1     1
      -1    1     -1    1     -1    1     1     1     1     1     1     1
```

Figure A.43: Channel Model Step 15

```
BPSK Demodulation...
0     0     1     1     0     1     1     1     1     1     0     1     0     1
      1     1     0     0     0     1     0     0     1     1     0     0     1
      1     0     0     0     1     1     1     1     0     1     0     1     0
      0     1     0     0     0     0     0     1     1     1     0     1     1
      1     0     1     1     1     0     1     1     1     0     0     0     0
      1     1     1     1     0     1     0     1     1     0     0     1     0
      1     1     0     0     1     1     0     1     1     0     0     1     1
      0     1     0     1     0     1     1     1     1     1     1     1
```

Figure A.44: Channel Model Step 16

```
Counting the errors...
Number of Errors: 0
```

Figure A.45: Channel Model Step 17

A.3 WORKLOAD MODEL TESTING

Testing for the workload model was carried out to see if the model produces IATs and packet sizes. Four tests were carried out, one for each traffic type. Figure A.46 shows test results for the VoIP model, Figure A.47 shows results for the video model, Figure A.48 shows results for the best-effort P2P model, and Figure A.49 shows results for the background HTTP model.

All the tests passed, and models produce the expected results. A long sequence of IATs for each traffic type showed that the traffic generators followed the correct distributions, but the sequence was too long to put in this section

```
IAT   Packet Size
78.0  1720
63.0  6192
3.0   1432
9.0   3240
39.0  2096
39.0  9568
6.0   2152
97.0  10760
66.0  4320
18.0  9248
40.0  3848
146.0 10168
```

Figure A.46: MMPP VoIP workload model output snippet

```
IAT   Packet Size
70.0  9472
18.0  7424
33.0  2040
4.0   9376
142.0 2528
148.0 11152
140.0 9688
95.0  4904
51.0  1112
89.0  2336
99.0  2232
105.0 10168
```

Figure A.47: Log-normal video workload model output snippet

```
IAT   Packet Size
69.0  10760
23.0  3848
106.0 6056
32.0  4456
31.0  7392
12.0  1136
82.0  952
91.0  9752
110.0 11912
78.0  8688
136.0 11664
108.0 8008
```

Figure A.48: Pareto P2P workload model output snippet

```
IAT   Packet Size
58.0  9288
66.0  2328
73.0  8816
62.0  3440
67.0  9240
81.0  5696
100.0 6096
46.0  4448
41.0  7456
35.0  1008
12.0  5808
22.0  3432
```

Figure A.49: Weibull HTTP workload model output snippet

APPENDIX B

EXPERIMENTATION RESULTS TABLES

B.1 SCALABILITY EXPERIMENTS

No. of STAs	DCF Eb/NO 30	Lower Conf.	Upper Conf.	DCF Eb/NO 50	Lower Conf.	Upper Conf.
1	9.2932	8.2749	10.3115	19.2593	17.3608	21.1579
2	13.6997	13.0426	14.3568	28.7195	27.4042	30.0349
3	14.6854	14.2336	15.1372	32.0596	31.4192	32.7000
4	14.9199	14.5651	15.2747	32.2323	31.6246	32.8400
5	14.6744	14.3299	15.0188	31.7157	31.2620	32.1694
6	14.6863	14.2833	15.0894	30.9601	30.4267	31.4935
7	14.4867	14.1448	14.8285	30.0957	29.6384	30.5530
8	14.4877	14.1780	14.7975	29.2872	28.8473	29.7272
9	14.3607	14.0469	14.6746	29.0498	28.6310	29.4686
10	14.3953	14.0672	14.7234	28.0867	27.4736	28.6998
11	14.0860	13.8137	14.3583	27.9981	27.4949	28.5012
12	14.1579	13.9057	14.4101	27.4478	27.1418	27.7539
13	14.0101	13.7054	14.3149	27.2489	26.8097	27.6881
14	13.8177	13.4813	14.1541	27.0558	26.6768	27.4348
15	13.9021	13.6112	14.1929	26.4775	26.1290	26.8260
16	13.7146	13.4107	14.0185	26.3627	26.0130	26.7124

Table B.1: Normalised Aggregate Throughput percentage for DCF for an increasing number of STAs for ideal channel conditions, with 99 percent confidence interval, for EbNO of 30 and 50

No. of STAs	EDCA Eb/NO 30	Lower Conf.	Upper Conf.	EDCA Eb/NO 50	Lower Conf.	Upper Conf.
1	29.8039	29.7581	29.8497	34.3862	34.3447	34.4276
2	30.2876	30.1983	30.3768	35.4001	35.1477	35.6525
3	30.8500	30.6690	31.0309	36.1182	35.8116	36.4248
4	31.0936	30.9252	31.2619	37.2608	36.7137	37.8079
5	31.3388	31.1601	31.5175	37.4608	37.0015	37.9201
6	31.8546	31.6352	32.0739	38.5254	38.0034	39.0475
7	32.0844	31.8992	32.2696	38.9385	38.2534	39.6236
8	32.3820	32.1336	32.6303	39.8511	39.2356	40.4666
9	32.6636	32.3968	32.9303	41.3057	40.4368	42.1747
10	32.8329	32.5566	33.1092	41.9075	41.1691	42.6460
11	33.3227	33.0065	33.6389	42.7703	41.9851	43.5556
12	32.7944	31.1276	34.4612	42.9520	41.4193	44.4846
13	33.0710	31.0984	35.0436	42.8885	38.7247	47.0522
14	34.0492	33.7844	34.3140	42.9638	41.0226	44.9049
15	32.8425	30.0298	35.6553	43.1541	40.1036	46.2046
16	33.1417	30.3037	35.9796	42.6449	37.8924	47.3975

Table B.2: Normalised Aggregate Throughput percentage for EDCA for an increasing number of STAs for ideal channel conditions, with 99 percent confidence interval, for Eb/NO of 30 and 50

No. of STAs	DCF Eb/NO 30	Lower Conf.	Upper Conf.	DCF Eb/NO 50	Lower Conf.	Upper Conf.
1	0	0	0	0	0	0
2	2.6623	2.3366	2.9880	3.4933	3.1318	3.8548
3	5.4448	5.0234	5.8661	7.0898	6.6546	7.5249
4	8.0896	7.6890	8.4901	10.2480	9.7440	10.7521
5	10.0819	9.5659	10.5978	12.7363	12.2486	13.2240
6	12.2854	11.8443	12.7265	14.7930	14.3538	15.2323
7	13.9752	13.5528	14.3976	17.0369	16.5875	17.4863
8	15.4736	15.0852	15.8620	18.6800	18.3520	19.0080
9	16.8729	16.5708	17.1750	19.6668	19.3350	19.9986
10	18.2385	17.8660	18.6110	21.1110	20.7855	21.4366
11	19.6527	19.2612	20.0441	22.1207	21.7845	22.4570
12	20.4236	20.1134	20.7337	23.0492	22.7223	23.3761
13	21.4507	21.1230	21.7783	23.9975	23.7879	24.2071
14	22.2317	21.9631	22.5003	24.8674	24.6319	25.1029
15	23.2387	22.9550	23.5224	25.5934	25.3404	25.8465
16	23.8206	23.5093	24.1320	26.2977	26.0917	26.5038

Table B.3: Collision Probability percentage for DCF for an increasing number of STAs for ideal channel conditions, with 99 percent confidence interval, for Eb/NO of 30 and 50

No. of STAs	EDCA Eb/NO 30	Lower Conf.	Upper Conf.	EDCA Eb/NO 50	Lower Conf.	Upper Conf.
2	0.2305	0.2103	0.2507	0.2290	0.2117	0.2463
3	0.4187	0.3907	0.4466	0.4039	0.3796	0.4281
4	0.5912	0.5575	0.6248	0.6066	0.5804	0.6329
5	0.8093	0.7773	0.8414	0.8032	0.7608	0.8457
6	1.0105	0.9612	1.0598	1.0175	0.9797	1.0553
7	1.2305	1.1712	1.2897	1.2221	1.1719	1.2723
8	1.5401	1.4630	1.6171	1.4570	1.3952	1.5187
9	1.7507	1.6774	1.8240	1.7468	1.6821	1.8115
10	2.0837	2.0048	2.1625	1.9609	1.8843	2.0375
11	2.3917	2.3060	2.4774	2.2158	2.1488	2.2828
12	2.6599	2.5065	2.8132	2.5155	2.4352	2.5957
13	2.9874	2.8278	3.1469	2.8557	2.6466	3.0648
14	3.2973	3.1176	3.4170	3.1074	2.9908	3.2241
15	3.6698	3.4225	3.9172	3.4565	3.4412	3.5718
16	4.0092	3.6226	4.3958	3.8131	3.4884	4.1379

Table B.4: Collision Probability percentage for EDCA for an increasing number of STAs for ideal channel conditions, with 99 percent confidence interval, for Eb/NO of 30 and 50

No. of STAs	DCF Eb/NO 30	Lower Conf.	Upper Conf.	DCF Eb/NO 50	Lower Conf.	Upper Conf.
1	523	502	545	269	267	272
2	795	750	840	391	377	405
3	1,199	1,139	1,260	575	551	598
4	1,618	1,529	1,708	789	753	826
5	2,064	1,973	2,155	1,017	969	1,065
6	2,534	2,444	2,624	1,265	1,219	1,311
7	3,042	2,914	3,170	1,588	1,523	1,653
8	3,427	3,331	3,522	1,881	1,816	1,947
9	3,970	3,823	4,117	2,102	2,052	2,153
10	4,379	4,207	4,551	2,405	2,350	2,461
11	4,857	4,704	5,010	2,679	2,612	2,747
12	5,305	5,127	5,483	2,951	2,861	3,040
13	5,629	5,471	5,766	3,218	3,146	3,290
14	6,211	5,997	6,424	3,522	3,469	3,574
15	6,611	6,398	6,825	3,794	3,701	3,887
16	7,159	6,975	7,342	4,022	3,948	4,096

Table B.5: Access Delay in μs for DCF for an increasing number of STAs for ideal channel conditions, with 99 percent confidence interval, for Eb/NO of 30 and 50

No. of STAs	EDCA Eb/NO 30	Lower Conf.	Upper Conf.	EDCA Eb/NO 50	Lower Conf.	Upper Conf.
1	88	88	88	77	77	77
2	260	213	308	223	169	277
3	524	353	695	337	276	398
4	672	494	849	578	423	733
5	797	640	954	672	569	775
6	1,254	966	1,542	999	811	1,187
7	1,592	1,295	1,889	1,280	1,038	1,521
8	2,328	1,919	2,737	1,645	1,391	1,900
9	2,313	1,891	2,734	2,288	1,939	2,637
10	2,933	2,520	3,347	2,445	2,051	2,838
11	3,507	3,014	3,999	2,671	2,390	2,951
12	3,690	3,025	4,355	2,965	2,605	3,326
13	4,476	3,804	5,148	3,384	2,887	3,880
14	4,708	4,163	5,254	3,421	3,118	3,723
15	4,567	4,008	5,126	3,730	3,323	4,136
16	5,922	5,296	6,548	4,194	3,609	4,780

Table B.6: Access Delay in μs for EDCA for an increasing number of STAs for ideal channel conditions, with 99 percent confidence interval, for Eb/NO of 30 and 50

B.2 COMPARISON EXPERIMENTS

Eb/NO	DCF Voice	Lower Conf.	Upper Conf.	DCF Video	Lower Conf.	Upper Conf.
20	0.0823	0.0699	0.0947	0.0849	0.0750	0.0946
25	0.9587	0.9063	1.0110	1.0894	1.0233	1.1556
30	3.1902	3.0785	3.3019	4.4751	4.3326	4.6175
35	4.8467	4.7444	4.9490	8.2343	8.0290	8.4396
40	5.5359	5.4280	5.6438	9.5996	9.2257	9.9735
45	5.8186	5.7083	5.9289	10.1975	9.9362	10.4589
50	5.8859	5.7629	6.0089	9.9514	9.5455	10.3573

Table B.7: Normalised Aggregate Throughput percentage against increasing Bit-Energy-To-Noise Ratio for DCF voice and video, with 99 percent confidence interval

Eb/NO	EDCA Voice	Lower Conf.	Upper Conf.	EDCA Video	Lower Conf.	Upper Conf.
20	5.1854	4.0299	6.3409	0.6010	0.1861	1.0160
25	15.2591	13.3101	17.2081	4.1287	3.0391	5.2183
30	26.6504	24.3894	27.9113	5.8639	4.4955	7.2323
35	30.4334	27.9814	32.8854	6.3564	5.2005	7.5123
40	32.8694	32.2086	33.5303	8.0123	6.6886	9.3361
45	33.3002	31.3186	35.2817	6.3986	4.5278	8.2695
50	33.5602	32.8585	34.2619	8.0045	6.5963	9.4126

Table B.8: Normalised Aggregate Throughput percentage against increasing Bit-Energy-To-Noise Ratio for EDCA voice and video, with 99 percent confidence interval

Eb/NO	DCF	Lower Conf.	Upper Conf.	EDCA	Lower Conf.	Upper Conf.
20	0.3429	0.3166	0.3693	5.8001	4.9408	6.6593
25	4.0805	3.8846	4.2763	19.5195	18.6178	20.4212
30	14.3953	14.0672	14.7234	32.8329	32.5566	33.1092
35	23.2394	22.8591	23.6197	37.0824	34.3475	39.8174
40	26.7815	26.2563	27.3067	41.2017	40.5330	41.8705
45	28.1914	27.8314	28.5515	40.0629	37.6815	42.4443
50	28.0867	27.4736	28.6998	41.9075	41.1691	42.6460

Table B.9: Normalised Aggregate Throughput percentage against increasing Bit-Energy-To-Noise Ratio for EDCA and DCF integrated traffic, with 99 percent confidence interval

Eb/NO	DCF Voice	Lower Conf.	Upper Conf.	DCF Video	Lower Conf.	Upper Conf.
20	19.0826	14.3500	23.8152	20.0920	16.2014	23.9826
25	16.0194	14.4103	17.6285	16.8291	15.3625	18.2957
30	21.0691	20.1501	21.9880	22.7302	21.8397	23.6207
35	24.3411	23.5775	25.1047	24.9902	24.2892	25.6911
40	25.2671	24.7118	25.8224	25.5456	25.0197	26.0716
45	25.7778	25.1010	26.4546	25.3000	24.5695	26.0305
50	25.4580	24.8698	26.0463	25.4507	24.7754	26.1261

Table B.10: Collision Probability percentage against increasing Bit-Energy-To-Noise Ratio for DCF voice and video, with 99 percent confidence interval

Eb/NO	EDCA Voice	Lower Conf.	Upper Conf.	EDCA Video	Lower Conf.	Upper Conf.
20	11.1684	10.1784	12.1584	19.7759	18.6959	20.8559
25	2.3002	2.0354	2.5649	3.5000	2.9007	4.0992
30	2.0158	1.9359	2.0957	1.8808	1.3369	2.4248
35	1.9560	1.8515	2.0606	1.6360	1.2776	1.9943
40	1.9205	1.8491	1.9920	1.4438	1.1354	1.7522
45	1.8811	1.8073	1.9549	1.5853	1.1950	1.9756
50	1.9271	1.8520	2.0022	1.5222	1.2123	1.8322

Table B.11: Collision Probability percentage against increasing Bit-Energy-To-Noise Ratio for EDCA voice and video, with 99 percent confidence interval

Eb/NO	DCF	Lower Conf.	Upper Conf.	EDCA	Lower Conf.	Upper Conf.
20	15.9563	14.5112	17.4014	11.8608	8.6552	15.0665
25	14.1026	13.6093	14.4958	2.4799	2.2863	2.6734
30	18.2385	17.8660	18.6110	2.0837	2.0048	2.1625
35	20.2159	19.9008	20.5310	2.0021	1.8922	2.1120
40	20.8780	20.6332	21.1227	1.9579	1.8814	2.0345
45	20.9133	20.5869	21.2397	1.9331	1.8597	2.0064
50	21.1110	20.7855	21.4366	1.9609	1.8843	2.0375

Table B.12: Collision Probability percentage against increasing Bit-Energy-To-Noise Ratio for EDCA and DCF integrated traffic, with 99 percent confidence interval

Eb/NO	DCF Voice	Lower Conf.	Upper Conf.	DCF Video	Lower Conf.	Upper Conf.
20	166,772	150,590	182,954	159,799	147,093	172,504
25	20,141	18,762	21,519	23,264	21,722	24,806
30	4,894	4,658	5,130	6,791	6,256	7,326
35	2,963	2,823	3,102	3,646	3,460	3,832
40	2,676	2,533	2,819	2,806	2,704	2,908
45	2,580	2,447	2,714	2,649	2,513	2,784
50	2,534	2,437	2,631	2,548	2,447	2,649

Table B.13: Access Delay in μs against increasing Bit-Energy-To-Noise Ratio for DCF voice and video, with 99 percent confidence interval

Eb/NO	EDCA Voice	Lower Conf.	Upper Conf.	EDCA Video	Lower Conf.	Upper Conf.
20	10,210	548	19,873	42,103	10,034	74,173
25	2,952	1,068	4,837	23,092	16,578	29,606
30	1,149	1,079	1,218	23,118	18,378	27,859
35	919	890	949	20,771	15,844	25,699
40	901	878	923	21,341	17,972	24,709
45	864	837	892	17,011	12,757	21,266
50	874	856	892	19,200	15,184	23,217

Table B.14: Access Delay in μs against increasing Bit-Energy-To-Noise Ratio for EDCA voice and video, with 99 percent confidence interval

Eb/NO	DCF	Lower Conf.	Upper Conf.	EDCA	Lower Conf.	Upper Conf.
20	76,533	70,080	82,985	9,786	6,612	12,960
25	12,735	12,044	13,426	5,125	3,634	6,615
30	4,379	4,207	4,551	2,933	2,520	3,347
35	2,859	2,782	2,936	2,523	2,133	2,913
40	2,501	2,437	2,565	2,505	2,192	2,818
45	2,414	2,340	2,488	2,104	1,753	2,455
50	2,405	2,350	2,461	2,445	2,051	2,838

Table B.15: Access Delay in μs against increasing Bit-Energy-To-Noise Ratio for EDCA and DCF integrated traffic, with 99 percent confidence interval