11-30-2012 12:00 AM

Situation-Aware Rate and Power Adaptation Techniques for IEEE 802.11

Jay Nadeau
The University of Western Ontario

Supervisor
Dr. Xianbin Wang
The University of Western Ontario

Graduate Program in Electrical and Computer Engineering
A thesis submitted in partial fulfillment of the requirements for the degree in Master of Engineering Science

Situation-Aware Rate and Power Adaptation Techniques for IEEE 802.11

(Spine title: Situation-Aware Rate and Power Adaptation Techniques for IEEE 802.11)

(Thesis format: Monograph)

by

Jay Richard Nadeau

Graduate Program
in
Engineering Science
Electrical and Computer Engineering

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Engineering Science

School of Graduate and Postdoctoral Studies
The University of Western Ontario
London, Ontario, Canada

THE UNIVERSITY OF WESTERN ONTARIO
SCHOOL OF GRADUATE AND POSTDOCTORAL STUDIES
CERTIFICATE OF EXAMINATION

Chief Advisor:	**Examining Board:**
Dr. Xianbin Wang	Dr. Jagath Samarabandu
Advisory Committee:	Dr. Raveendra K. Rao
	Dr. Evgueni Bordatchev

The thesis by
Jay Richard Nadeau
entitled:
Situation-Aware Rate and Power Adaptation Techniques for IEEE 802.11
is accepted in partial fulfillment of the
requirements for the degree of
Master of Engineering Science

Date: ____________________

Chair of Examining Board
Dr. Serguei Primak

Abstract

The current generation of IEEE 802.11 Wireless Local Area Networks (WLANs) provide multiple data rates from which the different physical (PHY) layers may choose. The rate adaptation algorithm (RAA) is an essential component of 802.11 WLANs which completely determines the data rate a device may use. Some of the key challenges facing data rate selection are the constantly varying wireless channel, selecting the data rate that will result in the maximum throughput, assessing the conditions based on limited feedback and estimating the link conditions at the receiver.

Current RAAs lack the ability to sense their environment and adapt accordingly. 802.11 WLANs are deployed in many locations and use the same technique to choose the data rate in all locations and situations. Therefore, these RAAs suffer from the inability to adapt the method they use to choose the data transmission rate. In this thesis, a new RAA for 802.11 WLANs is proposed which provides an answer to the many challenges faced by RAAs. The proposed RAA is termed SARA which stands for Situation-Aware Rate Adaptation, and combines the use of the received signal strength and packet error rate to enable situational awareness. SARA adapts to the current environmental situation experienced at the moment to rapidly take advantage of changing channel conditions.

In addition to SARA, a method to optimize the transmission power for, but not limited to, IEEE 802.11 WLANs is proposed which can determine the minimum transmission power required by a station (STA) or base station (BS) for successful transmission of a data packet. The technique reduces the transmission power to the minimum level based on the current situation while maintaining QoS constraints. The method employs a Binary Search to quickly determine the minimum transmission power with low complexity and delay. Such a technique is useful to conserve battery life in mobile devices for 802.11 WLANs.

Both algorithms are implemented on an Atheros device driver for the FreeBSD operating system. SARA is compared to the benchmark algorithm SampleRate while an estimate of the energy consumed as well as the energy saved is provided for the minimum transmission power determination.

Acknowledgements

I present my sincerest appreciation to those who made this thesis possible. Specifically I would like to thank my parents and my girlfriend for their patience and support during my graduate studies, whom without, this work would not have been possible. I would also like to thank my colleagues Hao Li, Jason Liu and Kevin Mi for their encouragement and numerous conversations. I would also like to thank my supervisor Dr. Xianbin Wang for his guidance and direction during my graduate studies.

In addition, I would like to acknowledge the technicians at the electronics shop for their hardware expertise. I would also like to acknowledge the Faculty of Engineering, especially the Department of Electrical and Computer Engineering at Western University for the excellent facilities, libraries and resources available to graduate students.

Table of Contents

List of Tables

List of Figures

Acronyms

ACK	*Acknowledgement*
AMRR	*Adaptive Multi Rate Retry*
AP	*Access Point*
ARQ	*Automatic Repeat Request*
AWGN	*Additive White Gaussian Noise*
BER	*Bit Error Rate*
BICM	*Binary Interleaved Coded Modulation*
BPSK	*Binary Phase Shift Keying*
BS	*Base Station*
BSS	*Basic Service Set*
CP	*Cyclic Prefix*
CSI	*Channel State Information*
CP	*Distributed Interframe Space*
CRC	*Cyclic Redundancy Check*
DCF	*Distributed Coordination Function*
DFT	*Discrete Fourier Transform*
DS	*Distribution System*
DSSS	*Direct Sequence Spread Spectrum*
ESS	*Extended Service Set*
FCC	*Federal Communications Commission*
FCS	*Frame Check Sequence*
FFT	*Fast Fourier Transform*
FHSS	*Frequency Hopping Spread Spectrum*
FSK	*Frequency Shift Keying*
HCF	*Hybrid Coordination Function*
IBSS	*Independent Basic Service Set*
ICI	*Inter-carrier Interference*
IDFT	*Inverse Discrete Fourier Transform*

IEEE	*Institute of Electrical and Electronics Engineers*
IFFT	*Inverse Fast Fourier Transform*
IR	*Infrared*
ISI	*Inter-symbol Interference*
LLC	*Logical Link Control*
LOS	*Line of Sight*
MAC	*Medium Access Control*
MCS	*Modulation and Coding Scheme*
MI	*Mutual Information*
ML	*Maximum Likelihood*
MMSE	*Minimum Mean Squared Error*
MPDU	*MAC Protocol Data Unit*
MS	*Mobile Station*
NAV	*Network Allocation Vector*
NLOS	*Non Line of Sight*
OFDM	*Orthogonal Frequency Division Multiplexing*
OS	*Operating System*
PER	*Packet Error Rate*
PHY	*Physical Layer*
PLCP	*Physical Layer Convergence Procedure*
PMD	*Physical Medium Dependent*
PPDU	*PLCP Protocol Data Unit*
PSDU	*PLCP Service Data Unit*
PSK	*Phase Shift Keying*
QAM	*Quadrature Amplitude Modulation*
QoS	*Quality of Service*
QPSK	*Quadrature Phase Shift Keying*
RAA	*Rate Adaptation Algorithm*
RSSI	*Received Signal Strength Indicator*
RTT	*Round Trip Time*
SER	*Symbol Error Rate*
SIFS	*Short Interframe Space*

SINR	*Signal to Interference and Noise Ratio*
SNR	*Signal to Noise Ratio*
STA	*Station*
TPC	*Transmit Power Control*
WLAN	*Wireless Local Area Network*

Chapter 1

Introduction

High-speed communication has transformed the way we live and communicate, and the desire for faster communication continues to drive research. Always on and always available communication at any location is becoming a necessity in today's world and the growth in popularity of handheld wireless communication devices is proof of this. Furthermore, new applications such as streaming video, video phones and data intensive applications are constantly being developed to take advantage of newly developed wireless standards, hence, there is always a push to further increase the data rates available. Society demands high-speed wireless communication at all times and at all locations. One communication system that can provide that ability is the IEEE 802.11 WLAN standard. This standard provides wireless communication for a myriad of devices from laptops to cellular phones and will only become further integrated into our lives as more and more devices become wireless. Thus, the ability to ensure that the IEEE 802.11 standard is performing optimally becomes critical.

Many wireless protocols, including the IEEE 802.11 protocol, have multiple data transmission rates which transmitters may choose from. Higher quality channel conditions can support higher data rates to transmit more data, but may result in low throughput in poor channel conditions. Alternatively, lower data rates, in general, have a greater tolerance in poor channel conditions but increase the transmission time required to send the same amount of data. Each channel may support multiple data rates that successfully deliver packets, and the throughput a particular data rate achieves is a function of the current data rate and the delivery probability.

To adapt to varying channel conditions, the 802.11 PHYs provide multiple data transmission rates by employing different combinations of modulation and channel coding schemes. For example, the IEEE 802.11b PHY provides four PHY rates from 1 to 11 Mbps in the 2.4 GHz band, while the IEEE 802.11a PHY provides eight PHY rates ranging from 6 to 54 Mbps in the 5 GHz band. More recently, however, the IEEE 802.11g PHY was standardized which provides a combination of both 802.11b and 802.11a PHY rates for a total of twelve data transmission rates, ranging from 1 to 54 Mbps in the 2.4 GHz band. The 802.11n standard has also recently been standardized and provides even higher data rates up to 600 Mbps with up to 64 different rate combinations to choose from. Amazingly, the 802.11ac standard further improves upon 802.11n with wireless data transmission rates in the gigabit range. With each new standard, the data rate and number of data rates to choose from increases, thus making rate selection and therefore, the RAA, ever more important in order to take advantage of the new data rates as technology improves and provides greater throughput. Although each standard provides multiple PHY data transmission rates, the algorithm to choose the transmission rate is intentionally left unspecified by the standard and it becomes the responsibility of the chipset vendor to implement the RAA [2]. This has allowed for an area of open research and fostered increased competition among chipset vendors.

The mechanism to select one out of multiple available transmission rates at a given time is referred to as rate adaptation and the effectiveness of the implemented rate adaptation scheme can affect the system performance significantly. For example, due to the conservative nature of the rate adaptation schemes implemented in most 802.11 devices, the current 802.11 systems are likely to show low bandwidth utilization when the wireless channel presents a high degree of variation. Now, with up to twelve different transmission rates to choose from for 802.11g, the challenge for a rate adaptation algorithm design is to determine the best data rate for the transmission of a data packet.

Transmission of data at a higher PHY data rate will result in a shorter transmission time due to the increased number of bits transmitted in a single time interval. However, the use of a higher data rate will increase the probability of a transmission failure, due to the higher SNR requirements of a higher data rate, thus necessitating the use of a retransmission scheme. Thus, there is an inherent tradeoff between the number of data bits sent and the SNR or channel conditions and it is the role of the RAA to choose a rate that takes advantage of favourable channel conditions, while at the same time minimizing the number of retransmissions.

1.1 Motivation

In this section, the motivation for this thesis is described and is presented in three sub-sections, Adaptive Communications, Situation Awareness and Power Adaptation. Adaptive communications adapts the MCS to optimize the delivery of data based on changes in the radio environment [3]. Moreover, in order for a transmitter to fully utilize the wireless spectrum, the transmitter must be aware of its environment. The transmitter must recognize events and modify its behaviour, or data transmission rate, when necessary. The transmitter must automatically detect environmental changes and conditions and react accordingly. This detection and reaction ability is termed situation awareness and can allow an 802.11 transmitter to fully utilize the wireless spectrum.

Most modern communication systems are in one way or another adaptive and it is important for a communication system to be efficient in terms of spectrum use. However, it is not enough for a communication system to be adaptive, but situation aware as well. Adapting to the current conditions of a wireless channel is often performed in a generic way. That is, adaptation is performed regardless of the environment the communication system is operating in. A situation aware communication system adapts to its environment in different ways depending on the situation and can thus, increase the data rate during good channel conditions and increase robustness in poor channel conditions to maximize overall throughput in all situations.

In addition to improving the throughput, the reduction of transmission power can also be considered. Reducing transmission power can allow wireless devices operating on limited batter power to operate for longer periods of time, which, when combined with rate adaptation, can significantly improve the user experience. Furthermore, the frequency of device recharging can be reduced, potentially reducing energy requirements for the electrical grid when the sheer number of wireless or handheld devices is considered.

1.1.1 Adaptive Communications

The IEEE 802.11 standard employs multiple MCSs in order to accomplish the multiple data transmission rates available. Each data transmission rate uses a different combination of a modulation scheme and a coding scheme which results in various throughput levels or data rates. Table 1.1 shows a few modulation and coding schemes used in 802.11.

To demonstrate how different modulation schemes can be affected by varying

Modulation Scheme	Coding Rate	Bits per Symbol	Transmission Rate (Mbps)
BPSK	1/2	1	6
BPSK	3/4	1	9
QPSK	1/2	2	12
QPSK	3/4	2	18
16-QAM	1/2	4	24
16-QAM	3/4	4	36
64-QAM	2/3	6	48
64-QAM	3/4	6	54

Table 1.1: Various modulation and coding schemes used in 802.11.

channel quality, the different modulation schemes listed in Table 1.1 have been plotted under ideal and noisy channel conditions. Figures 1.1 - 1.4 show the results of a simulation using various modulation schemes and channels with different levels of AWGN.

The PHY layer encodes a sequence of bits into a PHY symbol which is represented by a position on a 2D complex plane called the constellation diagram. Figures

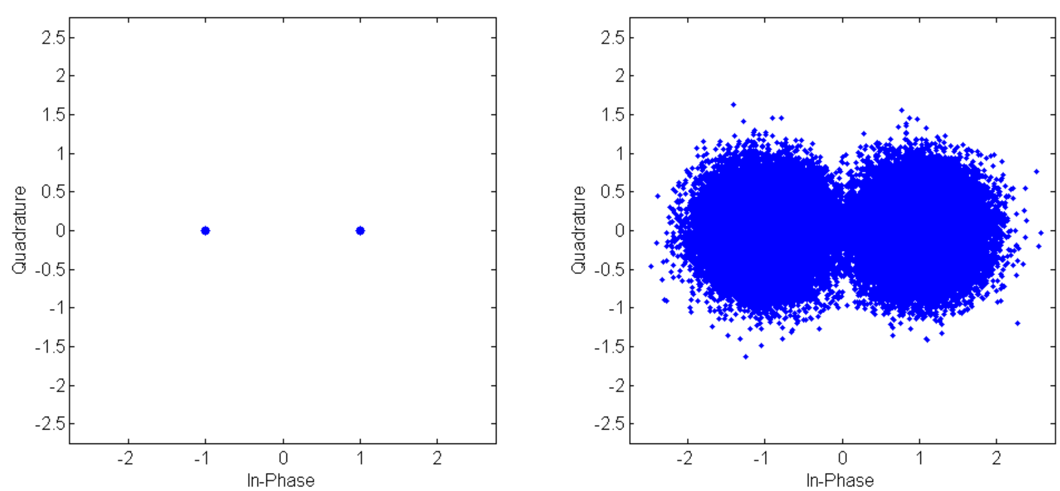

Figure 1.1: Ideal and noisy symbol constellations for BPSK.

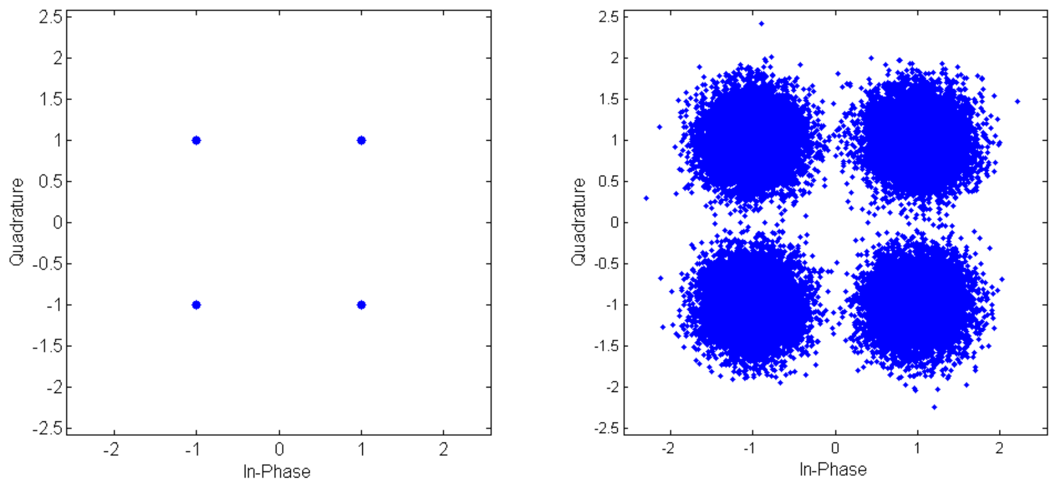

Figure 1.2: Ideal and noisy symbol constellations for QPSK.

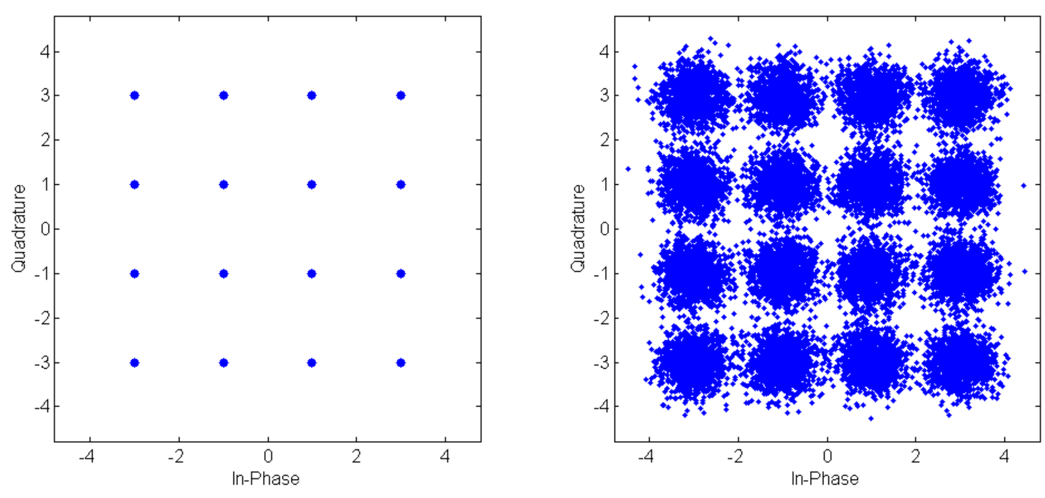

Figure 1.3: Ideal and noisy symbol constellations for 16-QAM.

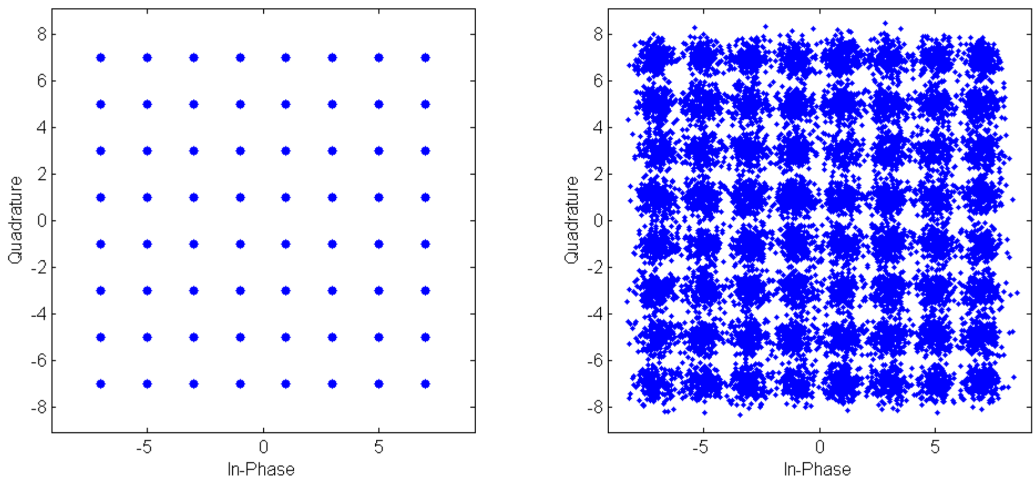

Figure 1.4: Ideal and noisy symbol constellations for 64-QAM.

1.1 - 1.4 show the ideal and noisy constellations for various data rates used in 802.11. Due to channel impairments, such as noise or interference, the received symbol positions deviate from their ideal counterparts. The further the distance from the ideal position, the larger the Error Vector Magnitude (EVM). The receiver must then make an estimate as to which location on the 2D constellation that the received symbol belongs to. As the channel distortion increases, the symbol can be mistakenly believed to belong to the wrong location and a symbol error occurs. As the number of bits per symbol increases, the constellations become more dense and hence, less channel distortion is required to cause a symbol error. This is due to the decreased distance between neighbouring symbols and thus, higher data rates become more susceptible to errors when the channel is poor. Channel quality is typically measured by the signal to noise ratio or SNR. Thus, we can see the relationship between channel quality or SNR and the modulation scheme that can be supported. Figures 1.1 through 1.4 confirm that the maximum tolerable SNR varies between different modulation schemes based on the distance between their symbol constellations. It can clearly be seen that the distances between symbols using BPSK is much larger than when using 64-QAM. Further techniques beyond the modulation scheme diversity, such as error coding and interleaving, can also be used to mitigate poor channel conditions, however, this discussion will be deferred until later chapters.

To illustrate the effect that the modulation scheme has on the end user performance, the theoretical BER for various modulation schemes using equations from [3] as well as the theoretical throughput for these modulation schemes were plotted as a function of the SNR in an AWGN channel. Figures 1.5 and 1.6 show that, as expected, the higher rate PHY modes result in better throughput but have higher SNR requirements. Alternatively, the lower rate PHY modes perform better in the lower SNR range, which clearly indicates the need to choose the correct rate based on the current channel conditions to achieve optimal performance.

The throughput for each data rate is estimated by calculating the number of correctly received bits raised to the power of the number of bits in the packet, i.e., $(1-BER(r))^l \times r$, where r is the data rate and l is the packet length. The IEEE 802.11 standard specifies that data bits should be grouped into packets, which allows multiple stations to share the physical medium. Grouping data bits into data packets can also allow for checksums to be used to detect bit errors and discard those packets that cannot be corrected. Furthermore, specific control information such as the current data rate and packet length can be relayed in a portion of the packet known as the packet header. Figure 1.6 assumes a nominal coding gain and a packet size of 1000 Bytes. As can be seen in the Figure, for each value of the SNR, there is an optimal date rate to transmit at which maximizes the throughput.

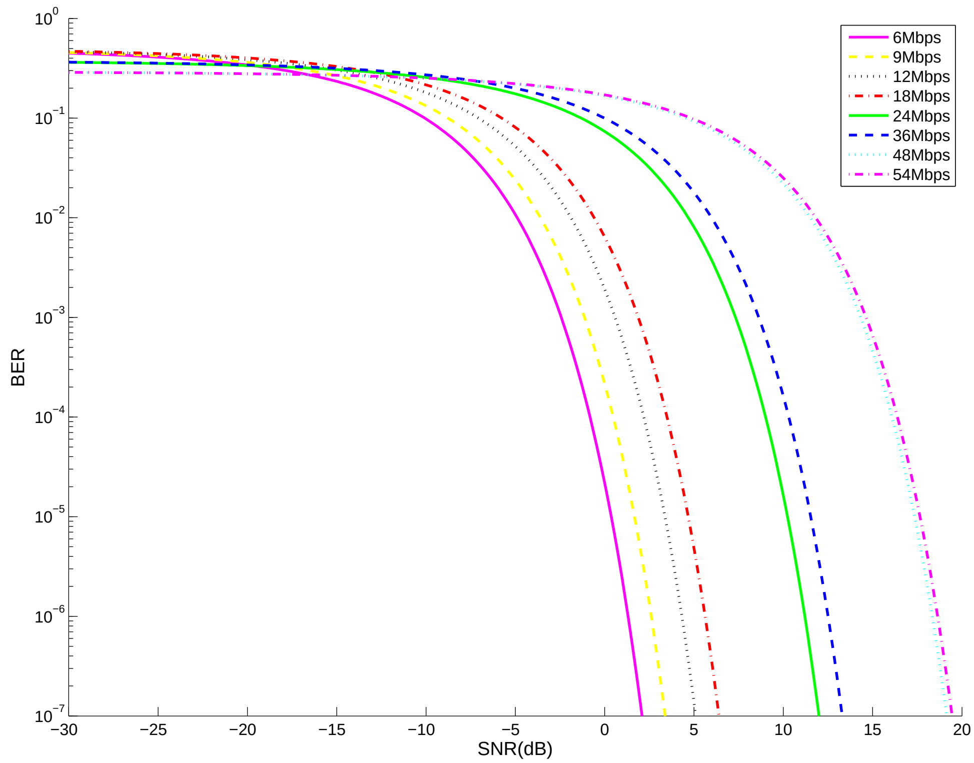

Figure 1.5: Theoretical BER simulation results for selected rates from 802.11.

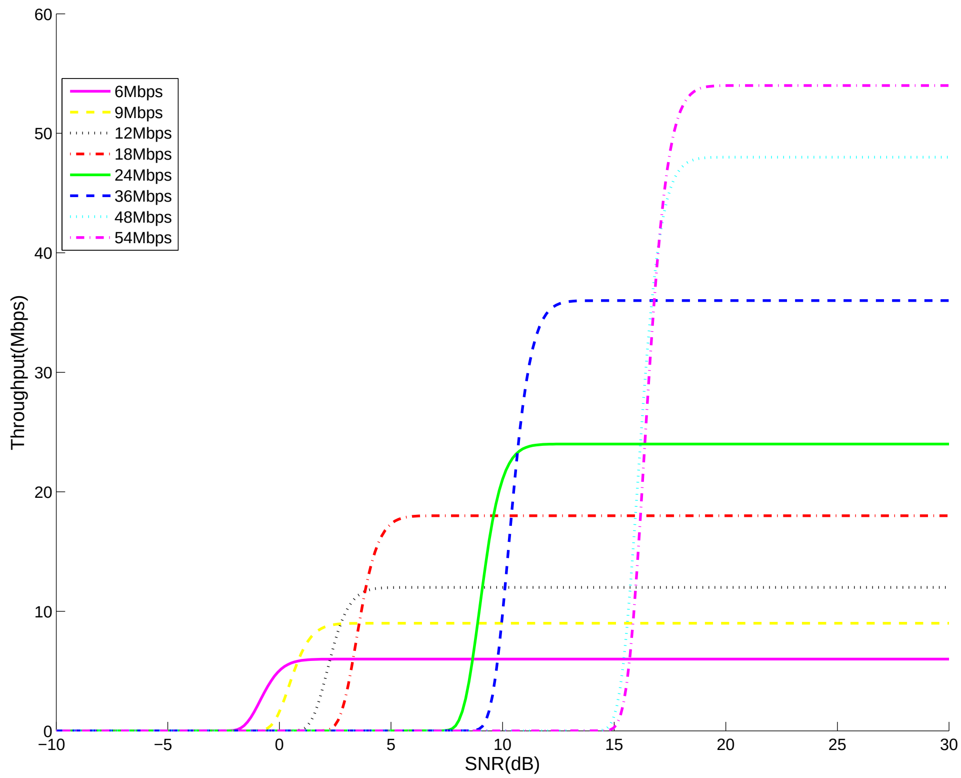

Figure 1.6: Theoretical throughput for selected rates from 802.11.

1.1.2 Situation Awareness

It is the responsibility of the RAA to determine at which moment it is appropriate to switch data rates and which data rate to switch to. If the current data rate begins to experience a large number of bit errors, or the transmission time becomes too long due to retransmissions, it may indicate that the RAA should step the data rate down to a lower data rate. Alternatively, if the number of bit errors in a packet for a particular data rate is very low, and the transmission time is short, the RAA may determine that it is time to try a higher data rate. The design and implementation of such a RAA is the main focus of this thesis. To illustrate how important it is for the RAA to select the correct data rate at the correct moment, a trace of the RSSI is shown in Figure 1.7. In this figure measurements were carried out and the RSSI was recorded over a period of time. Figure 1.7 shows that the signal strength can vary with time due to reflections and movement of objects and people. An RAA that does not adapt the data rate to the changes occurring in the environment fails to take advantage of good conditions when they arise and fails to avoid an accumulation of bit errors as a result of not reducing the data rate in poor conditions. The RAA must be aware of it's environment and situation in order to fully utilize the wireless spectrum. The RAA can determine the current environmental conditions and act aggressively in good channel conditions or act conservatively in poor channel conditions. In either case, the RAA must be situation aware in order to properly adapt the data transmission rate and as a result improve performance in the form of increased throughput.

1.1.3 Power Adaptation

In a wireless network, where nodes most likely operate on limited battery power, it is also important to conserve energy in order to maximize the lifetime of the battery.

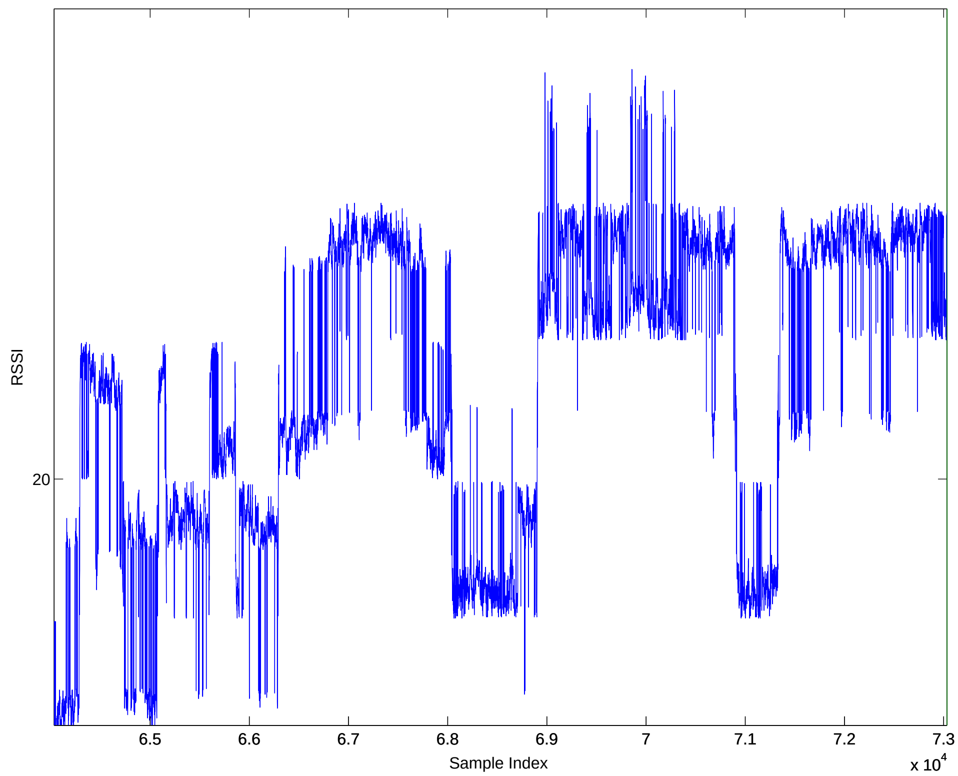

Figure 1.7: Trace of the RSSI over time showing the large swings that can occur as a result of the environment.

Furthermore, the conservation of energy is an important goal in today's society in order to attain a sustainable energy efficient future. One way to reduce the amount of energy consumed by wireless communication systems is to reduce the transmission power. In this way, mobile stations (MSs) (e.g., laptops, cell phones, PDAs, etc.) can conserve battery life and base stations (BSs) can reduce energy consumption. In addition to conserving energy, reducing the transmission power can also serve to reduce the amount of interference caused by wireless transmitters.

Reduction of the transmit power to the lowest possible value can help to increase battery life for MSs and reduce power usage for BSs, however, it is important to maintain connectivity and throughput. If the transmit power is reduced below a certain level, then connectivity will be lost and or the throughput reduced. Therefore, it is important to reduce the transmission power while, at the same time, maintaining connectivity [4] and throughput for a given quality of service (QoS). Alternatively, transmitting at the maximum transmission power leads to an unnecessary waste of energy if the link quality is high enough to support a lower transmission power while maintaining QoS requirements, and can also lead to an increase in interference among neighbouring WLANs. In addition, the optimal transmit power is the minimum transmit power required to maintain connectivity and that maximum power savings can be achieved by minimizing the transmission power on a link-by-link basis [5].

As an example, consider the following scenario as shown in Figure 1.8 : STA 1 is close to STA 2 and STA 2 is close to STA 3 but STA 1 and STA 3 are not close to each other. Therefore, STA 1 does not require the maximum transmit power in order to communicate with STA 2. Similarly, STA 2 and STA 3 can also communicate using a lower transmit power. If STA 1 communicates with STA 2 using its highest transmit power and STA 2 communicates with STA 3 using its highest transmit power, then those two stations would be wasting energy needlessly. Alternatively, STA 1 may need

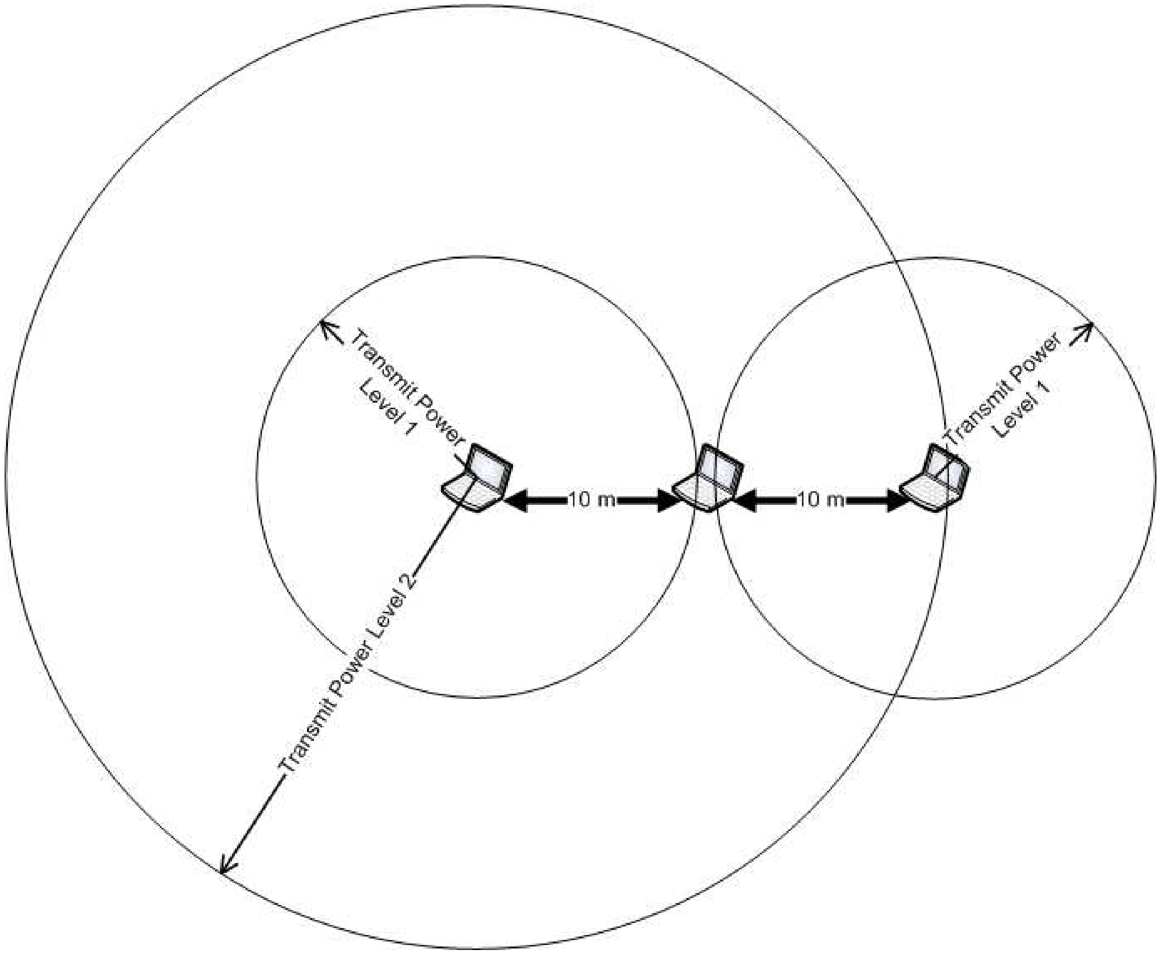

Figure 1.8: Transmission power scenario.

higher transmit power in order to communicate with STA3, in which case the energy is not wasted since that transmit power is required to maintain connectivity and or QoS constraints. Thus, we can see that a determination of the minimum transmission power is needed to prevent unnecessary energy consumption.

In addition to the development of a RAA, a simple, yet effective, algorithm for determining the minimum transmission power based on a binary search algorithm is developed and evaluated. The algorithm can determine the minimum transmission power in an IEEE 802.11 WLAN system by varying the transmission power and waiting for a MAC layer ACK. By using a binary search algorithm, an accurate estimate of the minimum transmission power required for successful transmission can quickly be obtained. The method is tested using IEEE 802.11 hardware based on an Atheros chipset.

1.2 Thesis Contribution and Organization

The main contribution of this thesis is the design, implementation and testing of a RAA for the IEEE 802.11 WLAN. The RAA is implemented on an AP running the FreeBSD OS using an Atheros 5212 chipset. The RAA is located in the Atheros device driver and is written in the C programming language.

Specifically, the RAA determines the current environmental conditions or situation during a calibration phase which allows the algorithm to obtain situational awareness. The algorithm then uses a combination of the RSSI and PER to determine which rate to transmit at. The algorithm is highly adaptive and is able to quickly react to changes in the wireless channel, but also provides long term, stable rate selection, thus, reducing unnecessary fluctuations in rate selection.

The algorithm is termed Situation Aware Rate Adaptation (SARA) and is proposed with the objective of throughput maximization for IEEE 802.11 WLANs, under time-varying wireless channels, by using a real-time updated RSSI-Rate look-up table. SARA determines the current environmental conditions by first collecting RSSI and PER data and then uses an adaptive table aligned to the current channel conditions to select an appropriate data rate. The algorithm begins by first discovering its environment then, by using the average RSSI, a set of feasible data rates from the calibrated table is chosen. Finally, the algorithm chooses from the available feasible data rates by using the PER. The algorithm also monitors current channel conditions and if a large change in environmental factors is detected, the algorithm quickly recalibrates and aligns the adaptive table to the new channel conditions. In this way, SARA can adapt to quickly changing channel conditions, but avoid unnecessary data rate changes. In addition, SARA avoids the use of probing packets and RTS/CTS handshaking, which reduces overhead and improves adaptability to rapid changes.

SARA attempts to provide rapid data rate adaptation that maximizes throughput. It provides stable data rate selection while being perceptive to environmental changes. The theoretical basis for the design of SARA and its implementation are provided in this thesis. In order to evaluate the performance of SARA, the theoretical analysis of the throughput and the measurement results are also presented. It can be found from the measurement results that SARA can significantly improve the throughput when compared to SampleRate in the mobile environment, and achieve as good or slightly better throughput in static channel conditions.

The following points outline the main contributions contained within:

- A new algorithm design which employs the use of adaptive RSSI rate selection based on PER curves aligned to current channel conditions. The use of RSSI rate selection ensures rapid adaptability while also being aware of the current situation by using the current PER.

- Theoretical proof that the RSSI can be adapted by using shifted PER curves dependent on the operating environment.

- Laboratory simulations which support the claim that the PER curves for each data rate can be linearly shifted within the normal operating range of an OFDM-based IEEE 802.11 system.

- The algorithm is implemented in an Atheros device driver for the FreeBSD operating system. The device driver code is written in the C programming language and adheres to the standards defined for programming the FreeBSD kernel.

- The algorithm is tested in moderately severe environments typical to normal OFDM-based IEEE 802.11 systems usage such as in libraries and in other public spaces where the use of WiFi is popular.

- Measurement results show a significant performance improvement over the benchmark algorithm SampleRate and we see an observed overall improvement of approximately 20% in the measurements performed.

- In addition, a technique to reduce the transmission power of an 802.11 device is developed and evaluated. The method varies the transmission power using a binary search technique to quickly achieve a minimum power level for transmission.

The remainder of this thesis is organized as follows:

Chapter 2 provides a brief description of general wireless communication systems and detailed background information on the 802.11 standard. The PHY and MAC layer of the 802.11 standard are described with particular attention paid to the DCF algorithm used by the MAC layer of the standard.

Chapter 3 discusses existing RAAs, comparing and contrasting their strengths and weaknesses. Currently, there are three RAAs implemented in the Atheros device driver, however, more recent RAAs have been proposed which will also be discussed. RAAs use various methods to determine the data transmission rate such as the SNR, PER and RTT. In particular, we will examine three RAAs implemented in the Atheros device driver such as, SampleRate, AMRR and ONOE as well as some popular RAAs found in the literature, such as SmartRate, SmartSender, CHARM, RBAR, RRAA, OAR and RSSLA.

Chapter 4 describes the design of the RAA. The measurements performed to characterize the wireless channel, which were used in the design of the RAA, will be presented and discussed. Additionally, the design limitations faced and assumptions made will be presented. In this chapter the implementation details of the RAA will also be presented. In particular, the specific data structures, algorithms and limitations imposed by the FreeBSD OS during the implementation of the RAA are presented.

Chapter 5 provides a thorough examination of the experimental results of the RAA. The throughput achieved by the RAA is compared to the throughput of the benchmark algorithm SampleRate implemented in the Atheros device driver. Tests are carried out in various typical real world scenarios including static, dynamic and congestion scenarios. The benefits and drawbacks of SARA are also discussed.

Chapter 6 will discuss the minimum transmission power determination technique for IEEE 802.11 WLANs. In this chapter the technique and its evaluation is presented.

Chapter 7 concludes the thesis with a summary of the work accomplished and a discussion of possible future work.

Chapter 2

Background

2.1 Background on IEEE 802.11

In this section, a brief background on the IEEE 802.11 standard is given which includes a review of OFDM technology, the different 802.11 PHY layers and the 802.11 MAC layer. OFDM is used as the PHY layer in IEEE 802.11a/g/n/ac and is becoming popular as a PHY layer in many other standards such as WiMAX and LTE.

2.1.1 Review of OFDM

OFDM consists of multiple subcarriers with each subcarrier being modulated by a traditional modulation scheme such as QAM or PSK. Each subcarrier is orthogonal to all other subcarriers, thus eliminating the need for guard bands between subcarriers. By maintaining orthogonality, a high bandwidth efficiency can be accomplished [6]. In traditional modulation schemes, as the data rate increases, the symbol duration decreases which causes the system to suffer from more severe ISI caused by the dispersive fading of a wireless channel. However, OFDM divides the entire frequency selective fading channel into many narrow band flat fading subcarriers in which high rate data are transmitted in parallel thus reducing the ISI due to a longer symbol duration. Figure 2.1 shows an OFDM signal with 16 subcarriers.

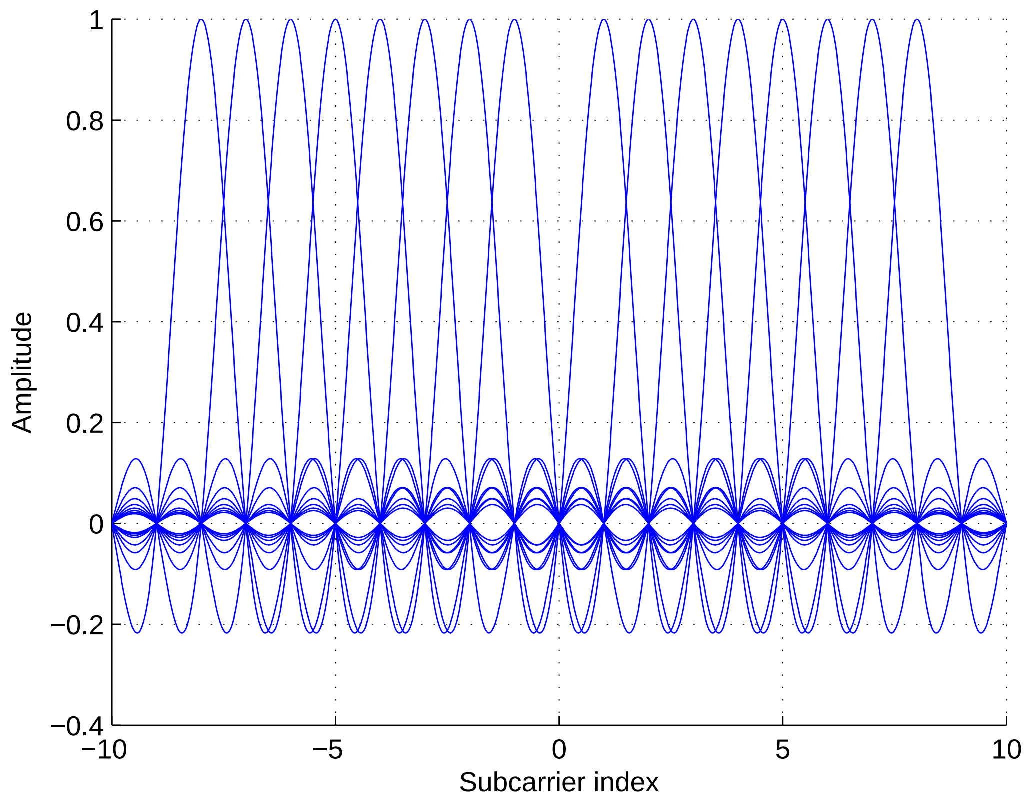

Figure 2.1: Frequency-domain representation of an OFDM signal consisting of 16 subcarriers.

2.1.1.1 Mathematical Description of OFDM

Let $\{s_k\}_{k=0}^{N-1}$ be the complex symbols to be transmitted by OFDM, then the OFDM signal can be written as

$$s(t) = \sum_{k=0}^{N-1} s_k e^{j2\pi f_k t} = \sum_{k=0}^{N-1} s_k \phi_k(t), \text{ for } 0 \le t \le T_s \tag{2.1}$$

In most practical systems OFDM modulation can be accomplished through the use of the FFT algorithm. If $s(t)$ is sampled at an interval of $T_{sa} = \frac{T_s}{N}$, then

$$S_n = s\left(n\Delta_s\right) = \sum_{k=0}^{N-1} s_k e^{j2\pi f_k \frac{nT_s}{N}} \tag{2.2}$$

Since $f_k T_s = k$, then we can rewrite Eq. 2.2 as

$$S_n = \sum_{k=0}^{N-1} s_k e^{j\frac{2\pi kn}{N}} = \mathrm{IDFT}\{s_k\} \tag{2.3}$$

Therefore, in an OFDM system, the IDFT can be used to implement the transmitter and alternatively, the DFT can be used to implement the receiver. The FFT algorithm can be used as an efficient implementation of the DFT and due to it's simplicity and low cost has become a very popular component in OFDM transmitters and receivers. The following figures provide an idealized diagram of an OFDM transmitter and receiver using an FFT block for modulation.

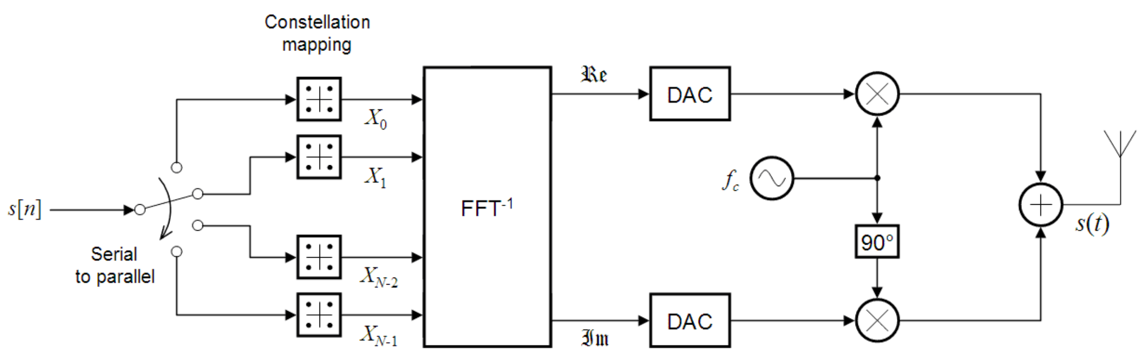

Figure 2.2: An ideal OFDM transmitter [1].

The OFDM symbol is typically extended in a circular fashion in order to deal with the delay spread of wireless channels [6] as well as providing the circular extension to the signal which allows for a circular convolution to be used in the DFT/FFT processing. This extension is known as the CP. Unfortunately, despite it's many benefits, OFDM systems do suffer from some drawbacks such as a high PAPR, frequency offset sensitivity and loss of efficiency due to the use of the CP. The PAPR is due to

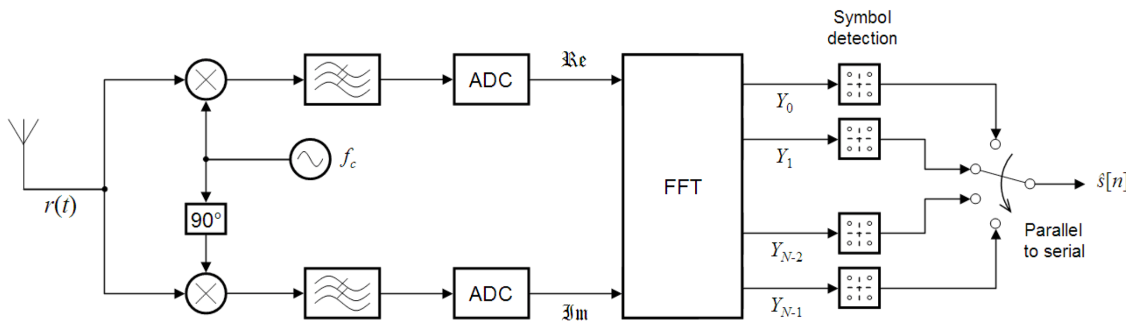

Figure 2.3: An ideal OFDM receiver [1].

the widely varying amplitude levels found among different subcarriers. This forces the power amplifier to operate in a much broader linear range, reducing power efficiency. In addition, the sensitivity to frequency offset is due to the tight spacing between subcarriers. The loss of orthogonality between subcarriers can introduce many bit errors, thus much effort must be spent reducing the effects of frequency offset.

2.1.2 IEEE 802.11 PHY Layer

This section describes the various PHY layers that have been adopted over time with a specific emphasis on the OFDM PHY layers found in the IEEE 802.11a/g specifications. The 802.11 specification defines multiple PHY layers as was previously described in Chapter 1. The PHY layers range from FHSS systems, IR systems, DSSS systems and OFDM systems. While the work of this thesis is applicable to any multi-rate system, we will concentrate primarily on the OFDM PHYs.

2.1.2.1 IEEE 802.11b DSSS PHY

This section describes the DSSS PHY that is known as 802.11b. The original DSSS PHY consisted of two data rates of 1 Mb/s and 2 Mb/s and was later extended to include 5.5 Mb/s and 11 Mb/s data rates, the latter which is known as the high rate

(HR) extension to the DSSS PHY. The DSSS PHY operates in the 2.4 GHz range with a 22 MHz bandwidth and spreads the information signal using an 11-chip Barker sequence. The following PN code sequence is used to spread the data:

+1, -1, +1, +1, -1, +1, +1, +1, -1, -1, -1

There are two modulation formats specified for the DSSS PHY, DBPSK for the 1 Mb/s rate and DQPSK for the 2 Mb/s rate. Table 2.1 and 2.2 show the encoding schemes for DBPSK and DQPSK respectively.

Bit Input	Phase Change
0	0
1	π

Table 2.1: DBPSK encoding format used for the 1 Mb/s data rate of the DSSS PHY.

Dibit Input	Phase Change
00	0
01	$\pi/2$
11	π
10	$3\pi/2$

Table 2.2: DQPSK encoding format used for the 2 Mb/s data rate of the DSSS PHY.

As mentioned previously, there is a high rate (HR) extension to the DSSS PHY which provides rates of 5.5 Mb/s and 11 Mb/s. To provide these higher data rates, a different modulation technique, known as CCK, is employed. CCK uses an 8-chip sequence, but maintains an 11 Mchip/s chipping rate in order to remain interoperable with the original DSSS PHY.

In the 5.5 Mb/s transmission scheme, four data bits are modulated onto the eight chips and in the 11 Mb/s transmission scheme eight bits are modulated onto the eight chips. The eight chips of the symbol are determined by the following equation:

$$\vec{\mathbf{c}} = (c_0, ..., c_7) = \Big(e^{j(\phi_1+\phi_2+\phi_3+\phi_4)}, e^{j(\phi_1+\phi_3+\phi_4)}, e^{j(\phi_1+\phi_2+\phi_4)}, -e^{j(\phi_1+\phi_4)}, e^{j(\phi_1+\phi_2+\phi_3)}, e^{j(\phi_1+\phi_3)}, -e^{j(\phi_1+\phi_2)}, e^{j\phi_1}\Big) \tag{2.4}$$

where $\phi_1, ..., \phi_4$ are the phase changes determined from Tables 2.1 and 2.2. This CCK modulation is a form of generalized Hadamard transform encoding.

DSSS was chosen as a PHY due to it's resistance against narrowband interference, enhanced security due to spreading and increased resistance to multipath fading present in indoor environments [7]. Furthermore, CCK was chosen to extend the original DSSS PHY since this form of modulation allows more data bits to be transmitted without sacrificing resistance to interference and multipath fading. The CCK form of modulation uses code words which possess excellent autocorrelation properties, that is, they have a low cross correlation and have a large correlation at zero offset [8]. It is this property that makes CCK modulation a good choice for the 802.11b standard. Despite the benefits, a DSSS system is limited by the amount of information it can carry in a given bandwidth. In order to transmit more data a DSSS system has to increase the chipping rate which increases the amount of bandwidth required and requires more expensive high frequency hardware. This drawback requires a new technique to increase data rates and hence a new PHY is needed for high speed 802.11 communications.

2.1.2.2 IEEE 802.11a OFDM PHY

The IEEE 802.11a PHY is an OFDM based communication system that allows data rates of up to 54 Mb/s, which is a significant improvement over the 11 Mb/s maximum rate available in 802.11b. The system uses 52 subcarriers modulated by BPSK, QPSK, 16-QAM and 64-QAM with coding rates of 1/2, 2/3 or 3/4. The data bits are scrambled, then passed through a convolutional encoder with a coding rate of 1/2. Further coding rates are obtained by puncturing the mother coding rate of 1/2 by deleting coded bits from the output of the encoder. The coded bits are then interleaved to reduce burst errors, then finally the interleaved bits are grouped into symbols modulated by one of the modulation schemes mentioned previously. Table 2.3 presents the data rates, modulation schemes, coding rates and bits per subcarrier available in the 802.11a PHY.

Data Rate (Mbits/s)	Modulation Scheme	Coding Rate	Coded Bits per Subcarrier	Coded Bits per OFDM Symbol	Data Bits per OFDM Symbol
6	BPSK	1/2	1	48	24
9	BPSK	3/4	1	48	36
12	QPSK	1/2	2	96	48
18	QPSK	3/4	2	96	72
24	16-QAM	1/2	4	192	96
36	16-QAM	3/4	4	192	144
48	64-QAM	2/3	6	288	192
54	64-QAM	3/4	6	288	216

Table 2.3: Data rates, modulation schemes, coding rates, coded bits per subcarrier and OFDM symbol as well as data bits per OFDM symbol available in the 802.11a PHY.

The 802.11a PHY uses 48 data subcarriers with four pilot subcarriers used. The pilot subcarriers are used to compensate for frequency offsets and phase noise. The pilot subcarriers transmit known pseudo binary sequences modulated using BPSK. The bandwidth of the 802.11a system is 16.66 MHz with a channel spacing of 20 MHz. This results in a subcarrier spacing of 312.5 kHz which in turn gives an FFT period of 3.2 μs. The cyclic prefix or guard length is 0.8 μs giving a total symbol period of 4 μs. The subcarriers are divided using a 64 point FFT. Figure 2.4 show

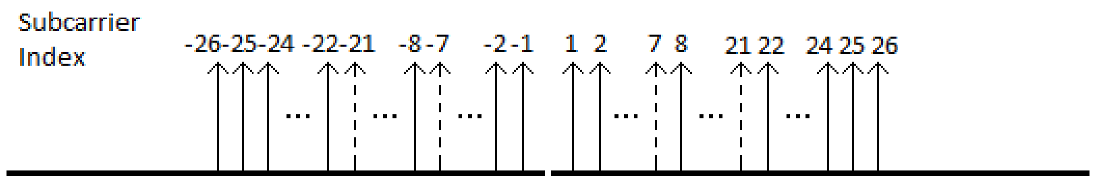

Figure 2.4: 802.11a OFDM subcarrier allocation.

the subcarrier arrangement in the 20 MHz 802.11a channel, respectively while Table 2.4 shows a summary of the 802.11a OFDM PHY characteristics. In Figure 2.4, the dashed arrows indicate the pilot subcarriers.

Characteristic	**Value**	**Comments**
tSlotTime	9 μs	Slot time
tSIFSTime	16 μs	SIFS time
tDIFSTime	34 μs	DIFS = SIFS + 2 Slot
aCWmin	15	Min contention window
aCWmax	1023	Max contention window
tPLCPPreamble	16 μs	PLCP preamble duration
tPLCP_SIG	4 μs	PLCP signal field duration
tSymbol	4 μs	OFDM symbol interval

Table 2.4: IEEE 802.11a OFDM PHY characteristics.

In addition to specifying the OFDM parameters, the 802.11a PHY also specifies PHY layer specific framing parameters to facilitate communication between unique STAs and between STAs and APs. The PHY layer adds a PLCP preamble and a PLCP header to the MPDU MAC payload to create a PPDU. The preamble consists of two sections, the short training sequence and the long training sequence which are used for synchronization. The short training sequence consists of ten short symbols without guard intervals using only 12 subcarriers, with the first seven symbols used for signal detection, AGC and antenna diversity selection, while the final three symbols

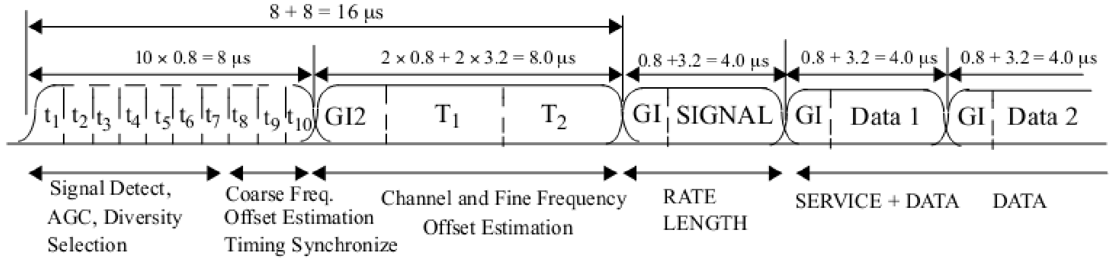

Figure 2.5: 802.11a OFDM preamble and header structure [2].

are used for coarse frequency offset estimation and timing synchronization. The long training sequence contains a guard interval and consists of two long symbols which uses 53 subcarriers including the 0th (DC) subcarrier and is used for channel estimation and fine frequency offset estimation. The PLCP header section of the 802.11a PHY is known as the SIGNAL field and consists of the Rate field, the Length field, the reserved bit and the parity bit. The Rate field is a four bit field that indicates the MCS used in the following data symbols, while the Length field indicates the length in bytes of the following MAC frame. The header is transmitted using BPSK with 1/2 coding rate and corresponds to the 6 Mb/s data rate. This data rate is chosen as a common data rate so that all STAs may decode this portion of the frame and is also chosen since it is the most robust data rate, reducing the possibility of error in decoding the PLCP preamble and header. The Rate and Length fields are required for decoding the Data portion of the PPDU by the recieving STA since it must know which modulation scheme was used or Rate and how long the Data section is as indicated by the Length field. Figure 2.5 presents the 802.11a preamble and header with t_1 to t_{10} denoting the short training symbols and T_1 and T_2 denoting the long training symbols. Figure 2.6 also shows the PPDU frame format in detail.

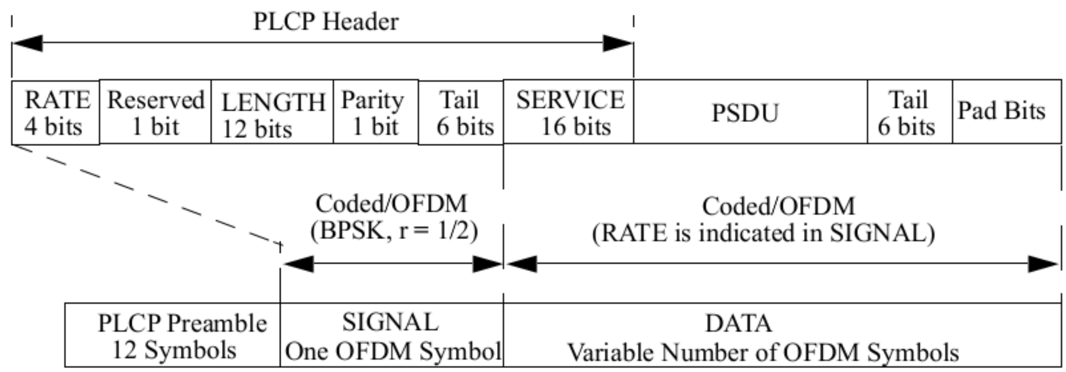

Figure 2.6: 802.11a OFDM PPDU frame format [2].

2.1.2.3 IEEE 802.11g PHY

The 802.11b and 802.11a PHYs are non-interoperable and thus a device manufactured for one PHY cannot operate with a device manufactured for another. The solution is to combine the two PHYs into a single interoperable PHY which can support the legacy 802.11b PHY but provide data rates that are at the same level as the 802.11a PHY. 802.11g was standardized to fill this need and supports both the 802.11b data rates as well as the 802.11a data rates but remaining in the 2.4 GHz range.

Supporting the legacy 802.11b data rates and maintaining interoperability requires that the OFDM portion of the transmit signal must be preceded at all times by an 802.11b preamble and header. This preamble and header is transmitted using the same 1 Mb/s MCS used in 802.11b, that is, the DSSS PHY using the Barker sequence described in Section 2.1.2.1. The inclusion of the 802.11b legacy preamble means that the transmission of a frame takes longer and as such, the throughput is reduced. In addition, energy is consumed transmitting the legacy preamble and thus, the support of the legacy 802.11b PHY can reduce throughput and can also increase energy consumption. Fortunately, most manufacturers include an option to

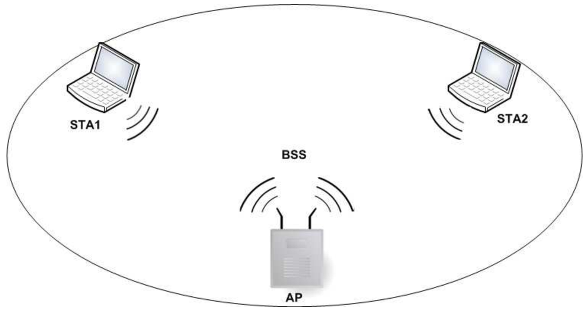

Figure 2.7: Infrastructure mode WLAN, known as a BSS.

cease transmission of the legacy preamble if it is known that no 802.11b devices are present in the WLAN.

2.1.3 IEEE 802.11 MAC Layer

The MAC layer is the controlling layer for any PHY layer and controls the transmission of data using the PHY. The MAC layer in 802.11 uses a technique known as CSMA/CA and supports two communication architectures, the BSS and IBSS. The BSS is also known as infrastructure mode and contains at a minimum an AP and a STA with all communication going through the AP. The AP can connect to a wired network, known as a DS, to form an ESS. The IBSS is also known as ad-hoc mode and lacks an AP which only requires two STAs in order to exist. IBSSs are typically short-lived and unplanned and communication is achieved through direct STA to STA data transfer. Figures 2.7 and 2.8 present the typical configurations found in BSS and IBSS modes respectively. When two or more BSSs are connected through a DS, the result is an ESS. Figure 2.9 illustrates an ESS.

The 802.11 MAC supports three different medium access methods known as coordination functions, the PCF, HCF and DCF. The PCF provides contention free access to the medium and the HCF provides access to the medium for QoS STAs.

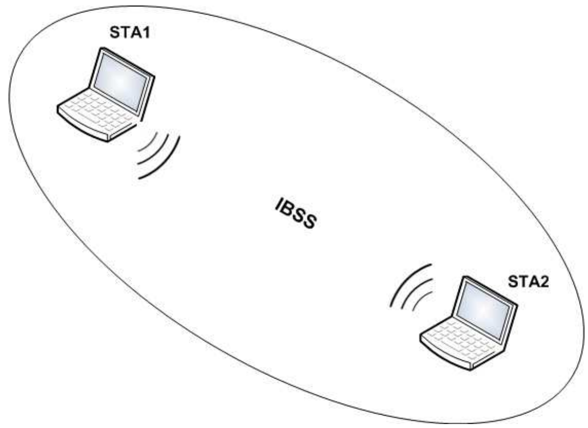

Figure 2.8: Ad-hoc mode WLAN, known as an IBSS.

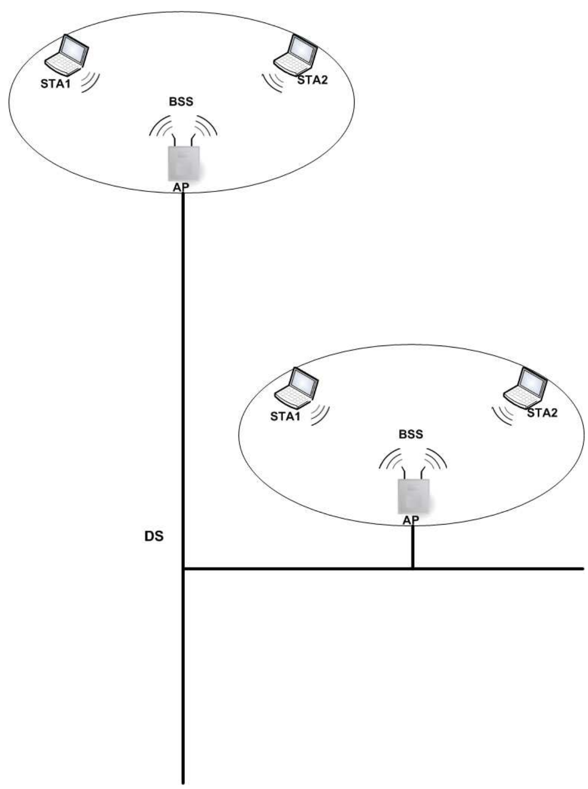

Figure 2.9: Illustration of an ESS, with multiple BSSs connected through a DS.

The DCF is the most widely used coordination function and also forms the basis for both the PCF and HCF hence, only the DCF will be discussed in this manuscript.

2.1.3.1 DCF Operation in 802.11

Due to the inherent instability in the wireless channel, the DCF requires that data transmission follow an ARQ procedure. That is, each transmission should be followed by an ACK packet if successful and a transmission failure is assumed if no ACK packet is received. Furthermore, due to the half-duplex nature of 802.11 devices, collision detection is impossible and a system of collision avoidance is implemented to mitigate inevitable collisions.

The collision avoidance mechanism is designed to reduce the probability of collisions amongst individual STAs who wish to use the medium at the same time while at the same time maintaining fair access to the medium for all STAs. Both physical and virtual carrier sensing is used to determine the state of the medium. Physical carrier sensing uses energy detection to determine whether the medium is busy or not and requires listening to the channel before attempting to transmit data. Virtual carrier sensing uses a RTS/CTS exchange before attempting to access the medium and incurs additional overhead. Since the RTS/CTS is rarely used in typical WLANs, the details of this access method will be omitted from this manuscript.

The typical usage of the DCF is as follows. Before transmitting a packet, the transmitter waits for the channel to be idle for at least a DIFS, if the transmitter experiences a successful packet reception, or an EIFS, if the transmitter experiences a failed packet reception. After a DIFS/EIFS, the transmitter starts a backoff timer by selecting a random number of backoff time slots and setting this as the value of the backoff timer. This random number is uniformly selected from the range [0, CW], where CW is the current value of the contention window. During the decrementing of the backoff timer, if the channel becomes busy, the backoff timer is suspended until such time as the medium becomes free again. The backoff timer resumes once the channel is idle again for at least a DIFS/EIFS. Once the backoff timer times out, the transmitter proceeds to transmit the packet. If the receiver receives the packet successfully, it waits for a SIFS and responds by sending an ACK packet to the transmitter without listening to the channel status. If the packet is received with errors, the receiver discards the packet and no ACK packet is sent to the transmitter. If the received packet is a duplicate packet and is received without errors, the receiver discards the packet and sends an ACK packet to the transmitter. Upon the arrival of an ACK packet at the transmitter, the transmitter starts another carrier sensing (CS) and backoff cycle for transmitting subsequent packets, if necessary. At this time, the contention window (CW) is then reset to CWmin. At the transmitter, after a SIFS plus propagation delay, known as an ACK timeout, if the transmitter does not receive a valid ACK packet, it increases the CW to the next value in the CW table, unless the current CW value is CWmax. Then, the transmitter starts the CS and backoff procedures for a retransmission, if the retransmission counter for the current packet has not reached its maximum allowable value otherwise, the packet is finally discarded. The value of the slot time varies with different PHYs. In 802.11a the value for a slot time is 9 μs and for 802.11b, the value for a slot time is 20 μs with 802.11g providing both values for the slot time. The exponential backoff value for the CW is shown in Figure 2.10, as can be seen, the value for the CW is doubled each time until it reaches a value of CWmax. The basic access mechanism as described is graphically depicted in Figure 2.11.

Based on Figures 2.10 and 2.11 we can now see how multiple stations vying for the medium might interact. Figure 2.12 illustrates the backoff procedure and access mechanism amongst multiple STAs.

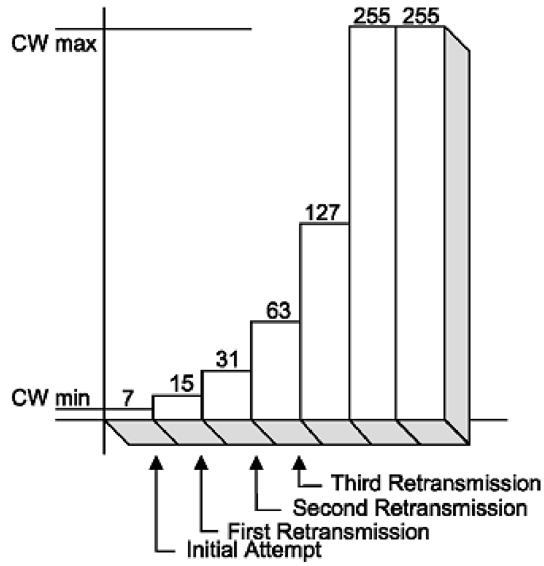

Figure 2.10: Possible values for the CW used in the backoff procedure, based on an exponential increase [2].

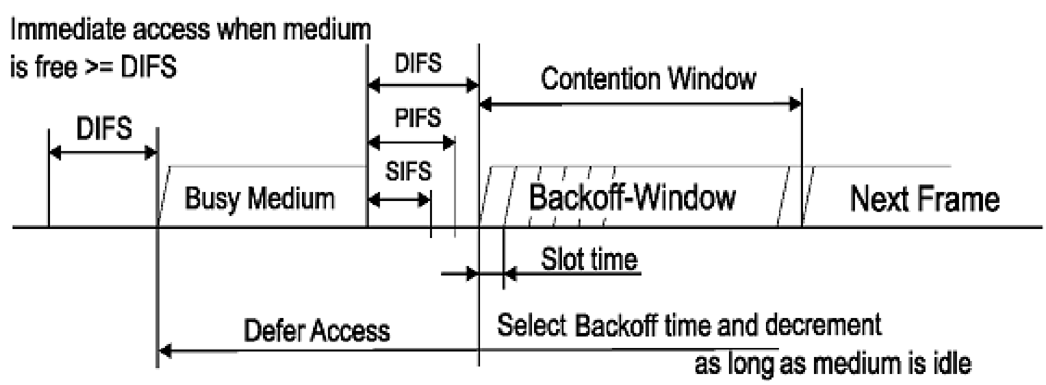

Figure 2.11: Graphical depiction of the basic access mechanism in the DCF [2].

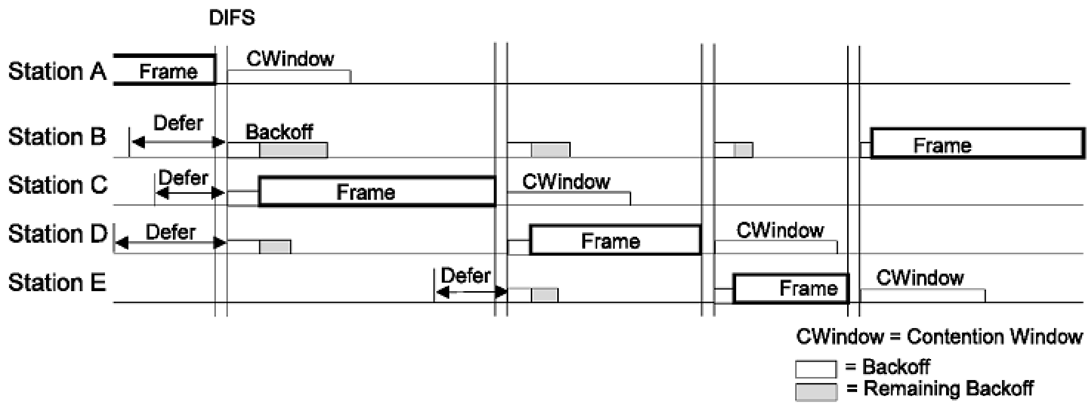

Figure 2.12: Multiple stations contending for the medium using the backoff procedure in the DCF [2].

The interframe spacings are an important component of the DCF in order to provide fairness and priority to particular transmissions. The SIFS is the shortest interframe space and allows successive frames to be separated by an amount of time that does not require sensing the medium first. For example, the SIFS separates a transmitted data frame from it's associated ACK frame. The DIFS is the typical interframe spacing required in the DCF. A frame sequence from any STA participating in a DCF MAC must begin it's transmission at least a DIFS after any previous transmission. Finally, the EIFS is the time required for a STA to wait following a failed transmission before it may retransmit that frame. Setting the values of the various interframe spacings allows the MAC to prioritize frames, that is, SIFS $<$ DIFS $<$ EIFS. Thus, a STA waiting for a SIFS has a higher priority than a STA waiting a DIFS with the least priority given to a STA waiting a EIFS.

In order to facilitate communication between STAs, a common MAC frame header is required and is added to every data packet which forms the frame body. Each MPDU consists of a MAC header, comprised of the frame control, duration, address and sequence control information. The MAC header is followed by the frame

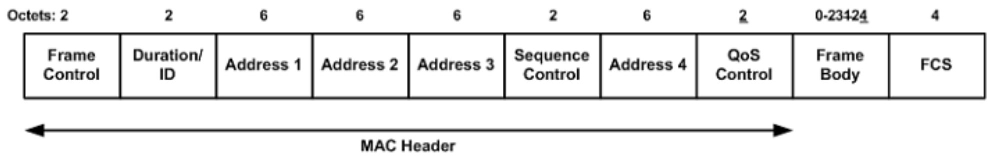

Figure 2.13: General format of the 802.11 MPDU frame [2].

body and finally by the FCS. The FCS is a 32-bit CRC used to determine if the frame was received in error or not and is calculated over the MAC header and frame body. Figure 2.13 shows the general frame format for the 802.11 MAC.

2.2 Background on Rate Adaptation

In this section, a brief background on rate adaptation is given which discusses why rate adaptation is needed and what rate adaptation is.

IEEE 802.11 WLANs suffer from a wide variety of impairments including, but not limited to, interference from devices in the same spectrum, fading and attenuation. Additional impairments also come from the movement of people and objects causing changes in carrier frequency known as Doppler shifting. In any case, the severity of impairments changes with time and rate adaptation provides a means to adapt the data rate to the current channel conditions. As discussed in Chapter 1, when channel conditions are good, the rate adaptation algorithm selects a high data rate. Alternatively, when channel conditions are poor, the rate adaptation algorithm selects a low data rate.

RAAs can be grouped into three broad categories and within each category there are two subcategories. The RAA can be either transmitter or reciever based algorithms, packet statistics or SNR based algorithms, or finally, window or frame based algorithms [9]. Transmitter based RAAs determine the data rate to transmit at at the transmitter side, while receiver based RAAs determine the data rate to transmit at at the receiver side. Obviously, receiver based RAAs must provide a means of relaying the chosen data rate back to the transmitter for the upcoming data transmission. Packet statistics based RAAs monitor success and loss statistics of transmission attempts to infer channel conditions and choose an appropriate data rate to transmit at. On the other hand, SNR based RAAs measure or estimate the SNR of the channel to determine which data rate to transmit at. Packet statistics information is either directly supplied by the device driver or implicitly determined through the number of ACKs received, whereas the SNR at the receiver is not easily obtained. Finally, RAAs that use a window to calculate statistics or success information employ some form of averaging or smoothing to determine the data rate to transmit at. Alternatively, frame based RAAs do not average the collected information and decisions are made per-frame or per-packet.

RAAs fall under the scope of RRM along with TPC amongst others and is an important aspect of any wireless communication system. The only alternative to using a RAA is to use a fixed data rate. Using a fixed data rate means that any changes in the channel conditions that are severe enough to cause a high number of bit errors can render the transmission of data impossible at the current data rate. Furthermore, using a fixed data rate also eliminates the performance gains associated with a higher data rate which can be chosen during favourable channel conditions. In order for a fixed rate to outperform a RAA, the channel conditions must remain relatively stable for a lengthy period of time. This situation is rare even in stationary conditions, but even more so given the ubiquity of mobile devices using 802.11 radios.

2.3 Background on Transmit Power Control

In this section a brief overview of TPC is given including a background on the binary search algorithm used in determining the minimum transmit power in 802.11.

2.3.1 Transmit Power Control

TPC determines the optimal transmit power level to use in transmit mode [10] with the goal of minimizing the amount of power or energy used. While minimizing power consumption can increase battery life, a trade-off exists between the reduction of power and the maintenance of connectivity [11] and it is important that connectivity be maintained. Often, TPC is combined with rate adaptation to determine an energy efficient combination which maintains QoS while reducing power consumption as found in the paper titled Adaptive Transmit Power Control in IEEE 802.11a Wireless LANs [10]. In order to reduce the 'hidden terminal' problem, the RTS/CTS method of 802.11 transmission is used. The RTS/CTS is transmitted at full power so that all stations will have an idea of when the medium is available for transmission. In the algorithm Symphony [12], a two-phase algorithm is developed which also combines TPC and RA. Symphony uses a reference phase to estimate the best achievable performance and an operational phase which tunes the link to the lowest transmit power while achieving the performance estimated in the reference phase. The reference phase is also used to periodically update the estimated performance to counteract changes in the environment or mobility. To minimize receiver side interference, Symphony also employs the RTS/CTS mechanism and adaptively switches RTS/CTS on and off based on the current packet loss rate.

2.3.2 Binary Search

The binary search is an example of a divide-and-conquer algorithm. The divide-and-conquer strategy solves a problem by reducing a large problem into a number of subproblems by applying the following three steps:

1. Break the problem up into subproblems.
2. Recursively solve the subproblems.
3. Combine the solutions to the subproblems into the solution for the original larger problem.

Divide-and-conquer algorithms follow a generic pattern and can tackle a problem of size n by recursively solving subproblems of size n/b. If the problem is small enough, that is for $n \leq c$, where c is some constant, then the solution is straightforward, taking constant time $\Theta(1)$. Otherwise, if we assume that it takes $D(n)$ time to divide the problem into subproblems and $C(n)$ time to combine the solutions, then the running time of the algorithm $T(n)$ is [13]

$$T(n) = \begin{cases} \Theta(1) & \text{if } n \leq c, \\ aT(n/b) + D(n) + C(n) & \text{otherwise.} \end{cases}$$

In the case of the binary search $c = 2$, $b = 2$, $a = 1$ and $D(n) + C(n) = O(1)$ since the division into subproblems and combination of solutions takes constant time. Therefore, the running time is given by

$$T(n) = \begin{cases} \Theta(1) & \text{if } n \leq 2, \\ T(n/2) + O(1) & \text{otherwise.} \end{cases}$$

Using Master Theorem [13] we can obtain a worst-case running time for the binary search of $O(log_2(n))$.

2.4 Summary

In this chapter a background on the details and operation of the 802.11 standard were given. While there are many details from the standard that have been omitted, the details relevant to this thesis have been discussed.

A brief review of OFDM communication systems was given, with the benefits being high spectral efficiency, robustness against ISI and efficient and low cost implementation using the FFT. As discussed OFDM also suffers from high PAPR, frequency offset sensitivity and loss of efficiency due to the use of the CP.

The 802.11 standard defines four different PHY classes ranging from IR, FHSS, DSSS and OFDM. The most popular PHYs are the DSSS and OFDM PHYs found in 802.11b and 802.11a/g respectively. Due to their popularity, only the DSSS and OFDM PHYs were discussed in this chapter. The DSSS PHY provides data rates of 1, 2, 5.5 and 11 Mb/s by using spreading sequences along with various modulation schemes such as DBPSK, DQPSK and CCK. Details of the OFDM PHYs were discussed with an emphasis on how the PHYs achieve their respective data rates. In particular, 802.11a defines eight data rates of 6, 9, 12, 18, 24, 36, 48 and 54 Mb/s, which is clearly an improvement over the 11 Mb/s maximum data rate found in the DSSS PHY. Due to the popularity of the DSSS PHY found in 802.11b and the fact that the OFDM PHY of 802.11a is incompatible with 802.11b, the 802.11g or extended rate PHY was discussed which combines the high data rate found in 802.11a and compatible operation with legacy 802.11b devices.

Every PHY defined in 802.11 also requires a MAC in order to provide access to the medium in a fair and timely manner. There are different architectures available that can be used such as the BSS, IBSS and ESS. This chapter also discussed some of the important aspects of the 802.11 MAC layer. These aspects include the CSMA/CA access method and the DCF. The important details like the interframe spacings and the backoff procedure were also discussed. An example of the typical operation of the DCF was also given to explain how the MAC uses the various components to allow STAs to communicate with each other.

The operation of the 802.11 MAC and various PHY layers is important since the RAA must understand the factors affecting the throughput of an 802.11 device. Understanding these factors will allow the RAA to make a rate selection that maximizes the throughput of the STA.

Additionally, a brief overview of TPC is given including a summary of the binary search algorithm. TPC is a technique used in wireless communications to optimize the transmission power for each transmission and can be used to conserve energy when possible. The binary search algorithm is discussed and describes a method of breaking problems up into smaller subproblems and will be used as a technique for minimizing transmit power as will be described in following chapters.

Chapter 3

RelatedWork

In this chapter, popular RAAs implemented in the Atheros driver as well as those found in the literature are discussed. Their underlying premise and suitability as RAAs will be presented. The chapter begins by discussing the currently implemented algorithms available in the Atheros driver then moves on to the algorithms found in the literature.

3.1 RAAs Available in the Atheros Driver

In this section we discuss three RAAs that have been implemented on commodity hardware. The Atheros 5212 driver on both the Linux and FreeBSD platform have the following RAAs implemented and are freely available.

3.1.1 SampleRate

The SampleRate algorithm was proposed by John Bicket from MIT [14]. SampleRate begins by sending packets at the highest possible data rate. The algorithm then attempts to predict which data rate will have the highest throughput based on estimates of the expected transmission time per packet for each data rate. The expected transmission time for a frame is defined as the time to send this frame successfully, i.e., an ACK packet is received. The expected transmission time also includes the time for retransmissions and includes the random backoff as defined in the IEEE 802.11 standard. SampleRate switches rates to another data rate when another data rate's estimated per packet transmission time becomes lower than the current data rate. In order to estimate the average expected transmission time, SampleRate periodically samples data rates from a set of eligible data rates. In order to be an eligible data rate, the data rate must not have experienced four consecutive losses in the last ten seconds and it's lossless transmission time without retries must be less than the current data rate's. The estimated expected transmission time is averaged using samples collected over the last ten seconds. Samples older than ten seconds are discarded to avoid using stale information. Periodic sampling is performed every ten frames using a random data rate. The algorithm selects a data rate randomly from the set of eligible rates and sends a probe packet to assess the performance of the randomly chosen rate. If the randomly chosen data rate experiences a lower average expected transmission time than the current data rate, then the algorithm selects this data rate to transmit at.

SampleRate provides good average throughput results and generally achieves a high throughput in lossy or low-quality links [15]. SampleRate performs well in high quality links but is at times too conservative and may choose a lower rate when a higher data rate may succeed in the current conditions. SampleRate suffers from an inability to adapt to rapid changes in the environment due to it's ten second averaging window as well as it's periodic ten frame probing mechanism. Furthermore, sending probing packets wastes valuable throughput if the chosen data rate fails since the sending STA will have to perform a random backoff and then repeat the frame transmission all over again.

3.1.2 AMRR

AMRR or AARF proposed by Lacage, et. al. [16], is based on one of the earliest RAAs named Auto Rate Fallback or ARF, designed to optimize the application throughput for WaveLan II devices. ARF is a fairly simple RAA which uses success and failure counters to determine which rate to transmit at. The transmit data rate is increased every ten consecutive successfully delivered packets. ARF reduces the rate if the current data rate fails two consecutive times or immediately following a rate increase if the first packet failed at the new data rate. AARF extends this idea by introducing an adaptive threshold for increasing the rate. When a newly adapted rate fails on the first transmission, AARF immediately reduces the rate but also doubles the number of required consecutive successes for that rate with a maximum bound of 50. If the rate fails twice consecutively, then the rate is reduced and the threshold for the failed rate is reset to ten. The effect of this is that fewer failed transmissions occur by trying higher rates that do not work.

The implemented version of AARF is known as AMRR and is freely available as a RAA in the Atheros driver for Linux and FreeBSD. The Atheros driver provides a multi-rate retry mechanism with up to four rate/try pairs. These rate/try pairs are used for retransmissions and indicate a rate to use and the number of times to retry at that rate. r_0 indicates the first rate to transmit at, while r_1, r_2 and r_3 represent the second, third and fourth rate to attempt retransmission, respectively. Additionally, c_0 represents the number of times to try the first rate at, while c_1, c_2 and c_3 represent the second, third and fourth rate counts to attempt retransmission, respectively. AMRR modifies the rate/try pairs using the binary exponential backoff described in AARF. To ensure that the short term variations of the wireless medium are quickly acted upon, the AMRR RAA chooses $c_0 = 1$, $c_1 = 1$, $c_2 = 1$ and $c_3 = 1$.

The rates r_1 and r_2 are determined from r_0, that is, r_1 is one data rate less than r_0 and r_2 is one data rate less than r_1 with r_3 always being the lowest data rate. r_0 is determined from the previous value of r_0.

With the addition of adaptive thresholds, AARF is able to increase the time interval between rate increases in stable channels and thus produce fewer rate fluctuations than ARF. Unfortunately, both ARF and AARF both assume that losses are due to channel conditions and thus perform poorly in high traffic scenarios, where losses are dominated by collisions. As exposed in [14], AARF also spends too much time sending packets at data rates that work poorly, in which case throughput is significantly reduced. In addition, the MRR chain used in the implemented AMRR algorithm amounts to guessing at the reason for failures and then automatically reduces the rate when in fact the failed transmission may be due to a collision or a transient fade.

3.1.3 ONOE

Onoe [17] is a credit based RAA developed by the creators of the MadWifi open source device driver for Atheros 802.11 devices. The Onoe algorithm attempts to find the highest data rate that has a packet loss ratio less than 50 percent. According to the author of SampleRate, Onoe is much less sensitive to individual packet failure than the ARF algorithm, is relatively conservative and can take time to stabilize since it will only increase or decrease a rate after a relatively long period of time whether a data rate is a successful one or a failing one.

Onoe starts at a median data rate, 24 Mbps for 802.11a/g and 11 Mbps for 802.11b with the credits for all data rates initialized to zero. The algorithm then keeps track of success and failure credits for various data rates over a period of one second. Onoe increases a data rate's credit count by one if less than ten percent of the packets needed a retry, otherwise if more than ten percent of the packets needed a retry, the credit count is decremented by one. After a one second period if the current data rate has ten or more credits, Onoe will increase the data rate to the next higher data rate. Alternatively, after the one second period if either no packets have succeeded or ten or more packets have been sent and out of those packets the average number of retries is greater than one, then the rate is decremented to the next lower date rate. In all other cases, the algorithm continues at the current data rate. Credit counters are reset to zero whenever a data rate change is made.

Clearly, Onoe is insensitive to rapidly varying conditions, since the algorithm can only change rates at least once per second. Furthermore, Onoe can only switch to adjacent rates when the algorithm does decide to change rates. The credit thresholds in Onoe are static and are not adaptive to channel conditions. In addition, Onoe tries to find data rates that have less than 50 percent loss, which is only applicable for data rates found in 802.11b where the rates are separated by large gaps. In 802.11a/g the data rates are grouped much closer together and as such a data rate that experiences up to 50 percent loss will have a much lower throughput than another data rate below it that may perform better.

3.2 RAAs Available in the Literature

In this section, popular RAAs that can be found in the literature but may not be implemented in hardware is discussed. Most of the RAAs found in the literature are more complicated and thus more difficult to implement in real hardware. Some of the RAAs often make incompatible changes to the 802.11 standard or are unimplementable in practice. Despite some of these RAAs impracticality, the ideas developed in the literature are often useful and more up to date with current developments.

3.2.1 SmartRate

SmartRate [18] is a dynamic RAA for 802.11 WLANs which shows good performance in both static and mobile scenarios. The algorithm is considered dynamic since it can update various parameters under different channel conditions.

SmartRate uses what the authors have termed Physical PER or PPER. The authors claim that by using fragmentation that fits with the IEEE 802.11 standard they can achieve a PER that is free from collisions and represents the true channel PER. In addition, SmartRate uses the variance of the RSSI to detect channel volatility and fine tunes various parameters according to the current channel conditions. The algorithm selects a rate based on the estimated throughput found from the PPER which is gained through sampling of different rates which in turn is dictated by the channel volatility found from the RSSI variance.

SmartRate uses the ACK RSSI variance to categorize the current wireless channel conditions into three environments, low (stationary), medium (for mild variations cause by the movement of people and objects) and high (mobile) volatility. In addition, the algorithm uses fragmentation to achieve PPER isolation free from collisions. Fragmentation is only performed a fraction of the time in order to reduce overhead and maintain efficiency. According to the 802.11 standard, the first fragment is unprotected, i.e., it can be subject to collisions, however, subsequent fragments are free from collision if the first fragment is successfully delivered. Therefore, the PER of the fragmented packets represent the real channel conditions independent of collisions incurred during a congested scenario. SmartRate also divides the PER into three packet size bins since different packet sizes experience different PERs. The bin sizes are chosen according to a tri-modal distribution of the most popular packet sizes found in the Internet.

SmartRate's rate update interval depends on the RSSI variance and is set to 500 ms for low volatility conditions, 250 ms for medium volatility and 100 ms for high volatility conditions. The algorithm also uses the MRR chain with the current rate tried twice, the back up rate tried twice and finally, the lowest rate four times for normal unfragmented packets. The fragmented packets are used to sample different rates. Sampling is performed 20 percent of the time, 40 percent of the time and 60 percent of the time in low, medium and high volatility conditions respectively. The sampling of rates is then divided among rates above, below and at the current rate with the percentage of time spent sampling based on the current channel volatility. SmartRate samples rates below the current rate 10, 10 and 20 percent of the time for low, medium and high volatility conditions respectively. Sampling rates above the current rate is performed 50, 50 and 60 percent of the time for low, medium and high volatility conditions respectively. Finally, the current rate is sampled 40, 40 and 20 percent of the time in low, medium and high volatility conditions respectively. The rates to sample at, whether up or down, are based on their PER which indicates their probability of being successful. Packet sizes are sampled at 40 bytes, 750 bytes and 1500 bytes. SmartRate then calculates the expected transmission time and takes the reciprocal of the expected transmission time and multiplies it by the success probability for a given rate to get the throughput. The algorithm then picks the rate based on this expected throughput.

The algorithm begins by using the RSSI average in order to start gathering statistics and starts in the high volatility mode in order to be able to adapt quickly at the beginning until the algorithm is able to determine what kind of channel conditions are present. The authors show that the algorithm performs better than both SampleRate and AMRR in both static and mobile scenarios. The authors implemented SmartRate on the MadWifi Atheros device driver for all comparison experiments.

3.2.2 Smart Sender

The authors of Smart Sender [15] propose a new RAA which monitors and adapts to changes in the link quality by tuning the transmission data rate. Smart Sender maintains both long-term and short-term statistics, which are carefully selected to better reflect channel conditions. In order to improve the responsiveness of the algorithm, Smart Sender uses the RSSI of ACK packets to predict the dynamics of the link conditions at the receiver for rate adaptation at the sender.

The algorithm operates by probing at data rates that may outperform the current date rate under the following conditions: the expected throughput at the new rate exceeds the current rate; or transmissions at the current rate are suboptimal, and scaling up/down can actually improve performance. If the algorithm determines that the probing rate is not performing as expected, probing is ceased immediately and the algorithm resorts to the previous long term transmission rate, which is selected based on periodical analysis of transmission history. Smart Sender also uses the expected transmission time as long term statistics for system stability. Smart Sender calculates the expected throughput using the expected transmission time at the current data rate and one data rate above and below the current data rate. The expected throughput of the data rates above and below are multiplied by a proportional value based on the ratio of the higher data rate to the lower data rate. The strategy here is that the larger the rate values differ, the more difficult it becomes to make a data rate switch.

In addition to using the expected throughput and probing, Smart Sender uses counters such as consecutive successes, consecutive ACKs and consecutive failures as short term statistics for system responsiveness and is termed the statistics adaptor. Based on these counters, two thresholds are defined: the failure threshold and the success threshold. The failure threshold is kept small to allow the algorithm to quickly react to deteriorating channel conditions. There are two values for the failure threshold, 2 and 4 with 2 being the default and 4 chosen if the packets fall below a certain size, i.e, for small packet sizes. The success threshold is a dynamic threshold and follows the rules of the TCP congestion window. The threshold follows the multiplicative increase and linear decrease when modifying its value. If the success threshold is reached, then the rate increases and the success threshold is multiplicatively increased. The first transmission at the new rate is termed the recovery rate, if the recovery rate fails, the algorithm immediately returns to the previous rate and increases the success threshold once again to stay away from the failing rate. However, when the rate is decreased, the success threshold is linearly decreased to encourage rate increases by having a low success threshold. The minimum success threshold is set to eight and the multiplicative increase value is set to 2. The maximum success threshold value is 50 with the minimum linear decrease value being 6. All counters are reset upon data rate change.

Smart Sender also uses the RSSI as a regulator of sorts and is termed the RSSIA Regulator. The rate selected by the statistics adaptor must be feasible according to the average RSSI. That is, the RSSI must be high enough to support the new data rate. The RSSI is also used to allow a rate increase without reaching the success threshold if the channel is seen to be improving rapidly.

Smart Sender employs the use of the MRR chain with the number of tries for the first rate $c_0 = 2$ and c_1, c_2 and c_3 all set to one. The data rate r_0 is chosen by the algorithm with r_1, r_2 and r_3 following the same strategy as ONOE, that is r_1 is one data rate below r_0, r_2 is one data rate below r_1 and r_3 is the lowest data rate available.

In their paper, the authors compare Smart Sender with ONOE and show it to have superior performance, however, a direct comparison with SampleRate is not shown despite SampleRate's ability to outperform ONOE. Based on the performance evaluation presented, Smart Sender shows good performance and seems adaptive to changing channel conditions, however as mentioned, a direct comparison to SampleRate is absent. The authors performed all experiments by implementing the algorithm on the MadWifi Atheros device driver.

3.2.3 CHARM

CHARM is an acronym for CHannel Aware Rate selection algorithM presented by the authors in [19]. The algorithm uses the average RSSI to estimate the SNR at the receiver. CHARM assumes that the wireless channel is symmetric and that the channel conditions at the receiver are similar to those found at the transmitter. Using the estimated SNR, CHARM calculates the path loss in order to estimate the SNR at the receiver. CHARM continuously monitors all packets sent by potential receivers in order to calculate the path loss. However, the transmitter needs the transmit power used and the noise level at the receiver to estimate the receiver SNR before packet transmission. In order to obtain this information an additional information element is introduced to all beacon and probe packets specified in the 802.11 standard. This change makes CHARM incompatible with STAs that do not have this modification. Using the predicted path loss, the transmitter estimates the SNR at the receiver and uses it to look up the best transmission rate in a rate selection table that lists the minimum required SNR threshold for each destination and for each transmission rate.

The average RSSI is calculated using a linearly weighted moving average (LWMA).

The use of the LWMA is due to implementation issues in the MadWifi driver, since floating point support is not available at the driver level. In addition, CHARM identifies small-scale transient fades and discards those samples when calculating the RSSI average. The algorithm first checks the value of the RSSI sample and compares it to the current average RSSI. If the current RSSI sample's deviation is larger than a certain threshold then the sample is marked as transient and not used in the average calculation. The algorithm then monitors the next RSSI sample and if the next sample has a deviation from the average less than the threshold, then the sample marked as transient is discarded. Otherwise, the change in RSSI is not transient and the RSSI sample marked as transient is included in the average calculation. CHARM uses a deviation threshold value of five in the determination of transient samples.

CHARM also employs the use of SNR thresholds to adapt to changing channel conditions. The SNR thresholds are updated according to success and failure counts for each data rate and each data rate has an associated threshold. As an example, consider that if 11 Mbps has a SNR threshold of 10 dB but experiences successful transmission at 9 dB, then the packet success counter for bin -1 is incremented. Alternatively, if 11 Mbps is found to fail at 12 dB, then the failure bin for +2 is incremented. In this way, CHARM can determine if rates are experience success or failure at thresholds other than their default. If a rate is found to be successful at a lower SNR dB, then the threshold should be lowered and if a rate is failing at a higher SNR dB, then the threshold for the data rate should be increased.

CHARM first invokes the path loss prediction algorithm to estimate the current path loss to the receiver. Then, using the transmit power and noise level at the receiver, which is provided by the receiver, the SNR at the receiver can be estimated. Using the estimated SNR as an index, CHARM lookups the associated data rate based on the SNR thresholds. Upon selection of the data rate, the MRR chain is then filled out. If the first rate is at least 36 Mbps, then the second data rate is to two data rates less than the selected data rate. Otherwise, if the data rate is less than 36 Mbps, the second data rate is chosen to be one less than the selected data rate. The third data rate, if the first is at least 36 Mbps, is chosen as 11 Mbps, otherwise it is selected as one below the second data rate. The final fourth data rate is always selected to be 1 Mbps in any case with the number of tries set to the maximum for that data rate. The reasoning for selecting the 1 Mbps data rate as the final data rate, is that the authors feel that a successful packet delivery is more important than the time wasted by sending data at the lowest data rate since a successful packet delivery implies that an ACK packet will be transmitted following successful delivery which will provide valuable up to date channel information in the form of RSSI.

CHARM is shown by the authors to perform better than SampleRate in most cases in both static and mobile scenarios. The algorithm uses SNR estimates based on the RSSI of ACK, probe and beacon packets and noise levels present at the receiver. The estimated SNR is then used as a lookup into a table of rates based on SNR thresholds. The SNR thresholds are dynamic and updated based on success and failure counters. The algorithm performs well, but requires a modification to certain packets in order to obtain the noise level at the receiver for the SNR estimation. This change makes the algorithm incompatible with any STA not using this algorithm and makes CHARM infeasible in most situations.

3.2.4 RBAR

RBAR [20], stands for Receiver Based Auto Rate and is a RAA that, unlike most RAAs, resides at the receiving STA. Placing the RAA at the receiver obviates the need to for the transmitter to determine the channel conditions at the receiver. In order for a transmission to be successful, the SNR at the receiver must support the MCS chosen by the RAA. Since the RAA resides at the transmitter, the transmitter must attempt to determine the SNR at the receiver. In addition, due to the rapidly changing conditions of the wireless channel, determining the best data rate to transmit at means minimizing the delay in obtaining the channel conditions at the receiver. Therefore, the authors have implemented RBAR at the receiver in order to make the best possible estimates about the channel quality and to ensure the most reliable rate selection.

To allow for data rate selection at the receiver, RBAR uses the RTS/CTS exchange prior to packet transmission to deliver information from the receiver to the transmitter. Typically, the RTS/CTS packets contain the expected duration of the upcoming data transmission, known as the NAV. In RBAR, the NAV information is replaced with the modulation rate and packet size which is placed by the receiver in the CTS packet. Based on this information any STA that overhears the CTS, including the transmitter, can still calculate the duration information that is normally found in the NAV. The authors also introduce a new MAC data frame which includes a header check sequence to protect what the authors have termed a reservation subheader. This subheader is necessary due to the modifications of the RTS/CTS frames. The RTS/CTS frames no longer carry the reservation information and as such, the calculated duration information may differ between receiver and transmitter and now carries a tentative reservation time. The final reservation is contained in the newly added reservation subheader of the transmitted data packet.

RBAR uses the instantaneous RSSI value obtained from the RTS packet as a channel quality estimation. Using this RSSI value, the rate is chosen by comparing the RSSI to a series of thresholds representing the desired performance bounds of the available data rates. In RBAR, the thresholds are obtained from a simulation model using a Rayleigh fading model. From the simulation, SNR values for each data rate at a BER of 10^5 are obtained and used as the thresholds for data rate selection.

The authors show that RBAR outperforms ARF, but do not compare it to any other algorithms. RBAR shows good performance in both static and mobile scenarios and by placing the RAA at the receiver presents a novel solution to rate selection. Unfortunately, RBAR requires extensive changes to the 802.11 standard, such as modifications to the RTS/CTS packets, plus the addition of a subheader to data packets, thus making it non-compliant with the 802.11 standard. Furthermore, the use of RTS/CTS exchange for every data transmission represents additional overhead that is not required in traditional RAAs. The authors also neglect to mention how to translate the RSSI values obtained in the RTS/CTS exchange to SNR used as the threshold values obtained from the simulated BER curves. The RSSI is not a measure of SNR and is a proprietary value unique to each chipset and must be mapped to SNR values using real world measurements.

3.2.5 RRAA

RRAA was proposed by the authors in a paper titled Robust Rate Adaptation for 802.11 Wireless Networks [21]. The algorithm uses a short-term loss ratio to ensure adaptability and to determine the data rate for which to transmit at. Maintaining short-term loss statistics keeps only the most recent samples, which ensures the loss statistics is as up to date as possible and therefore a dependable estimate of the channel conditions. The algorithm also uses the RTS option of the 802.11 standard, but employs an adaptive RTS filter to selectively enable or disable the RTS/CTS exchange which permits collision detection with minimal overhead. The authors compare RRAA to ARF, AARF and SampleRate in static and mobile scenarios with and without hidden terminals and is shown to have improved performance over all three RAAs.

RRAA starts with the highest data rate and uses what the authors term an estimation window. Data rates are only updated once every estimation window. Whenever a new rate is chosen, the new rate is chosen to transmit during the next estimation window frames. The loss estimation is estimated based on how many frames over the estimation window are lost. RRAA uses the following equation to estimate the loss ratio:

$$P = \frac{numberoflostframes}{numberoftransmittedframes}$$

where both numbers of lost frames and transmitted frames are counted over the window and include all retries. Once the estimation window is complete, a new rate is chosen based on the estimated loss ratio P. The algorithm uses two thresholds to determine rate increase or decrease, termed opportunistic rate increase threshold and maximum tolerable loss ratio respectively. If the loss ratio is higher than the maximum tolerable loss threshold, then the data rate is decreased to the next lower data rate. If the loss ratio is smaller than the opportunistic rate increase threshold, the data rate is increased to the next higher data rate. In these two cases, a new estimation window is begun for the newly selected data rate. If the current loss ratio, P, is between the two thresholds then the data rate is left unchanged. A new estimation window is started every time a new data rate is selected, otherwise, the estimation window continues to slide forward. The thresholds are used under the assumption that low loss ratios indicate a good channel and high loss ratio indicate a poor channel.

The maximum tolerable loss threshold is calculated according to the transmission time for the current rate and the transmission time for the rate one below the current rate. The authors use the minimum backoff time in the calculation with the result then multiplied by a tunable constant, $\alpha = 1.25$. The opportunistic rate increase threshold is calculated by taking the maximum tolerable loss threshold and dividing it by another tunable constant, $\beta = 2$. The estimation window size used is smaller for lower data rates and larger for high data rates. This is because at higher data rates, more packets can be sent in the same amount of time and the RRAA attempts to balance adaptability with stability by making the estimation window size small enough to be adaptive but large enough such that random fluctuations do not disturb the actual value.

RRAA also uses an adaptive RTS filter to minimize collision losses for the loss ratio estimation. The algorithm selectively enables and disables RTS/CTS exchange to suppress collision losses using an RTS window. The RTS window size is initially set to zero, thus effectively disabling the RTS/CTS exchange. If the last frame was lost without RTS, then the RTS window is incremented by one. When the last frame was lost with RTS or the last frame succeeded without RTS, the RTS window is set to half its previous value because the last frame did not encounter a collision condition. When the last frame succeeded with RTS, the RTS window is left unchanged. The authors state that more frames are sent with RTS on when collision losses are severe which protects the loss ratio estimation from being poisoned by collision losses. Alternatively, when collision losses are mild, the RTS window is small due to the multiplicative decrease and the overhead of the RTS/CTS exchange is kept to a minimum.

RRAA uses short-term loss ratios and static thresholds to determine the rate and adaptively enables RTS/CTS exchanges when it determines that losses are due to collisions. The authors have implemented RRAA on a programmable AP platform and compare it's performance to ARF, AARF and SampleRate. In the performance comparisons, RRAA shows improvement over ARF, AARF and SampleRate in mobile and static scenarios, including scenarios with hidden terminals. The algorithm performs well as explained but suffers from the fact that it uses the RTS/CTS exchange which incurs overhead, but may also not be enabled on other terminals. Thus, RRAA is only compatible with other STAs running RRAA.

3.2.6 OAR

Opportunistic Auto Rate (OAR) [22] is an extension of the RAA RBAR. OAR is termed opportunistic since it can take advantage of good channel conditions when they arise by sending multiple back to back data packets during these periods of good channel conditions. The authors make the argument that the channel coherence time is typically longer than a single packet transmission time and thus lasts for multiple packet transmissions. OAR opportunistically takes advantage of this channel coherence time to send multiple packets within the channel coherence time with a high probability of successful delivery. To accomplish this, OAR extends RBAR and makes use of the RTS/CTS exchange to reserve the wireless medium.

OAR chooses a data rate in the same manner as in RBAR, however, when setting the NAV duration information for the RTS/CTS, it calculates the transmission time based on the lowest or base data rate, regardless of the actual data rate selected by the RAA. Since it takes less time to transmit data packets for higher data rates, every STA transmitting at a higher data rate than the basic or lowest rate can then send multiple packets in the same amount of time for a packet at the lowest data rate. OAR then uses fragmentation to send back to back packets with each fragment/ACK pair acting as a virtual RTS/CTS handshake. In this way, the transmitting STA is guaranteed access to the medium for a time t, the time to send the data packet at the base data rate. If channel conditions allow and the STA can use a higher data rate, then the STA simply uses the fragmentation mechanism in 802.11 to send multiple data packets at the higher data rate but in the same transmission time t.

OAR is shown to outperform RBAR since it can take advantage of good channel conditions to send data packets in bursts. OAR also provides increased fairness since all STAs are given the same amount of time to transmit data packets and for those STAs that are experiencing good channel conditions are able to send a higher number of data packets. However, OAR uses the RTS/CTS handshake for every transmission, thus introducing the overhead associated with that technique, however the use of transmission bursts reduces the overhead. OAR also assumes that the channel conditions remain constant for the transmission time t, which may not always be the case, thus OAR may not be able to react to rapid channel quality fluctuations.

3.2.7 RSSLA

In the paper Link Adaptation Strategy for IEEE 802.11 WLAN via Received Signal Strength Measurement [23], the authors propose a novel method for IEEE 802.11 data rate selection by using the RSS of received packets, dubbed Received Signal Strength Link Adaptation. The basic premise is that the algorithm adapts the transmission rate by measuring the RSS of frames received from the AP. The authors assume that the RSS has a linear relationship on average with the SNR found in the wireless channel and that changes in the RSS indicate changes in the SNR. This changing RSS necessitates adaptation of the data rate accordingly.

The algorithm measures the RSS and adapts the data transmission rate based on comparison with thresholds. RSSLA stores and updates twelve different thresholds for data rate adaptation. The algorithm begins by setting all thresholds to an initial value of zero but dynamically updates the thresholds during operation. The thresholds represent the minimum RSS required to support each data rate. Each STA updates the average RSS by measuring the RSS of any frame addressed to itself or a broadcast/multicast frame addressed to it from the AP. When the transmission of a frame is required, the thresholds are updated based on the successful or unsuccessful delivery of the frame at the current data rate. As an example, consider the following, if a frame at a given data rate is unsuccessful, the threshold for that rate is subsequently raised using the average RSS currently being used by the STA. Alternatively, if the transmission for a data rate is successful, then the threshold for that rate is reduced according to the average RSS currently being used by the STA. When selecting a data rate to transmit at, RSSLA will consider the values of RSS average thresholds, frame sizes and the number of retransmission attempts. RSSLA will then automatically decrease the PHY rate when the of retransmissions exceed the current retransmission limit. Following each retransmission, the thresholds are correspondingly updated.

Since the delivery probability of a frame is a function of the frame size, RSSLA keeps track of three frame sizes, 0-100 bytes, 100-1000 bytes and 1000-2400 bytes. These frame sizes are chosen based on known internet traffic statistics for popular frame sizes as well as having a widely different SNR requirement among the three frame sizes.

RSSLA uses the following equations to determine the RSS threshold and RSS average for each data rate:

$$Th = a_1 \times Th + a_2 \times RSSRSS_{avg} = a_3 \times RSS_{avg} + a_4 \times RSS$$

where RSS is the received signal strength measured from the latest received frame and the values a_1, a_2, a_3 and a_4 determine the speed of the adaptation in the exponential moving average.

The authors in this paper propose a novel RAA which uses the RSS of received frames to determine the data rate to transmit at. The use of dynamic thresholds helps to alleviate inaccuracy in RSS measurements. The authors show that the algorithm performs close to optimal in most cases in simulated scenarios. However, due to the thresholds being based on frame delivery success or failure, the algorithm can become sensitive to frame losses due to collisions. In addition, the authors do not compare to any other algorithms and is only evaluated using simulation.

3.3 Summary

In this chapter, popular RAAs implemented in the Atheros driver as well as those found in the literature were discussed. The chapter began with the three RAAs available in the Atheros driver, namely, SampleRate, AMRR and ONOE, then continued by discussing popular RAAs found in the literature.

SampleRate bases data rate selection upon an EWMA of the estimated RTT and provides good average throughput results in lossy or low-quality links. SampleRate also performs well in high quality links but is at times too conservative and may choose a lower rate when a higher data rate may succeed in the current conditions. SampleRate suffers from an inability to adapt to rapid changes in the environment due to it's ten second averaging window.

AMRR is based on one of the earliest RAA, ARF and uses success and failure counts to determine which data rate to transmit at. With the addition of adaptive thresholds to ARF, AARF is able to increase the time interval between rate increases in stable channels and thus produce fewer rate fluctuations than ARF. Unfortunately, AARF assumes that all losses are due to channel conditions and thus, may perform poorly in high traffic scenarios.

Onoe is a credit based RAA developed by the creators of the MadWifi open source device driver for Atheros 802.11 devices. The Onoe algorithm attempts to find the highest data rate that has a packet loss ratio less than 50 percent and can only change data rates once per second. Additionally, using static credit thresholds, Onoe is thus insensitive to rapidly varying channel conditions.

SmartRate is a dynamic RAA for 802.11 WLANs which shows good performance in both static and mobile scenarios. The algorithm uses the RSSI variance to detect channel condition changes and also employs the use of the PPER to get a true measure of the current channel conditions. The authors show that the algorithm performs better than both SampleRate and AMRR in both static and mobile scenarios. The authors implemented SmartRate on the MadWifi Atheros device driver for all comparison experiments.

Smart Sender maintains both long-term and short-term statistics, as well as the RSSI of ACK packets to predict the dynamics of the link conditions at the receiver. The authors compare Smart Sender with ONOE and show it to have superior performance, however, a direct comparison with SampleRate is absent. Based on the performance evaluation presented, Smart Sender shows good performance and seems adaptive to changing channel conditions. The authors performed all experiments by implementing the algorithm on the MadWifi Atheros device driver.

CHARM uses the average RSSI to estimate the SNR at the receiver and assumes that the wireless channel is symmetric. Using the estimated SNR, CHARM calculates the path loss then looks up the best transmission rate in a rate selection table that lists the minimum required SNR threshold for each destination and for each transmission rate. In order to calculate the path loss an additional information element is introduced to all beacon and probe packets specified in the 802.11 standard. CHARM thus requires modifications that make the algorithm incompatible with any STA not using CHARM. The authors show that CHARM outperforms SampleRate in both static and mobile scenarios.

RBAR, unlike most RAAs, resides at the receiving STA which allows the transmitter to determine the channel conditions at the receiver through feedback. RBAR makes use of the RTS/CTS exchange in order to transfer information about the conditions at the receiver to the transmitter. Through performance evaluations, the authors show that RBAR outperforms ARF, but do not compare it to any other algorithms. Unfortunately, RBAR also requires extensive changes to the 802.11 standard, thus making it non-compliant with the 802.11 standard.

RRAA uses a short-term loss ratio to ensure adaptability and to determine the data rate for which to transmit at. The authors argue that short-term statistics represent the most up to date information and is more reliable than long-term statistics. The algorithm also employs an adaptive RTS filter to selectively enable or disable the RTS/CTS exchange. The authors compare RRAA to ARF, AARF and SampleRate in static and mobile scenarios with and without hidden terminals and is shown to have improved performance over all three RAAs.

OAR is based on RBAR and is termed opportunistic since it can take advantage of good channel conditions by sending multiple back to back data packets during periods of good channel conditions. To accomplish this, OAR extends RBAR and makes use of the RTS/CTS exchange to reserve the wireless medium. OAR is shown to outperform RBAR since it can take advantage of good channel conditions to send data packets in bursts. However, OAR uses the RTS/CTS handshake for every transmission, thus introducing the overhead associated with that technique and also assumes that channel conditions remain constant for the transmission time, thus OAR may not be able to react to rapid channel quality fluctuations.

RSSLA is a novel method for data rate selection by using the RSS of received packets. The algorithm adapts the transmission rate by measuring the RSS of frames received from the AP. The authors assume that the RSS has a linear relationship on average with the SNR. The authors show that the algorithm performs close to optimal in most cases in simulated scenarios but can be sensitive to collision losses due to the use of thresholds based on frame loss.

Each of the algorithms described in this section provide novel ways of solving the problem of data rate selection and offer reasonable performance, however, none of the algorithms are truly situation-aware. Each algorithm performs well in some conditions, but performs poorly in other conditions. This highlights the inability of contemporary algorithms to detect their operating conditions and thus, their situation. In the following chapters, two situation-aware algorithms are described to address this problem.

Chapter 4

RAA Design and Implementation

In this chapter, the design of SARA is presented, followed by the details of the implementation on the Atheros device driver for FreeBSD. SARA determines its' current environmental conditions by first collecting RSSI and PER data and then uses an adaptive table aligned to the current channel conditions to select an appropriate data rate. The algorithm begins by first discovering its environment and then, by using the average RSSI, a set of feasible data rates from the calibrated table is chosen. Finally, the algorithm chooses from the available feasible data rates by using the PER. The algorithm also monitors current channel conditions and if a large change in environmental factors is detected, the algorithm quickly re-calibrates and aligns the adaptive table to the new channel conditions. In this way, SARA can adapt to quickly changing channel conditions, but avoid unnecessary data rate changes. In addition, SARA avoids the use of probing packets and RTS/CTS handshaking, which reduces overhead and improves adaptability to rapid changes. The algorithm is insensitive to collisions since the average RSSI is used to determine the feasible set of data rates regardless of the current contention conditions.

SARA provides rapid data rate adaptation and maximizes throughput, is resistant to collisions and also provides stable data rate selection while being perceptive to environmental changes. SARA consists of three major components, the calibration phase, the RSSI selector and environment detector and the PER fine tuning.

4.1 Theoretical Considerations in the Design of SARA

This section describes the theoretical considerations used in the SARA design. The operation of SARA hinges on a shifting of the PER-RSSI curves, which allows the algorithm to adjust the data rate based on the current environmental conditions. The following theoretical derivations support the algorithms ability to map PER curves in an AWGN channel to one in a fading channel. The system model is first described and then the EESM technique for mapping the PER curves in AWGN to a fading channel is described. The simulation and measurement results of the PER-RSSI curve shifting technique are discussed followed by a derivation of the window size used in the collection of PER and RSSI statistics.

4.1.1 Shift Property of PER-RSSI Curves under Different Channel Situations

In this section, a theoretical background of the shifting of the PER-RSSI curves is given which is performed during the calibration phase to align the algorithms rate table to the current channel conditions. As described in previous chapters, RSSI and PER are two general PHY parameters that is provided by the Atheros IEEE 802.11 chipset and can be exploited for rate adaptation. RSSI is fast and does not depend on packet type nor data rate, while PER is simple and stable, but time-consuming. It is widely acknowledged that there is a one-to-one mapping between the RSSI and PER. However, for different configurations, such as data rate, packet size, scenario and so on, this mapping relation between RSSI and PER may change [24]. Two different channel states with the same average RSSI can experience drastically different PER performance. Therefore, it is of great interest to analyze the property of the PER-RSSI mapping relationship under different channel scenarios.

OFDM with bit-interleaved coded modulation (BICM-OFDM) is utilized in IEEE 802.11a to achieve improved error rate performance when compared to uncoded OFDM systems. Exponential Effective SINR Mapping (EESM) [25], which was first proposed and patented in 2004, provides a useful method for the error rate analysis of coded OFDM in fading channels. EESM has been included in the 3GPP and WiMAX forum feasibility studies for both LTE and WiMAX, and has shown to achieve good performance [26], [27], [28], [29], [30]. The EESM technique is exploited in the following analysis related to the RSSI-PER mapping relationship.

4.1.1.1 System Model

The received signal of each subcarrier in an OFDM symbol can be modeled as

$$Y(k) = H(k)X(k) + W(k), \quad k = 0, 1, ..., N-1 \tag{4.1}$$

Let $h(l)$ denote the time domain channel impulse response of the lth path with L paths in total. It is assumed that $h(l),\ l = 0, 1, ..., L-1$ are independent (but not necessarily identically distributed) zero-mean complex Gaussian random variables with variances $\sigma_l^2,\ l = 0, 1, ..., L-1$. Also, let $H(k)$ be the frequency domain channel impulse response that is obtained from an N-point Fourier Transform of $h(l)$. Therefore, $H(k)$ is also a zero-mean complex Gaussian random variable with variance of $\sigma_H^2 = \Sigma_{l=0}^{L-1} \frac{\sigma_l^2}{N}$. It should be noted that $H(k),\ k = 0, 1, ..., N-1$ are identically distributed but not necessarily independent. It is trivial to prove that $|H(k)|$ is a Rayleigh distributed random variable and $|H(k)|^2$ is a Chi-square random variable with two degrees of freedom. Then, the average signal ($\overline{RSS}$) strength of the received signal

$(Y(k))$ can be calculated as

$$\begin{aligned}\overline{RSS} =& E\left\{\sum_{k=0}^{N-1}|Y(k)|^2\right\}\\ =& E\left\{\sum_{k=0}^{N-1}(|H(k)||X(k)|+|W(k)|)^2\right\}\\ =& E\left\{\sum_{k=0}^{N-1}\left(|H(k)|^2|X(k)|^2+2|H(k)||X(k)||W(k)|+|W(k)|^2\right)\right\}\end{aligned} \tag{4.2}$$

Since $H(k)X(k)$ and $W(k)$ are independent variables, their expectation is zero [31], therefore we can rewrite the above as

$$\begin{aligned}\overline{RSS} =& E\left\{\sum_{k=0}^{N-1}\left(|H(k)|^2|X(k)|^2+2|H(k)||X(k)||W(k)|+|W(k)|^2\right)\right\}\\ =& E\left\{\sum_{k=0}^{N-1}\left(|H(k)|^2|X(k)|^2+|W(k)|^2\right)\right\}\\ =& P_S E\left\{\sum_{k=0}^{N-1}|H(k)|^2\right\}+\sigma_w^2\end{aligned} \tag{4.3}$$

where P_S is the expected value of the signal $|X(k)|^2$ or signal power and σ_w^2 is the variance of the AWGN'. The instantaneous RSS of an OFDM symbol can be described as

$$RSS = \sum_{k=0}^{N-1}\left(|H(k)|^2 P_S + |W(k)|^2\right) \tag{4.4}$$

Similarly, the SINR of an OFDM symbol can be expressed as

$$\Gamma = \frac{P_S\sum_{k=0}^{N-1}|H(k)|^2}{\sigma_w^2} \tag{4.5}$$

while the instantaneous SINR for the kth subcarrier Γ_k can be written as

$$\Gamma_k = \frac{P_S|H(k)|^2}{\sigma_w^2} \tag{4.6}$$

Since the calculation of RSSI used in the Atheros IEEE 802.11 chipset is not available to the public, the relationship between RSSI and SINR can not be theoretically determined. Measurements were carried out at UWO to study this relationship and it has been found that there is a linear mapping between the RSSI and the SINR. Therefore, in the following analysis, RSSI and SINR are considered equivalent.

4.1.1.2 EESM-based Analysis for the Shift Property of PER-RSSI Curves

The basic idea of EESM is to find an equivalent SINR in an AWGN channel that results in the same error rate in the fading channel. This can be accomplished by using the Union-Chernoff bound to relate the error probability to the corresponding SINR in each subchannel. The EESM technique maps a set of instantaneous subcarrier SINRs into a single effective SINR. This effective SINR is then used to find an estimate of the SER from the basic AWGN link level performance. The EESM technique can be mathematically expressed as [26]

$$\Gamma_{eff} = -\lambda log\left(\frac{1}{N}\sum_{k=0}^{N-1} e^{-\frac{\Gamma_k}{\lambda}}\right) \tag{4.7}$$

where λ is used to match the EESM to a specific combination of modulation scheme and coding rate. λ is a parameter that depends only on the modulation scheme and coding rate, but is independent of the channel models. The value for each combination of MCS can be calibrated through adequate link-level simulations or real measurements with various optimization criteria. A reference set of calibrated values for the MCSs of interest can be found in Table 4.1 obtained from the Master's thesis titled Calibration and Evaluation of the Exponential Effective SINR Mapping (EESM) in 802.16 [32]. The effect of the channel condition, interference as well as

Data Rate	Modulation Scheme	Coding Rate	λ
6 Mbps	BPSK	1/2	0.91
9 Mbps	BPSK	3/4	0.98
12 Mbps	QPSK	1/2	1.57
18 Mbps	QPSK	3/4	1.69
24 Mbps	16-QAM	1/2	4.56
36 Mbps	16-QAM	3/4	7.33
48 Mbps	64-QAM	2/3	23.3
54 Mbps	64-QAM	3/4	27.7

Table 4.1: λ values for different MCSs.

noise is reflected in the term $log\left(\sum_{k=0}^{N-1} e^{-\frac{\Gamma_k}{\lambda}}\right)$, which can be rewritten as

$$log\left(\sum_{k=0}^{N-1} e^{-\frac{\Gamma_k}{\lambda}}\right) = log\left(\sum_{k=0}^{N-1} e^{-\frac{P_S|H(k)|^2}{\lambda\sigma_w^2}}\right) \tag{4.8}$$

Letting $|H(k)|^2 = \overline{|H|^2} + \Delta_k$ gives

$$\begin{aligned} log\left(\sum_{k=0}^{N-1} e^{-\frac{P_S|H(k)|^2}{\lambda\sigma_w^2}}\right) &= log\left(\sum_{k=0}^{N-1} e^{-\frac{P_S\left(\overline{|H|^2}+\Delta_k\right)}{\lambda\sigma_w^2}}\right) \\ &= -\frac{P_S\overline{|H|^2}}{\lambda\sigma_w^2} + log\left(\sum_{k=0}^{N-1} e^{-\frac{\Delta_k P_S}{\lambda\sigma_w^2}}\right) \end{aligned} \tag{4.9}$$

where $\overline{|H|^2} = \frac{1}{N}\sum_{k=0}^{N-1}|H(k)|^2$ is the average channel gain. Using the Max-Log approximation [33], i.e.,

$$log\left\{\sum_i e^{\theta_i}\right\} \approx max_i\left\{\theta_i\right\} \tag{4.10}$$

Thus, the effective SINR can be rewritten as

$$\begin{aligned}\Gamma_{eff} &\approx -\lambda\left[log\left(\frac{1}{N}\right) - \frac{P_S\overline{|H|^2}}{\lambda\sigma_w^2} + max_k\left\{-\frac{\Delta_k P_S}{\lambda\sigma_w^2}\right\}\right]\\ &= -\lambda log\left(\frac{1}{N}\right) + \frac{P_S}{\sigma_w^2}\left(\overline{|H|^2} + min_k\left\{\Delta_k\right\}\right) \qquad (4.11)\end{aligned}$$

Recall that $\Gamma = \frac{P_S\sum_{k=0}^{N-1}|H(k)|^2}{N\sigma_w^2} = \overline{|H|^2}\frac{P_S}{\sigma_w^2}$, thus, Γ_{eff} can be rewritten as

$$\begin{aligned}\Gamma_{eff} &\approx -\lambda log\left(\frac{1}{N}\right) + \frac{\Gamma}{\overline{|H|^2}}\left(\overline{|H|^2} + min_k\left\{\Delta_k\right\}\right)\\ &= -\lambda log\left(\frac{1}{N}\right) + \Gamma\left(1 + \frac{min_K\left\{\Delta_k\right\}}{\overline{|H|^2}}\right)\\ &= -\lambda log\left(\frac{1}{N}\right) + \Gamma\left(\frac{min_k\left\{|H|^2\right\}}{\overline{|H|^2}}\right) \qquad (4.12)\end{aligned}$$

Therefore, we can conclude that the effective SINR only depends on the following parameters:

- λ : MCS dependent parameter
- N : number of subcarriers in an OFDM symbol
- Γ : SINR of the received OFDM symbol
- $\frac{min_k\left\{|H|^2\right\}}{\overline{|H|^2}}$: ratio of the channel gain of the worst subcarrier and the average channel gain of all subcarriers

Rewriting the effective SINR in 4.12 in dB scale gives

$$\left\{\Gamma_{eff} + \lambda log\left(\frac{1}{N}\right)\right\}_{dB} = \left\{\Gamma\right\}_{dB} + \left\{\frac{min_k\left\{|H|^2\right\}}{\overline{|H|^2}}\right\}_{dB} \qquad (4.13)$$

For a particular modulation scheme, $\lambda log\left(\frac{1}{N}\right)$ is constant and as a result, different channel conditions will cause a shift of the SER curve along the SINR axis, especially in the high SINR range. Based on the approximation that $PER \approx N_{symbol} \times SER$, it can be concluded that different channel conditions will result in an approximate shift in the PER curve along the SINR axis. Therefore, if one point of the PER curve for an instantaneous channel condition is found, the entire PER curve can be obtained by shifting the original curve to match that point.

4.1.1.3 Shifting of PER Curve Simulation and Measurement Results

In order to verify the result found in the previous section practical measurements were carried out using the Atheros WiFi chipset as well as a MATLAB simulation. In both the real world measurements and the MATLAB simulation, it is found that various channel conditions cause a shift of the PER curve for a given MCS. The result of the simulation is shown in Figure 4.1 where PER curves for an IEEE 802.11a communication system were simulated under different channel conditions. The channel conditions used include multipath channels with an exponential decay of 50ns and 100ns as well as a uniform channel with 12 taps and a delay of 600ns. As shown in Figure 4.1, it can be seen that in the normal operating range of the 802.11a communication system, i.e., PER values up to approximately 20%, the different channel conditions cause a shift in the PER-SINR relationship.

The measured PERs of the IEEE 802.11a communication system using different data rates under different channel scenarios is presented in Figure 4.2. In this figure the curves with markers are measured in an outdoor field next to the Thompson Engineering Building at UWO, while the curves with dashed lines are obtained from measurements performed indoors at the University Community Centre at UWO. As

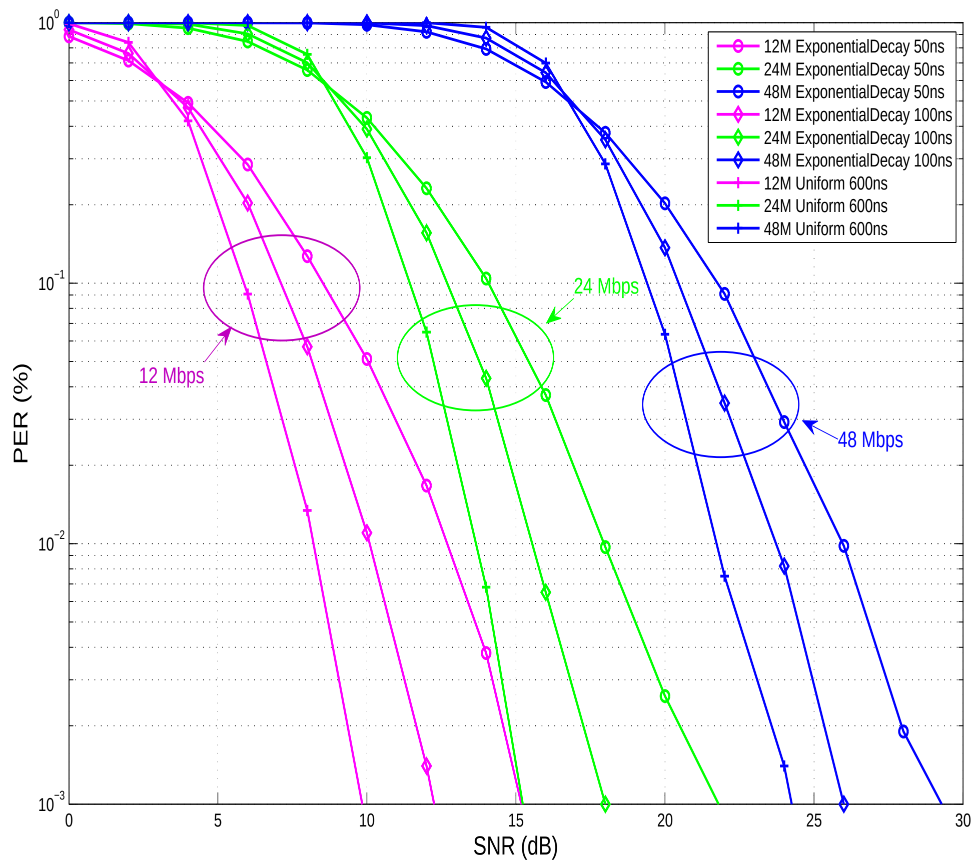

Figure 4.1: Simulated PERs under different channel scenarios for IEEE 802.11a.

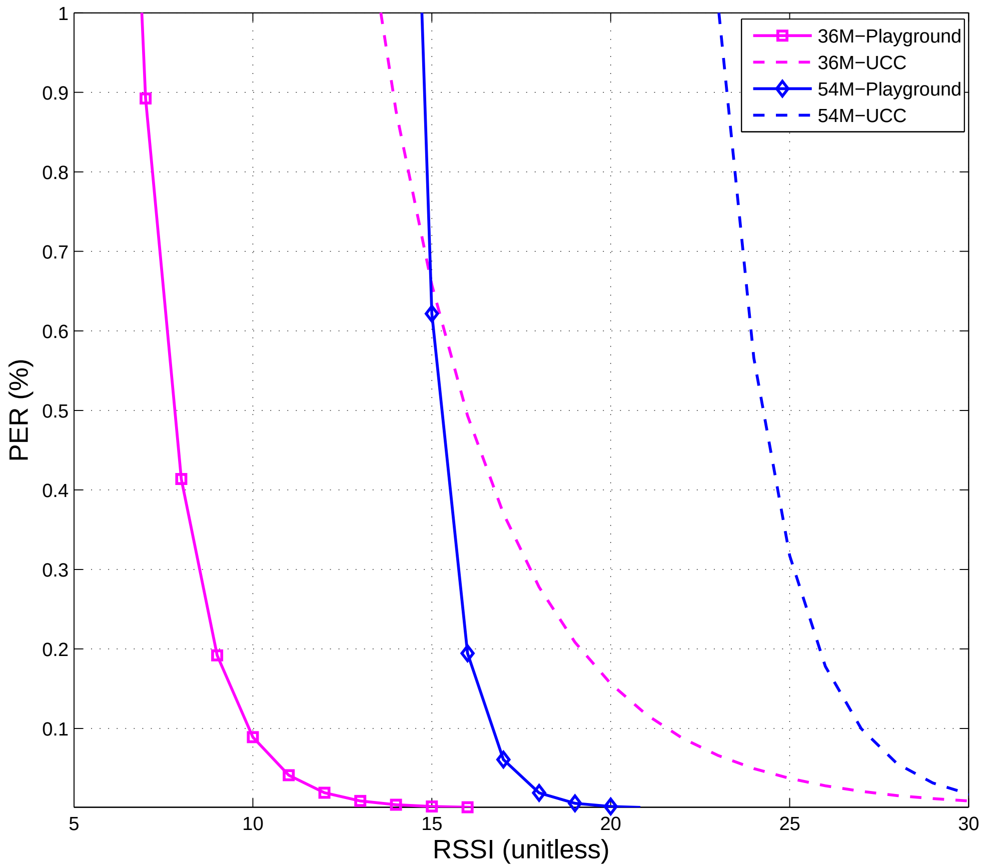

Figure 4.2: Measured PERs under different channel scenarios for IEEE 802.11a.

seen in the figure, different channel scenarios cause a linear shift of the PER curve along the SINR axis.

4.1.2 Window Size Selection for Average RSSI and PER Calculations

In order to calculate the average RSSI and PER, the number of samples to be used in the calculation must be determined. In the determination of the window size for PER and average RSSI estimation, the following points must be taken into account:

1. The latency for the calculations must not affect the short-term opportunistic gain over the wireless channel.

2. Samples used in the calculations must still provide relevant information about the current wireless conditions.

In order to gain a valid estimation of the average RSSI and PER under a given channel condition, the PER and average RSSI estimation must be updated each channel coherence time. As a result, the window size for the PER and average RSSI estimation can be based on the upper bound of the channel coherence time T_C and the average transmission time of each transmission attempt T_T, that is

$$W \leq \frac{T_C}{T_T} \tag{4.14}$$

The coherence time of a wireless channel can be calculated as [34]

$$T_C \approx \sqrt{\frac{9}{16 \cdot \pi \cdot f_m^2}} \approx \frac{0.423}{f_m} \tag{4.15}$$

where f_m is the maximum Doppler frequency, which can be calculated as $f_m = \frac{v}{c} f_c$. In the 802.11a OFDM PHY, $f_c = 5.2GHz$. In addition, during typical use of an 802.11 communication system, the most likely velocity v is walking speed or approximately 1 m/s. Using these values gives a channel coherence time of approximately $24.45ms$ or $T_C = 24.45ms$. The average time of each transmission attempt, T_T, must take into account two transmission scenarios, successful and failed transmission attempts, thus, $T_T = max\,(T_S, T_F)$. In this analysis the packet size is assumed to be $L = 1500B$ and there is no retransmission for the current packet. The average backoff interval after i

retransmission attempts can be calculated as

$$\overline{T_{backoff}}\left(i\right) = \frac{min\left[2^{i}\left(aCWmin+1\right)-1, aCWmax\right]}{2} \times tSlotTime \tag{4.16}$$

In order to evaluate the duration of each transmission attempt in IEEE 802.11, three situations need to be considered [35]: a) successful DATA and ACK transmission, b) successful DATA transmission with erroneous ACK reception and c) erroneous DATA transmission. For the case of the successful DATA and ACK transmission, the time duration T_{SDSA} for the i^{th} transmission attempt is

$$\begin{aligned} T_{SDSA}\left(i\right) = & \overline{T_{backoff}}\left(i-1\right) + T_{D}\left(L, R_{D}\right) + tSIFSTime \\ & + T_{ACK}\left(R_{ACK}\right) + tDIFSTime \\ & = 90\mu s + \frac{min\left[2^{i-1}\left(15+1\right)-1, 1023\right]}{2} \times 9\mu s + \lceil\frac{37.5+L}{Bps\left(R_{D}\right)}\rceil \times 4\mu \\ & + \lceil\frac{16.75}{Bps\left(R_{ACK}\right)}\rceil \times 4\mu s \end{aligned} \tag{4.17}$$

For the case of the successful DATA transmission with erroneous ACK reception, the time duration T_{SDEA} for the i^{th} transmission attempt is

$$\begin{aligned} T_{SDEA}\left(i\right) = & \overline{T_{backoff}}\left(i-1\right) + T_{D}\left(L, R_{D}\right) + tSIFSTime \\ & + T_{ACK}\left(R_{ACK}\right) + tEIFSTime \\ & = 150\mu s + \frac{min\left[2^{i-1}\left(15+1\right)-1, 1023\right]}{2} \times 9\mu s + \lceil\frac{37.5+L}{Bps\left(R_{D}\right)}\rceil \times 4\mu \\ & + \lceil\frac{16.75}{Bps\left(R_{ACK}\right)}\rceil \times 4\mu s \end{aligned} \tag{4.18}$$

However, since an ACK packet has a shorter packet length and is transmitted at a lower rate, an ACK packet transmission rarely fails and as such this situation happens

Data Rate	$T_{SDSA}\,(\mu s)$	$T_{EDNA}\,(\mu s)$
6 Mbps	2230.5	2200.5
9 Mbps	1547.5	1517.5
12 Mbps	1197.5	1175.5
18 Mbps	856.5	834.5
24 Mbps	676.5	663.5
36 Mbps	505.5	492.5
48 Mbps	420.5	407.5
54 Mbps	391.5	378.5

Table 4.2: Transmission time for each 802.11a data rate.

with a very low probability. Therefore, this situation is not considered in the analysis. For the case of the erroneous DATA transmission, the time duration T_{EDNA} for the i^th transmission attempt is

$$\begin{aligned} T_{EDNA}\,(i) &= \overline{T_{backoff}}\,(i-1) + T_D\,(L, R_D) + tACKTimeout \\ &= 65\mu s + \frac{min\left[2^{i-1}\,(15+1)-1, 1023\right]}{2} \times 9\mu s + \lceil \frac{37.5+L}{Bps\,(R_D)} \rceil \times 4\mu \\ &+ \lceil \frac{16.75}{Bps\,(R_{ACK})} \rceil \times 4\mu s \end{aligned} \tag{4.19}$$

Thus, T_{SDSA} and T_{EDNA} can be calculated for each data rate R_D for a packet length $L = 1500B$. The results are presented in Table 4.2. Therefore, the number of packets needed for the calculation of the PER and average RSSI for different data rates can be calculated using the transmission times found in 4.2. Table 4.3 shows the number samples calculated for each data rate. In order to balance the latency and the reliability of the PER and average RSSI calculation, the window size for the implementation is chosen as 32 for the PER calculation and 16 for the average RSSI calculation. It is felt that these values strikes a balance between maintaining up-to-date information and stability without impacting the algorithms adaptability.

Data Rate	N_{succ}	N_{fail}
6 Mbps	11	12
9 Mbps	16	17
12 Mbps	21	21
18 Mbps	29	30
24 Mbps	37	37
36 Mbps	49	50
48 Mbps	59	60
54 Mbps	63	65

Table 4.3: Number of successful and failed packet transmissions for each 802.11a data rate.

4.2 SARA Design

This section describes the overall design of the SARA algorithm in detail. SARA consists of two phases, 1) a calibration phase and 2) an operational phase. The calibration phase allows the algorithm to adjust to the current environmental conditions by shifting the PER-RSSI curves. As mentioned in Section 4.1, the impact of the environment causes a shifting of the PER-RSSI curves. Within the operational range of the 802.11 communication system, the shift is approximately linear and can be corrected for during the calibration phase. Following calibration, the algorithm moves into the operational phase which employs the RSSI and the shifted PER-RSSI curves to determine an initial data rate and then makes fine tune adjustments by using the PER.

4.2.1 Calibration Phase

The calibration phase consists of two parts, the maximum data rate search and the statistics collector. During the calibration phase SARA attempts to find the highest data rate that can be supported at the moment as well as to set the algorithm into the normal operating range of the 802.11 communication system. SARA accomplishes this by following a credit based procedure similar to ARF. The calibration phase begins by transmitting at a mid-level data rate, in our case at 36 Mbps. The algorithm then keeps track of the number of successful transmissions. If the calibration phase experiences ten consecutive successes, then the rate is increased. Alternatively, if the calibration phase experiences a single loss, then the rate is immediately reduced. The first part of the calibration phase is finished once a data rate increase has occurred following ten successes, but this newly chosen data rate fails during the next transmission, in which case the data rate is reduced by one and the data rate is returned to the previous successful data rate. The first part of the calibration phase helps SARA to find the highest supported data rate at the moment without staying at a data rate that is not successful for too long and also serves to place the algorithm into a suitable PER operating range. Once a data rate has been chosen, the algorithm then moves to the second part of the calibration phase. Using the selected data rate determined during the initial part of the calibration phase, the data rate remains unchanged for 32 packet transmissions in order to collect reliable PER information. Once the PER has been collected it is used to index into a table containing the typical RSSI value for that data rate and PER. This RSSI is obtained from many measurements performed in typical 802.11 environments such as an indoor scenario with reflections and movements of people and objects. This typical RSSI for the current PER, can now be compared to the current RSSI and calculate an RSSI offset. This RSSI offset is then used to shift the lookup table based on the amount the current RSSI differs from the typical scenario's RSSI. Figure 4.3 shows the shifting process graphically.

In Figure 4.3, the RSSI threshold between the data rates 24 Mbps and 36 Mbps is 21 (RSSI) in this example. However, it is found that the current PER for 24 Mbps is high enough to necessitate adding two RSSI units to the data rate table. After addition of the offset, the new RSSI threshold between 24 Mbps and 36 Mbps is 23

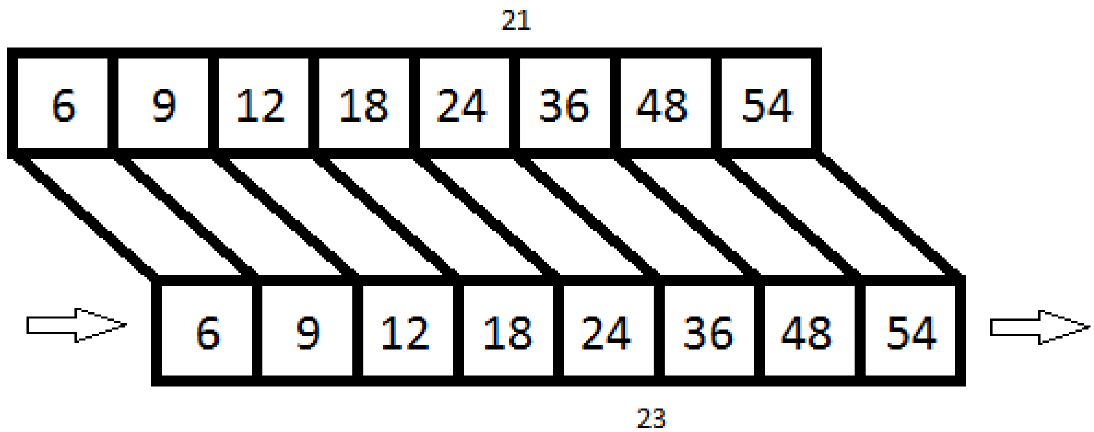

Figure 4.3: Shifting of the RSSI to data rate table with example RSSI values shown (21 and 23).

(RSSI) and it becomes more difficult to increase the data rate. As a general example, if a data rate is successful at a lower than typical RSSI, i.e., has an unusually low PER for the current RSSI, the algorithm calculates the difference between the current successful RSSI and the typical RSSI for that data rate. The RSSI difference is then used to shift the entire table to make it easier to transmit at that successful data rate by reducing the RSSI requirements for all data rates. Alternatively, if the RSSI is high but the PER is also high, then the algorithm again calculates the difference between the unsuccessful RSSI and the typical RSSI for that data rate. In this case however, the RSSI offset will cause the entire table to shift making it more difficult to transmit at that fairly unsuccessful data rate and increases the RSSI requirements for all data rates. Figure 4.4 presents the calibration phase flowchart.

After collecting 32 packets worth of PER data, the PER can be used as an index into a table obtained from Figure 4.5 to determine what the RSSI value should be. These PER curves are derived from measurements conducted in indoor environments. The data rate is fixed and then a data transfer is started to collect the RSSI information. After each test, the RSSI is reduced by either, moving the two STAs farther

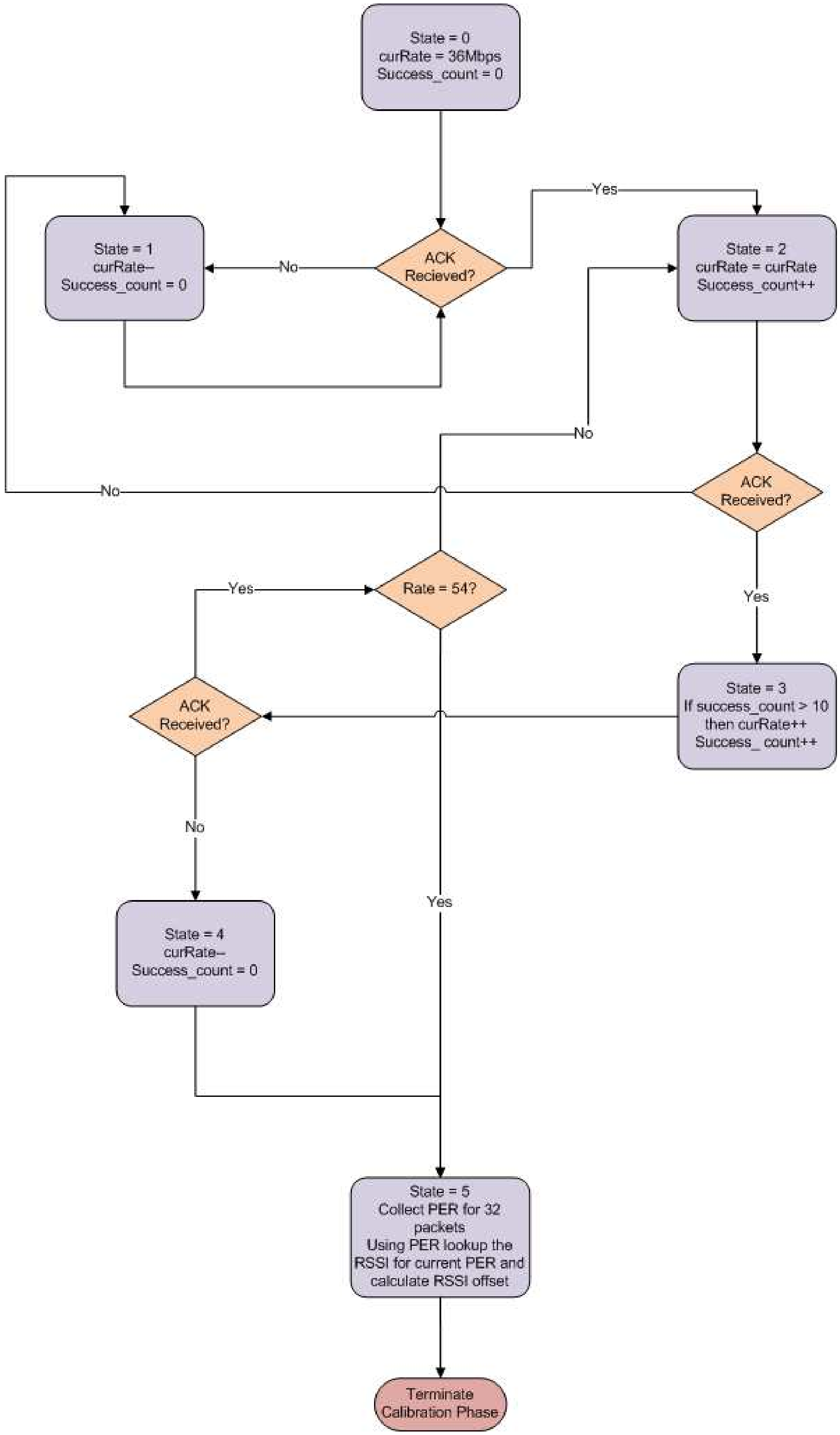

Figure 4.4: SARA calibration phase flowchart.

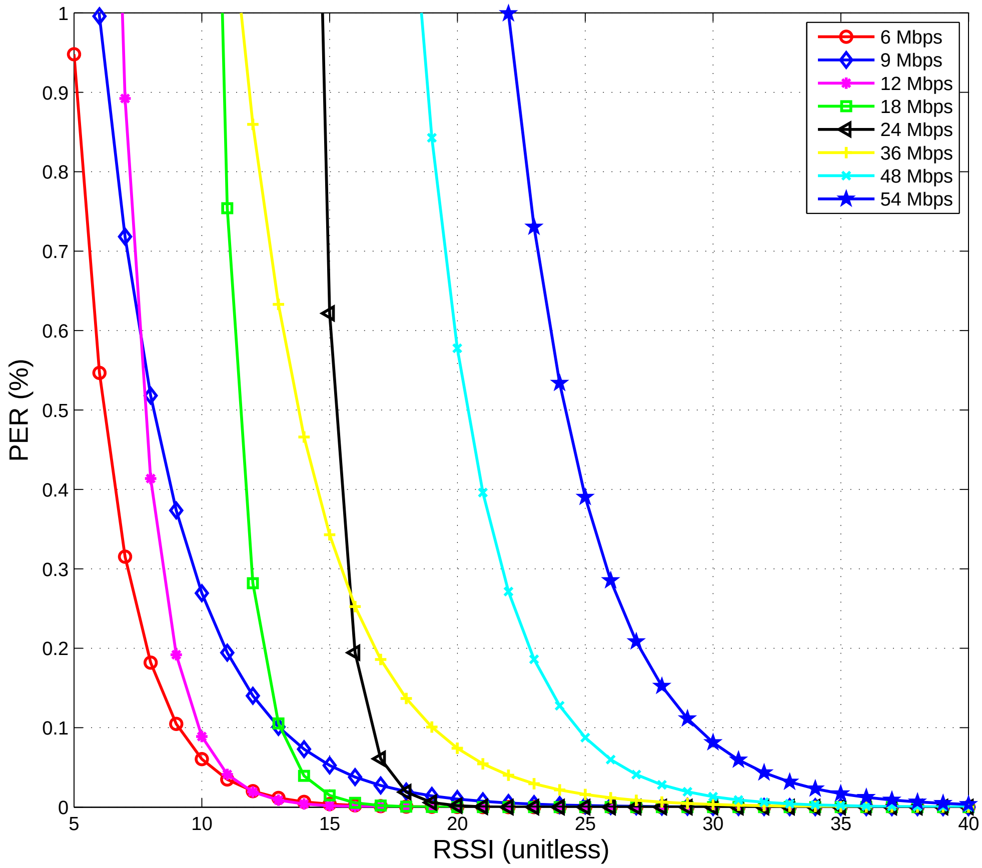

Figure 4.5: PER curves for each data rate over varying RSSI values.

apart or reducing the transmission power. After sufficient data has been collected, curve fitting is used to get continuous curves. From Figure 4.5 a table of RSSI values can be obtained for PER values for each rate in one percent increments.

The rationale behind this calibration phase is that the RSSI is a proprietary, unitless measure of signal strength and is loosely tied to the SNR, but in an unknown way. Thus, SARA must determine how the RSSI describes the current environmental conditions before further data rate selection can be accomplished, as will be discussed in the following section.

4.2.2 RSSI Selector and Environment Detector

In SARA, the RSSI is used for two purposes, as a method to select the data rate and also as a method to determine changes in the current environment. SARA calculates the average RSSI based on ACK frames from the receiving STA. The Atheros hardware calculates the RSSI using the preamble and header of every received frame [18], including ACK frames. Using the RSSI of the ACK frames the current channel conditions at the receiver can be determined based on a symmetric channel assumption [19]. In addition, since the ACK frame is mandatory in IEEE 802.11, the algorithm can gain an average indicator of signal strength very rapidly.

Once the average RSSI has been calculated, the value is used as an index into a lookup table. The Atheros hardware returns a value between 0 and 60, inclusive, which is then averaged. The average RSSI is rounded down due to the lack of floating support in the FreeBSD kernel and provides a lower bound on the RSSI values. Rounding the average RSSI down adds to the stability in rate selection and is also used as a direct index into the data rate table. The lookup table consists of rate indices for the RSSI values valid for each data rate. Table 4.4 shows the data rates corresponding to each RSSI value. This table was also obtained from measurements performed in the indoor environment. In order to obtain the rate indices, the throughput is plotted for each data rate over varying RSSI values. The intersection points of the curves represent the minimum and maximum RSSI values for the data rate curve. Figure 4.6 shows the throughput curves obtained from the measurements.

To illustrate the fact that the RSSI can be used to select an appropriate data rate for transmission, measurements were carried out for each data rate to see how the RSSI changes within a given environment and also how the corresponding PER for each data rate is affected by changes in the RSSI. Figures 4.7 and 4.8 show a

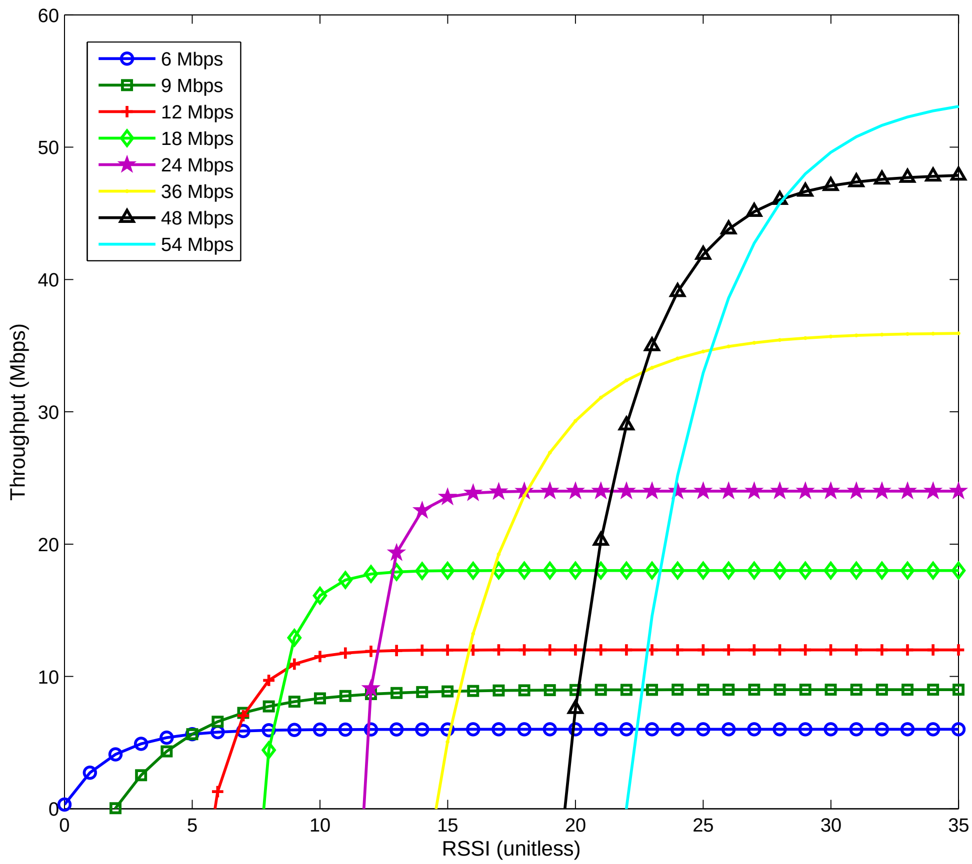

Figure 4.6: Measured throughput curves for each data rate over varying RSSI values.

Data Rate	Uncalibrated RSSI Value
6 Mbps	0-5
9 Mbps	6-7
12 Mbps	8-9
18 Mbps	10-13
24 Mbps	14-18
36 Mbps	19-23
48 Mbps	24-28
54 Mbps	29-60

Table 4.4: Data rates and corresponding uncalibrated RSSI values.

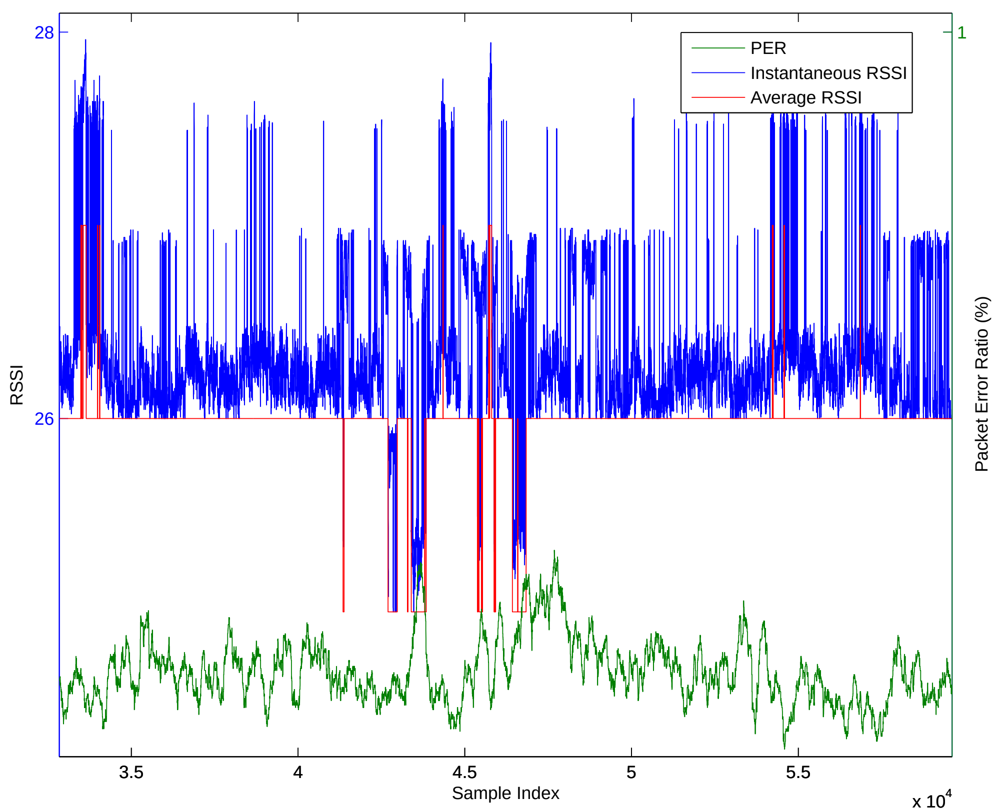

Figure 4.7: Instantaneous RSSI, average RSSI and PER trace for 54 Mbps in an indoor environment.

subset of a few of the measurements that were carried out. These measurements were performed in the University Community Centre, which is a high traffic and typical indoor environment with a large multipath component due to the structures found within. In addition, the movement of people and objects causes a Doppler effect on the received signal.

From Figures 4.7 and 4.8 it can be seen that the RSSI fluctuations affect the PER. In these traces one can see that periods of high RSSI roughly correspond to periods of low PER and periods of low RSSI roughly correspond to periods of high PER. This is most apparent in Figure 4.8, where there is a significant period of low

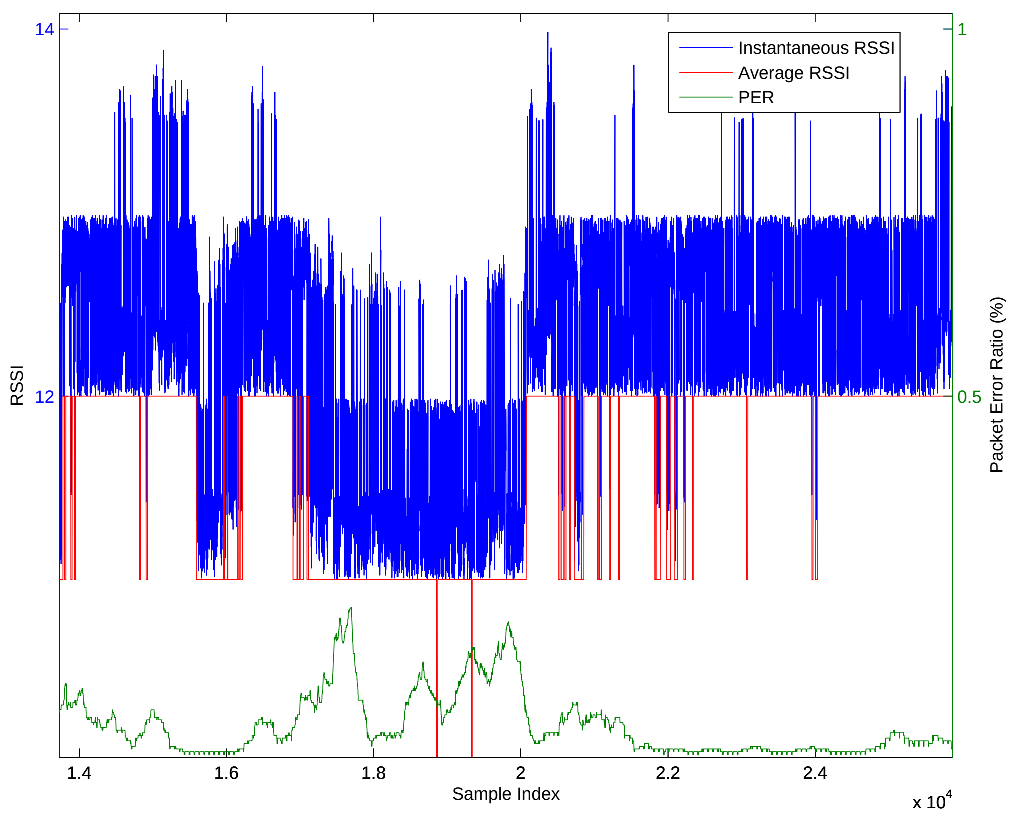

Figure 4.8: Instantaneous RSSI, average RSSI and PER trace for 9 Mbps in an indoor environment.

RSSI, and during this interval an increase in the PER can be seen which slowly decays as the RSSI returns to a higher level. Thus, the RSSI corresponds to some PER values regardless of the absolute value of the RSSI, hence, the use of the calibration phase to align the RSSI to an acceptable PER.

In addition to the RSSI selector, the RSSI is also used to determine major changes in the environment. The purpose of detecting major changes in the environment is to allow for recalibration of the algorithm to deal with the new environment conditions. Changes in the environment are detected by calculating the sample variance of the ACK RSSI which can then be used to detect major changes in environmental conditions. The environment may change if, for instance, the user moves to a new location or other factors change the environment surrounding the user, such as placement of objects or movement of people. Figure 4.9 shows the instantaneous RSSI along with the calculated variance of the RSSI.

As can be seen in Figure 4.9, when the RSSI experiences a major change, the RSSI variance shows a corresponding spike. Detecting when the RSSI variance reaches a given threshold allows the algorithm to determine when it should recalibrate given that a major change in the environment will cause a new PER/RSSI relationship.

4.2.3 PER Fine Tuning

Once SARA determines a suitable data rate to transmit at using the average RSSI, the algorithm ensures that the current data rate has been selected correctly by monitoring the PER. During the calibration phase, the RSSI and PER relationship is determined. However, if the calibration phase finds a relationship which is on the edge of a data rate transition, then, in this case, the algorithm may not know the exact optimum data rate to transmit at. In addition, conditions may vary slightly from those determined

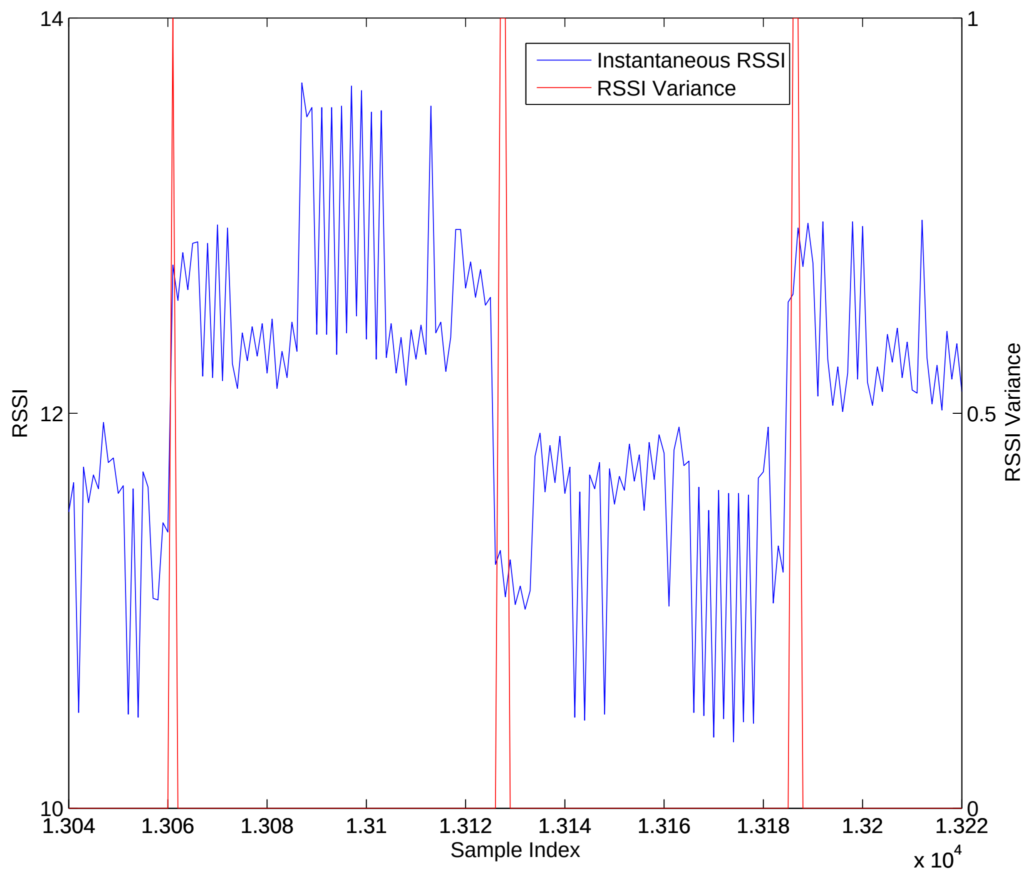

Figure 4.9: Instantaneous RSSI and RSSI variance trace for 9 Mbps in an indoor environment.

Data Rate	Success PER Threshold	Failure PER Threshold
6 Mbps	4%	100%
9 Mbps	27%	36.6%
12 Mbps	5%	27.5%
18 Mbps	1%	36.6%
24 Mbps	20%	27.5%
36 Mbps	3%	36.6%
48 Mbps	2%	27.5%
54 Mbps	0%	12.2%

Table 4.5: Success and failure PER thresholds for each data rate.

by the calibration phase but may not necessitate a complete recalibration. In order to protect the algorithm against these kinds of situations, the PER of the current data rate is monitored. If it is found that the PER of the current data rate is above a given failure threshold or below a given success threshold, the data rate is reduced or increased, respectively. The thresholds are determined from the same measurements used to determine the RSSI/PER relationship and the RSSI rate table. The thresholds obtained from measurements are shown in Table 4.5.

To illustrate how Table 4.5 is used, consider that the algorithm has currently selected a data rate of 36 Mbps for transmission. If the PER for 36 Mbps is less than 3%, then the algorithm realizes that during calibration the RSSI offset calculated may not be allowing the optimal data rate to be chosen, i.e., has underestimated the data rate. In this case, it may be possible to increase the data rate since the PER at 36 Mbps is so low, in other words, conditions may have improved following the calibration phase. At this point, the algorithm fine tunes the calibration by switching to the next higher data rate or 48 Mbps in this example. Alternatively, if the data rate is 36 Mbps, but the PER is greater than 36.6%, then the algorithm realizes that it may have overestimated the data rate and reduces the data rate by one or selects 24 Mbps in this example. In this way SARA can find the correct data rate for the current conditions, one where the data rate is experiencing an acceptable PER without being overly aggressive or without being overly conservative. In the case that the PER for the current data rate exceeds 50%, then the algorithm performs a fast recalibration by reducing the data rate by one and collecting 32 packets of PER information to realign the RSSI-PER table. The reason for this fast recalibration is that if the PER has grown large enough to reach 50%, then the table is misaligned and must be realigned. Figure 4.10 shows the flowchart displaying this process of fine tuning the data rate selection.

4.3 SARA Implementation

In this section, the specific details related to implementation on the Atheros device driver for FreeBSD is discussed. The Atheros device driver for FreeBSD is an open source device driver released under the BSD license and is based on the MadWifi device driver for Linux. FreeBSD is a UNIX like operating system providing a stable and customisable code base which is suitable for many application developments including communications research [36].

Programming is performed using the C programming language and compiled using the Make utility, which is a convenient method for building and installing kernel modules. The device driver can be built into the OS or can be compiled and installed separately as a kernel module. In order to build the device driver as a kernel module, a Makefile is used which indicates to the compiler and linker which source files to use and how to build the device driver. To build the driver, the 'make' command is issued and once the module is built it can be loaded by using the 'kldload' command.

Programming the device driver requires kernel level programming and as such, is very restrictive. At the kernel level, all compiler warnings are treated as errors, there

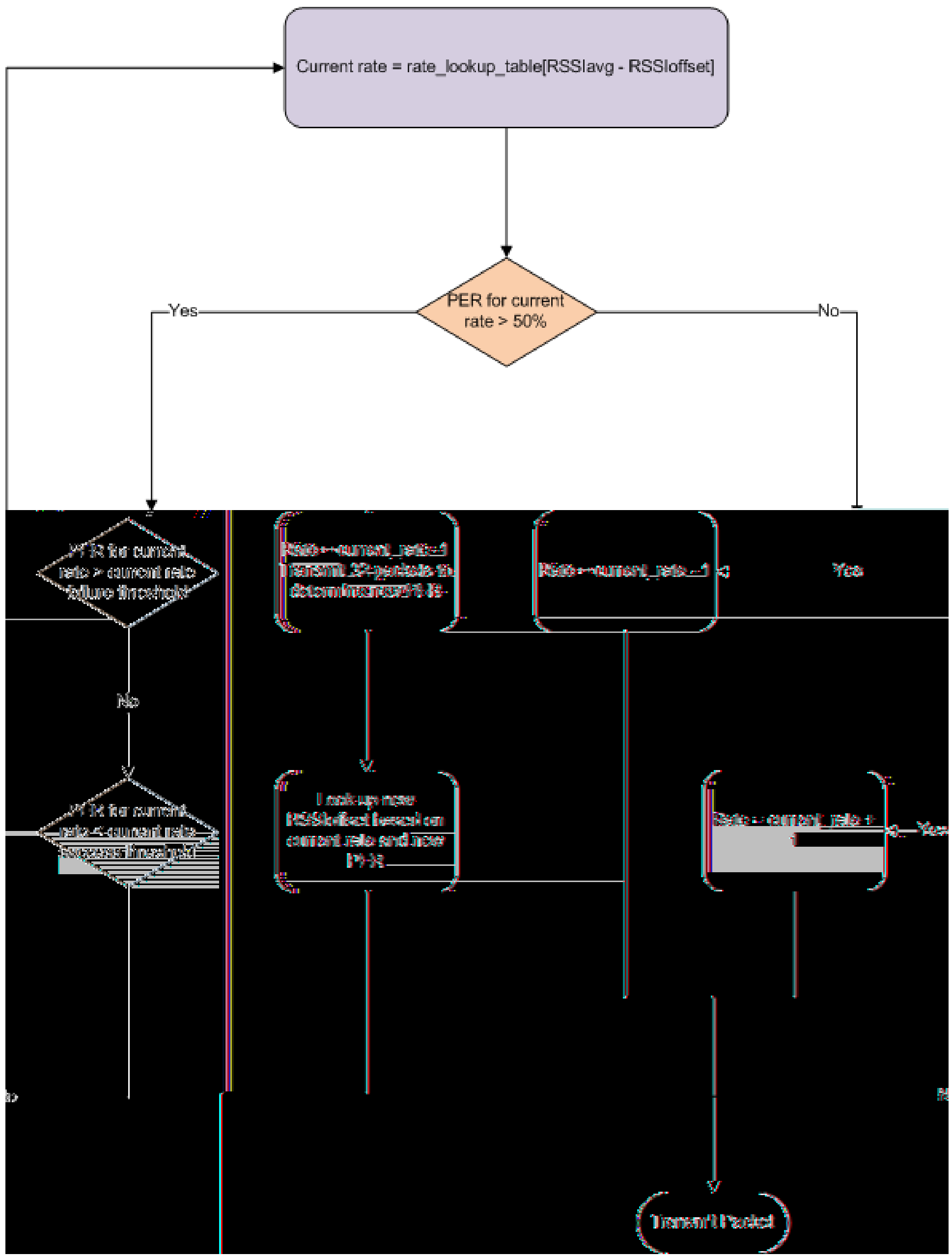

Figure 4.10: Flowchart of the data rate fine tuning procedure using the PER.

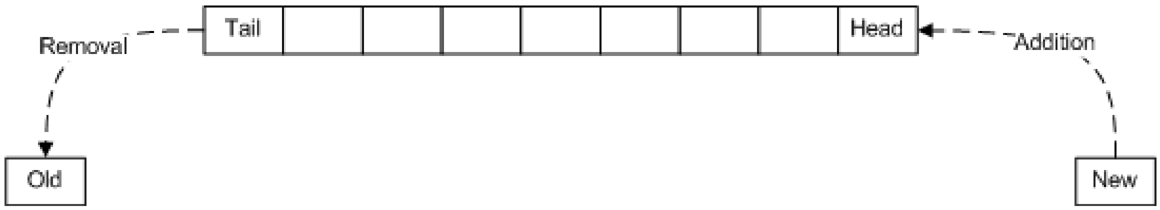

Figure 4.11: Insertion and removal of values from a singly-linked tail queue.

are no standard library routines and no graphical debugger is available. Furthermore, at the kernel level there is also no floating point or fixed point support, only integer programming is possible. As a result, all calculations must be performed with the knowledge that no fractional values are available and that all data structures used must be those available from within the kernel.

The FreeBSD kernel provides a handful of data structures for use by kernel programmers such as lists, queues and hash tables. For the purposes of the RAA, the average RSSI, RSSI variance and PER were all stored and calculated using a singly-linked tail queue. A singly-linked tail queue can be used to implement the required calculations since items are added at the head of the queue but can also be removed from the tail of the queue. In this way, the newest values are added at the head and the oldest are removed from the tail. In addition, the FreeBSD man pages state that the singly-linked tail queues can offer forward traversal through the list and are ideal for implementing a FIFO queue as required in the SARA implementation. Figure 4.11 illustrates the process of inserting and removing a value from the singly-linked tail queue.

The lack of support for fractional values in the kernel means that all calculations must be performed as integer data types. In the case of calculating the PER, a workaround was used which required shifting the data by the amount of precision required. In the case of SARA, a value of 100000 was used to shift the data, giving us three decimal points of precision, which is enough for our needs since the algorithm is only concerned with a PER percentage. For calculating the RSSI average and variance no shifting is required since SARA is not interested in the exact value of the RSSI but is more interested in the relationship between RSSI and PER. Furthermore, using the integer value of the average RSSI allows the algorithm to use it as a lookup value or as an index into a table. This usage reduces computational load and allows a quick table lookup for data rate selection.

In order to implement a new algorithm for the Atheros driver, one has to follow the interface defined in the file `if_athrate.h`. This interface defines several methods that must be implemented in order to be defined as a rate control algorithm. The Atheros device driver requires that the rate control algorithm be attached and detached to the driver using the `ath_rate_attach` and `ath_rate_detach` methods which allocate and free memory for the algorithm, respectively. Additionally, the rate control algorithm must implement `ath_rate_node_init` and `ath_rate_node_cleanup` to initialize and cleanup any memory and data related to a structure called `ath_node` which is used to encapsulate data required for the rate control algorithm. Additionally, when a STA associates or re-associates with an AP the following method is automatically called, `ath_rate_newassoc`, which updates the state of the rate control algorithm in preparation for upcoming data transmission. Transmission handling is accomplished with the methods `ath_rate_findrate`, `ath_rate_setupxtxdesc` and `ath_rate_tx_complete`. The `ath_rate_findrate` method returns the transmit info for a data packet to be used in the following data transmission. If multirate retry is supported, then the method `ath_rate_setupxtxdesc` is called to determine the number of tries and rates for the following retransmissions. Finally, `ath_rate_tx_complete` is called following transmission, whether successful or not, and is used to update the state of the rate control algorithm.

In addition to the required interface methods, SARA implements several custom helper methods to help manage the data structures, perform calculations and access the data rate tables. The following is a brief description of the most important methods used by SARA.

- `collectPERForCurrentTxRate` - This method is called following the transmission of a packet and keeps track of the total number of transmissions as well as the number of failed transmissions. Using this information, the PER for each data rate is collected and stored in the singly linked tail queue structure.

- `collectRSSIStats` - This method is called following the transmission of a packet and is used to calculate the average RSSI as well as the variance of the RSSI values.

- `calibrationTxComplete` - This method is called following the transmission of a packet and is only used during the calibration phase of the algorithm. This method counts the success and failures of packet transmission and also modifies the state of the calibration phase.

- `getPERIndexForSelectedRate` - This method is called from the method `calibrationTxComplete` during the last state of the calibration phase to calculate the RSSI given the measured PER.

- `select_rateidx_based_on_rssi` - This method is called from `ath_rate_ctl` prior to the transmission of a packet. The method converts the current average RSSI into an rate index for use in the upcoming transmission.

- `select_rateidx_for_calibration` - This method is also called from `ath_rate_ctl` and is used to select a rate index during the calibration phase. The method uses the state of the calibration phase to determine whether to increase, decrease or remain at the same data rate.

4.4 Summary

This chapter described the design and implementation of the Situation-Aware Rate Adaptation algorithm (SARA). The flowchart in Figure 4.12 shows the overall design of SARA. SARA consists of three main components, the calibration phase, the RSSI selector and environment detector and the PER fine tuning. The calibration phase consists of two parts, the maximum rate detector and the PER collector. The maximum rate detector finds the highest data rate currently supported by following a success and failure scheme. The data rate is increased following ten consecutive successes and decreased following a single failure. This portion of the calibration phase quickly finds a suitable data rate to begin transmitting at for the second portion of the calibration phase. The second portion of the calibration phase collects PER information and aligns the data rate table based on the difference between the current average RSSI and the RSSI obtained from a PER/RSSI table. Using this RSSI offset, RSSI requirements for choosing a given data rate can be made more difficult or more easier. Following the calibration phase, the operational phase begins, which consists of the RSSI selector and environment detector as well as the PER fine tuning. The algorithm uses the average RSSI and the RSSI offset calculated during the calibration phase to select a data rate for transmission. In addition, the algorithm detects major changes in the environment by monitoring the variance of the RSSI. If a large variance is detected, the algorithm recalibrates to the new environmental conditions. As packets are transmitted, the algorithm continues to keep track of the PER and if it is seen that the PER is either too high or too low for a given data rate, the algorithm decreases or increases the data rate accordingly. When the data rate is changed due to the PER exceeding a given threshold, the algorithm recalculates a new RSSI offset using this new PER value. The algorithm is environment aware by using a calibration

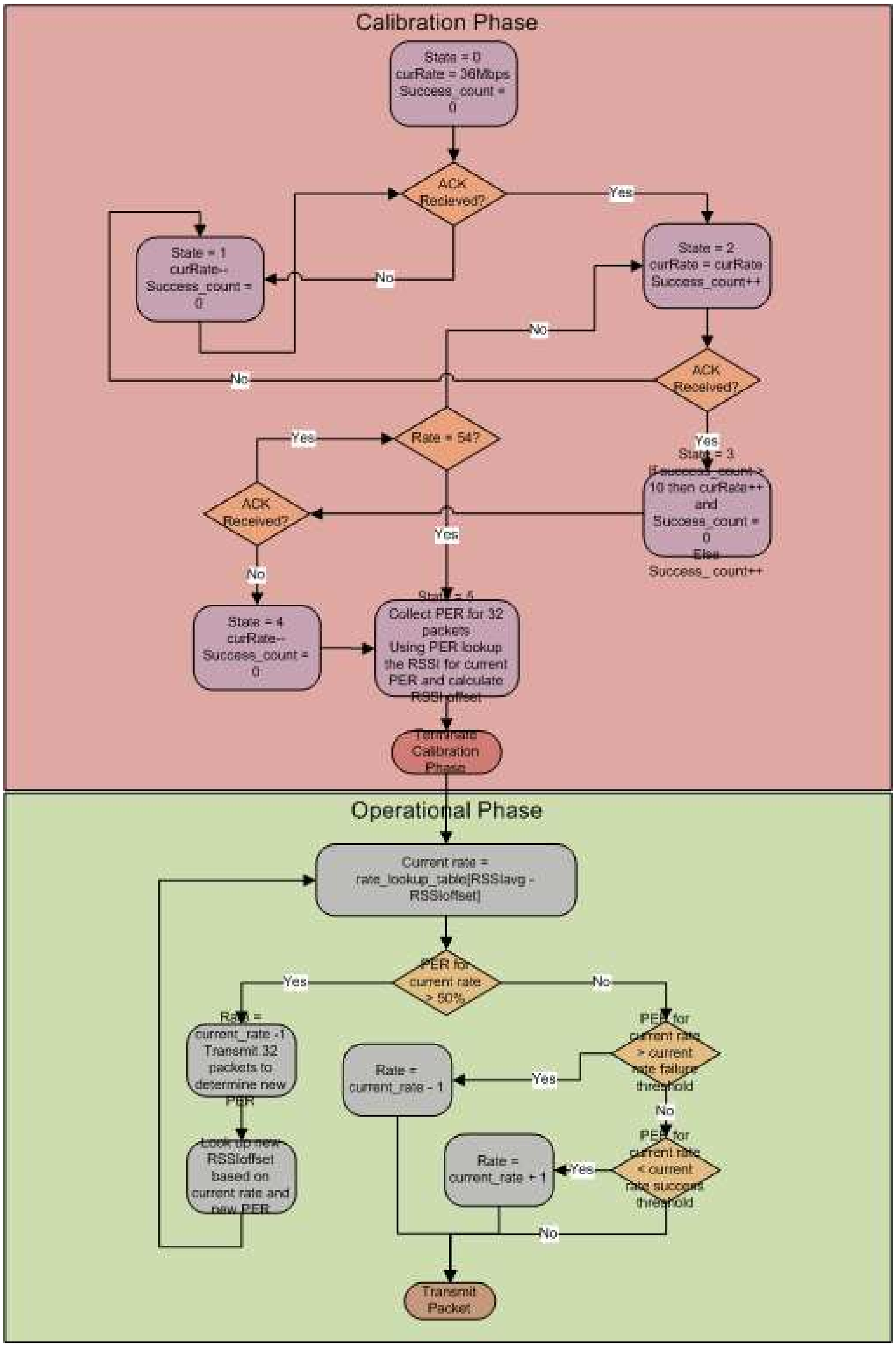

Figure 4.12: Overall design of SARA, including the calibration and operational phases.

phase, can adapt to rapid changes by using the average RSSI and measuring the RSSI variance and guarantees throughput by monitoring the PER conditions. Following the design, SARA was implemented in an Atheros device driver for FreeBSD using the C programming language.

Chapter 5

RAA Performance Evaluation

This chapter evaluates the performance of SARA and compares it to the SampleRate algorithm, which is considered the benchmark for RAAs. Extensive measurements were carried out in various scenarios such as static, dynamic and congested scenarios. This chapter starts with the static scenario results and continues with the dynamic scenario and finally discusses the congested scenario. In each scenario the test setup is described and the results presented.

5.1 Static Scenario

Evaluation of the RAAs in an indoor scenario is presented in this section. The test is performed inside the University Community Centre (UCC) at UWO on the top floor where there is a LOS and a minimal number of people walking around. This is an almost typical scenario in that it is an indoor scenario with a large multipath component, however, this would represent a better than average channel scenario since there are very few people moving around and no obstacles between the STAs. Figure 5.1 shows the locations of the two devices used in the test. At location one, the STA sends UDP traffic to location two using the jPerf application.

In order to evaluate the performance of the algorithms, the software program jPerf was used to both generate traffic and measure throughput. jPerf is a simple application which can be used to measure throughput and latency and is especially

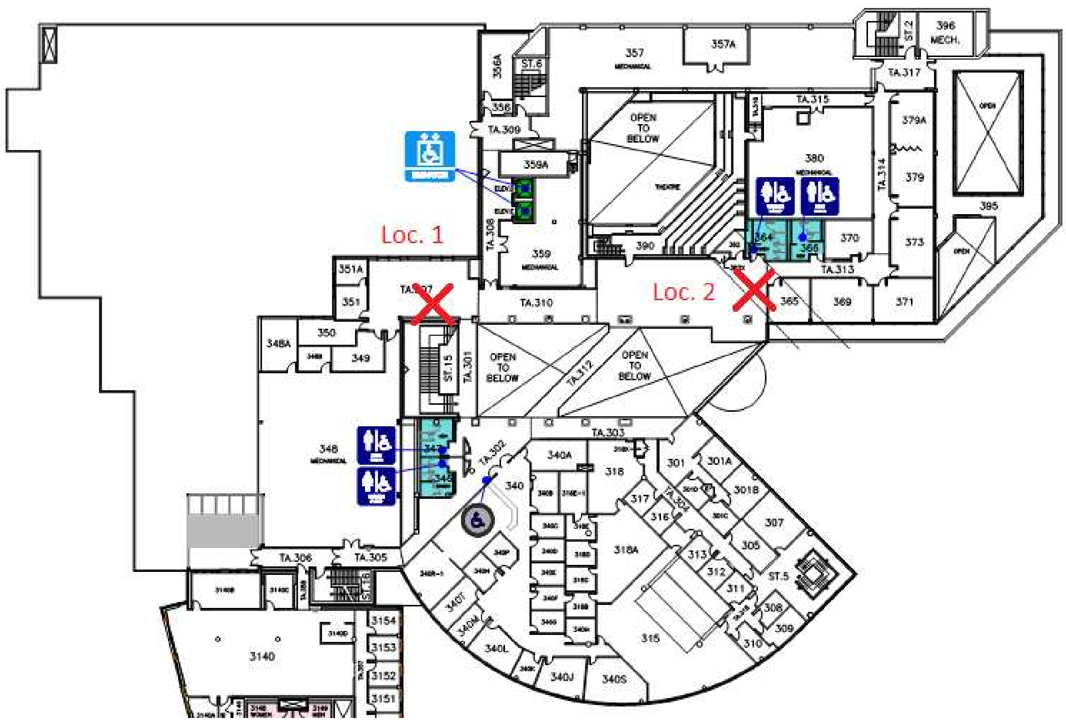

Figure 5.1: Map of the top floor UCC showing locations of test devices used during the RAAs evaluation in the static scenario.

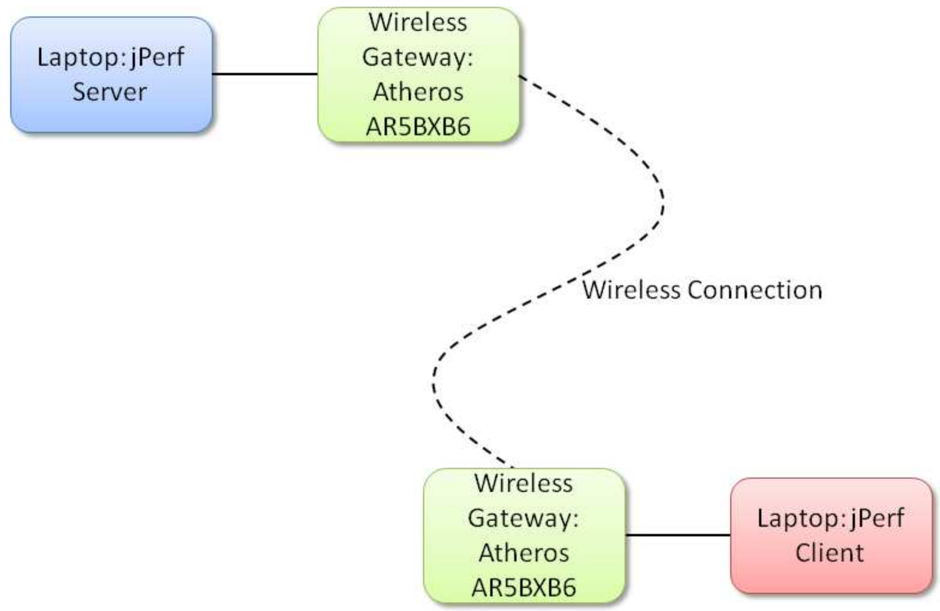

Figure 5.2: Connection diagram used during the RAAs evaluation.

useful in measuring the impact of performance optimizations. jPerf can report the average throughput, the amount of data delivered and the jitter experienced. A jPerf client is located at location one and a jPerf server is located at location two as shown in Figure 5.1. The jPerf application is executed on laptops connected to wireless gateways containing Atheros AR5BXB6 802.11 WiFi chipsets as shown in the connection diagram in Figure 5.2.

Once the connection has been established, the data transfer is started using jPerf. Three tests are carried out for 120 seconds using UDP with a data transfer of 50 Mbps for both SARA and SampleRate with all three results averaged. Figure 5.3 shows the results of the tests. As can be seen in Figure 5.3 SARA and SampleRate

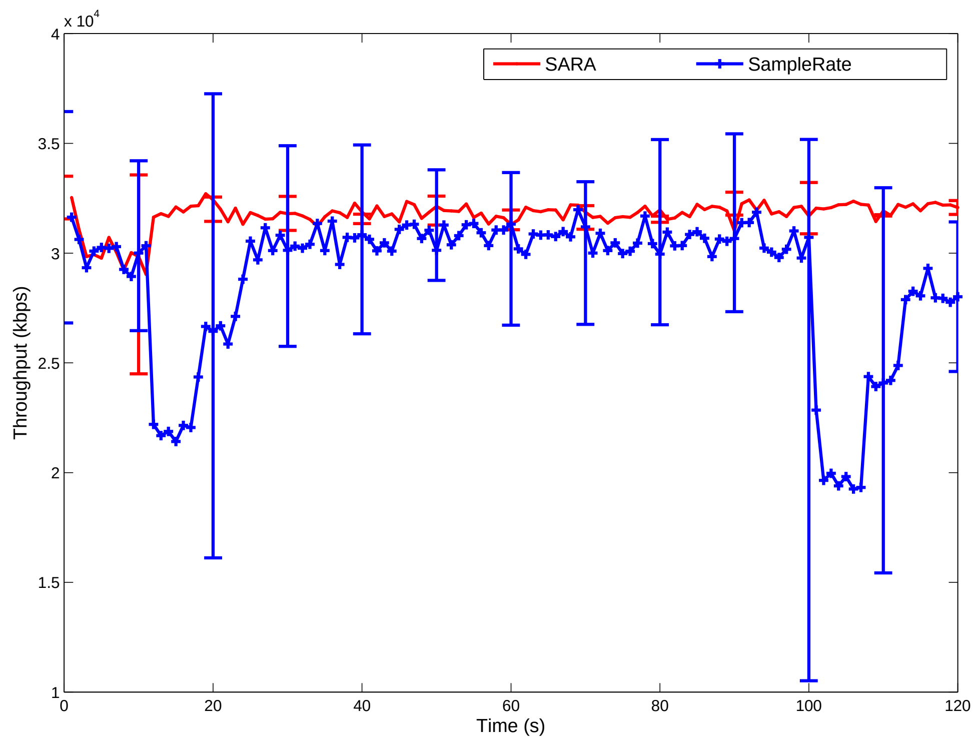

Figure 5.3: SARA versus SampleRate on the top floor of the UCC.

show similar performance with SARA showing slightly improved throughput. In addition, the standard deviation has been plotted and it is clear that SARA has much greater stability than SampleRate except for during the calibration phase at the beginning of the test. SARA is showing greater stability since the channel is relatively stable in this case. Figure 5.4 shows the average throughput comparison between SARA and SampleRate as well as with the samples taken during the calibration phase removed. There is a small performance reduction in SARA which is due to the calibration phase since some time is spent transmitting at data rates less than optimal in order to adapt to the current environment, however this impact is minimal.

In Figure 5.5 the jitter experienced by SARA and SampleRate during the test is shown and with jitter the lower the value the better. Jitter is important since it

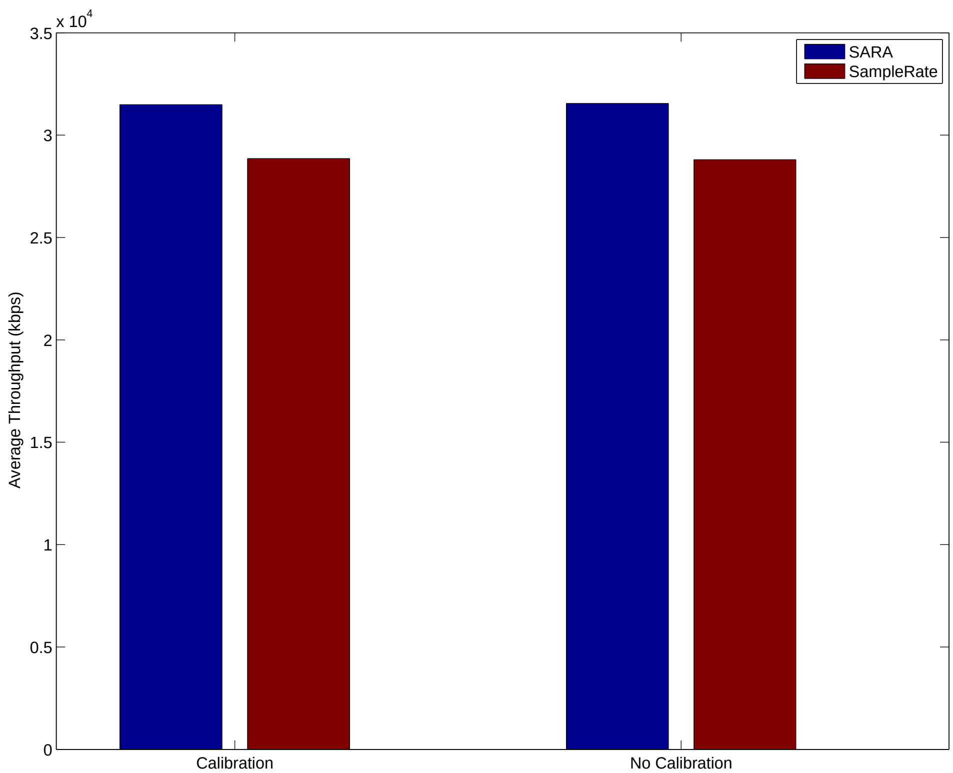

Figure 5.4: Average throughput between SARA and SampleRate on the top floor of the UCC.

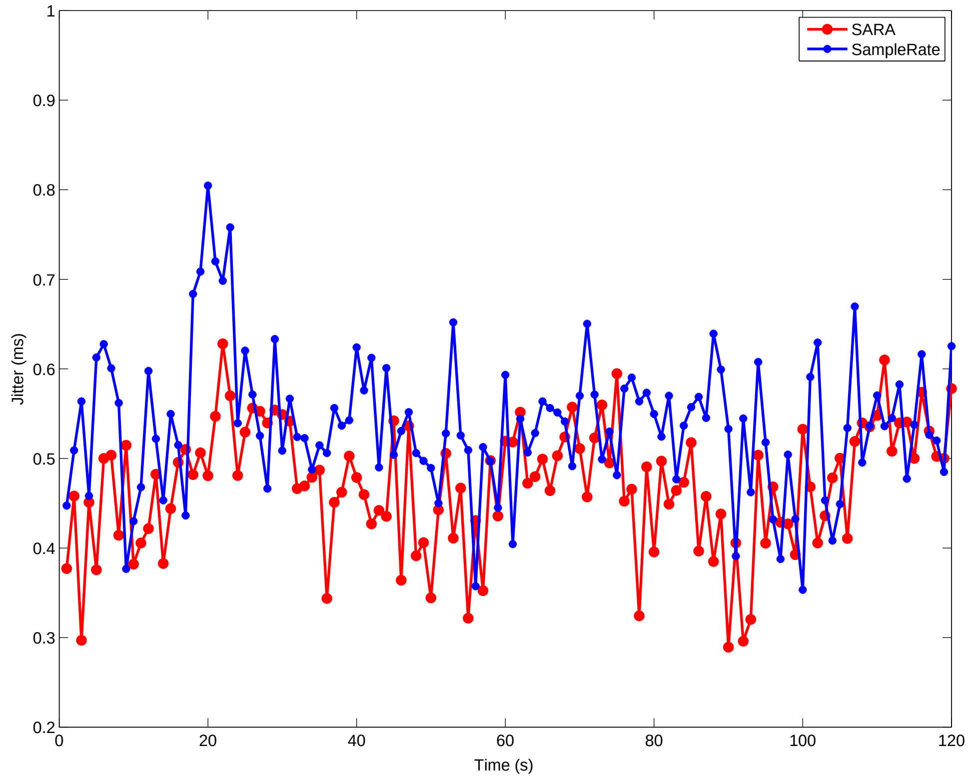

Figure 5.5: Jitter comparison between SARA and SampleRate on the top floor of the UCC.

represents the latency variation associated with the transmission of the UDP data packet and affects streaming data such as video or voice. A lower jitter means less delay or latency in the delivery of data packets, thus real time streaming applications can obtain smoother performance with lower jitter. Figure 5.6 shows the average jitter experienced by SARA and SampleRate with both calibration samples included and not included. As can be seen in the figure SARA displays a significantly lower jitter.

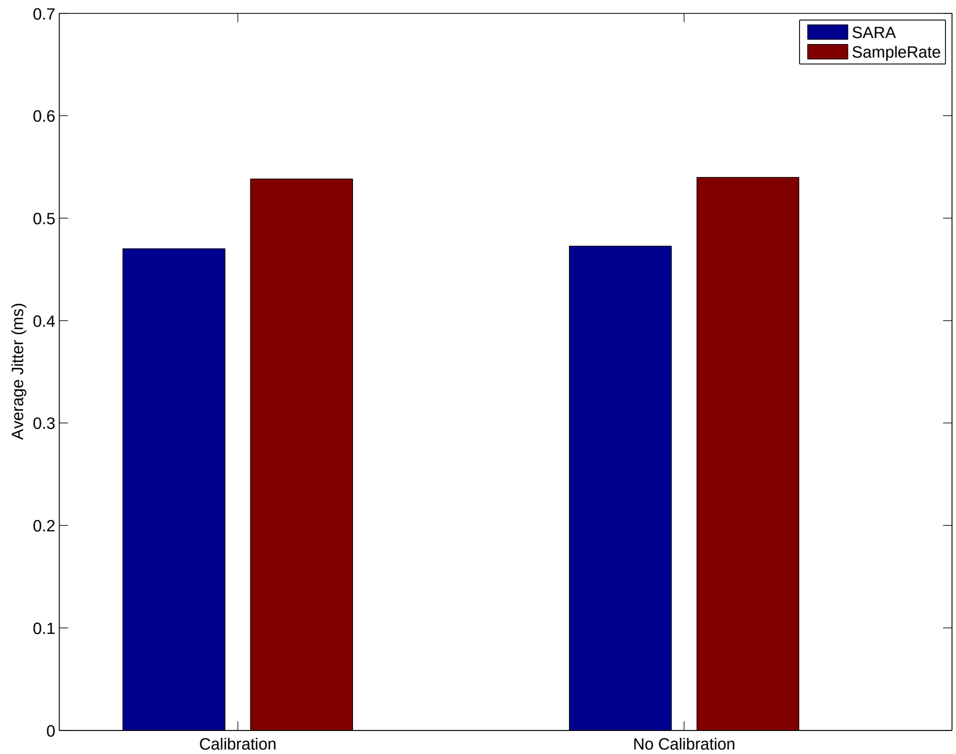

Figure 5.6: Average jitter comparison between SARA and SampleRate on the top floor of the UCC. (A lower jitter is better.)

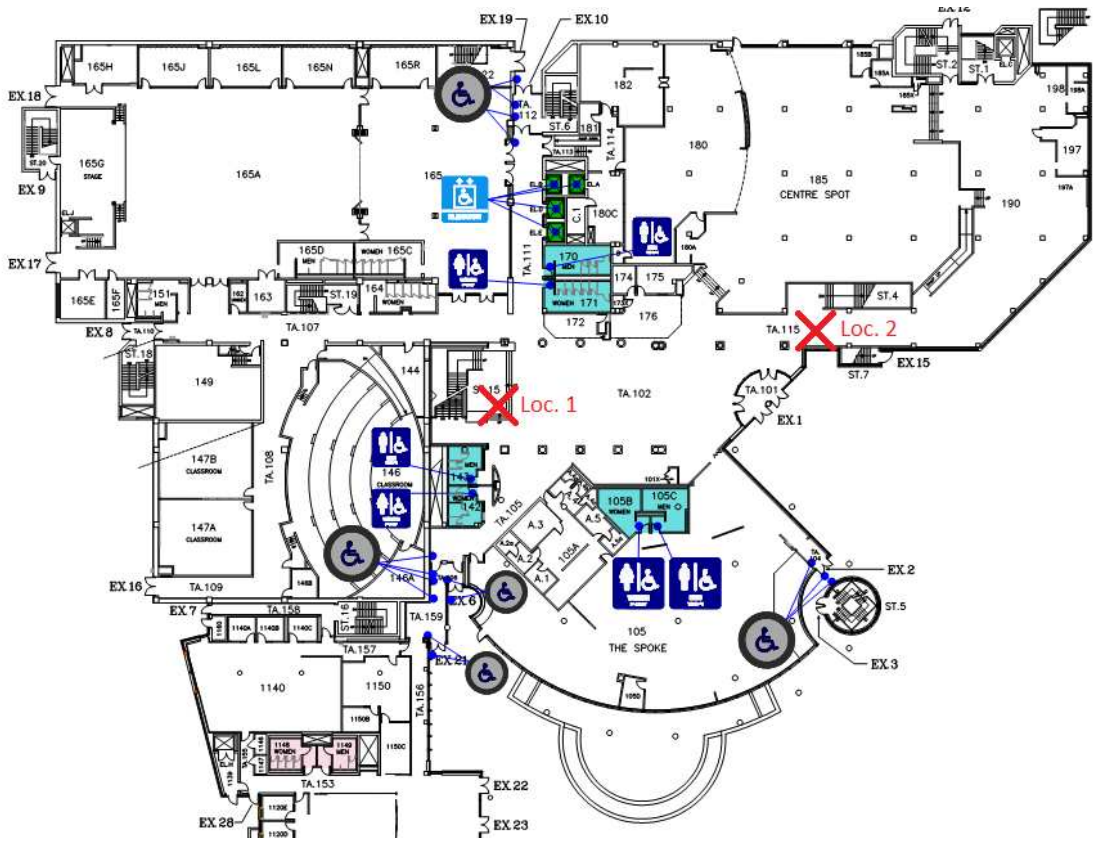

Figure 5.7: Map of the main floor UCC showing locations of test devices used during the RAAs evaluation in the dynamic scenario.

5.2 Dynamic Scenario

To evaluate the RAAs in a typical indoor scenario, test devices were set up at locations inside the UCC at UWO. Figure 5.7 shows a map of the UCC as well as the locations of the test devices. The UCC provides a good testing environment since it is an indoor location with a large multipath component as well as a large number of people and movement, including other unknown wireless devices which may cause interference. This scenario can be considered the typical usage scenario for an 802.11 device similar to a library or coffee shop. Again, two STAs are placed at location one and two and a jPerf data transfer is started for two minutes with three tests for each algorithm. Once the tests have completed the results are averaged and plotted.

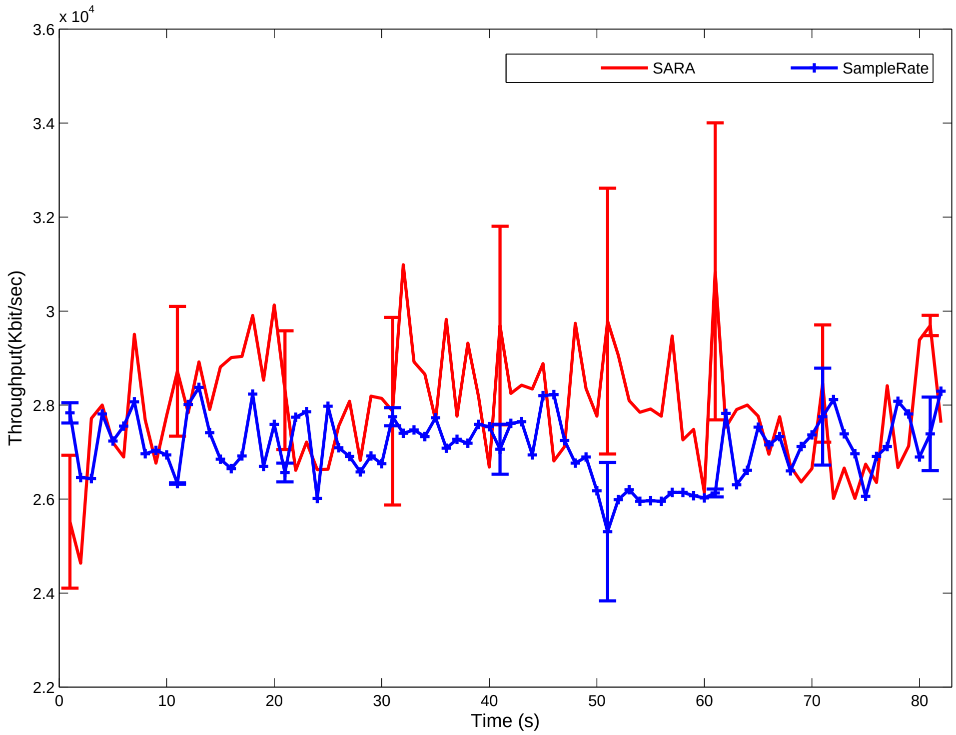

Figure 5.8: SARA versus SampleRate on the main floor of the UCC. In this test there are many people moving around between the two test devices.

Figure 5.8 shows the throughput results of the test. In this test, SARA shows improvement over SampleRate, but a performance hit is seen during the calibration phase which is evident by the low throughput points on the left side of the figure. In this case, SARA shows larger variation in throughput than SampleRate which indicates that SARA is following the variations of the channel conditions. Figure 5.9 shows the performance gain in terms of the average throughput of SARA over SampleRate, comparing both with and without calibration phase. Since the calibration phase happens only upon association or if the environment changes significantly, the effect of the performance loss due to calibration is reduced during long periods of usage.

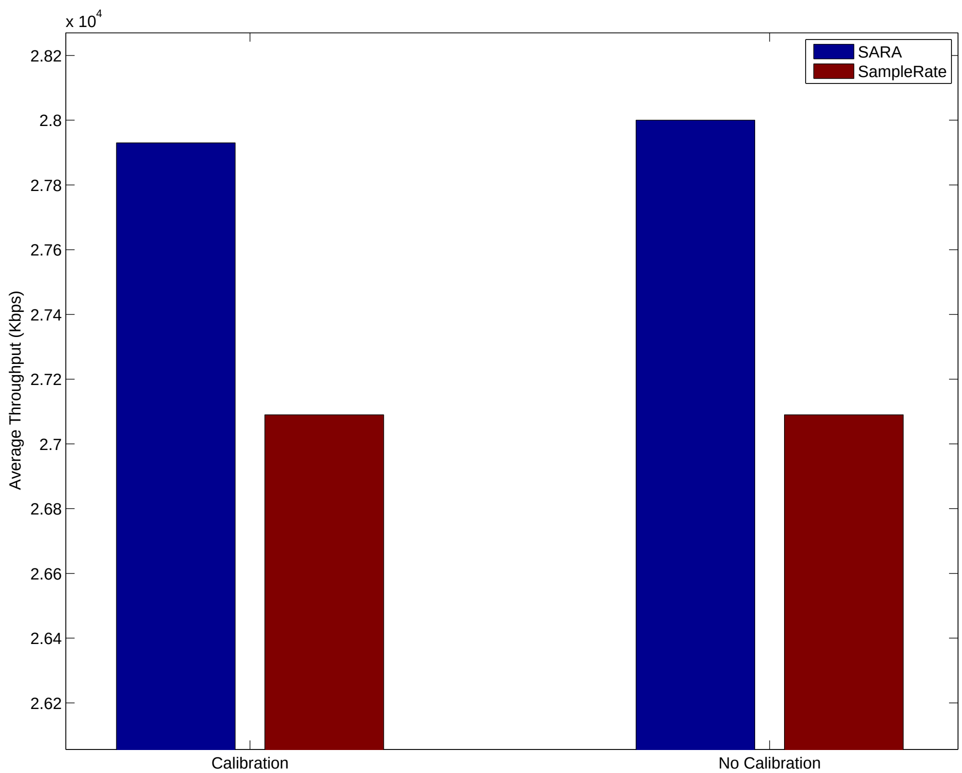

Figure 5.9: SARA versus SampleRate on the main floor of the UCC. The average throughput gain with and without calibration is displayed.

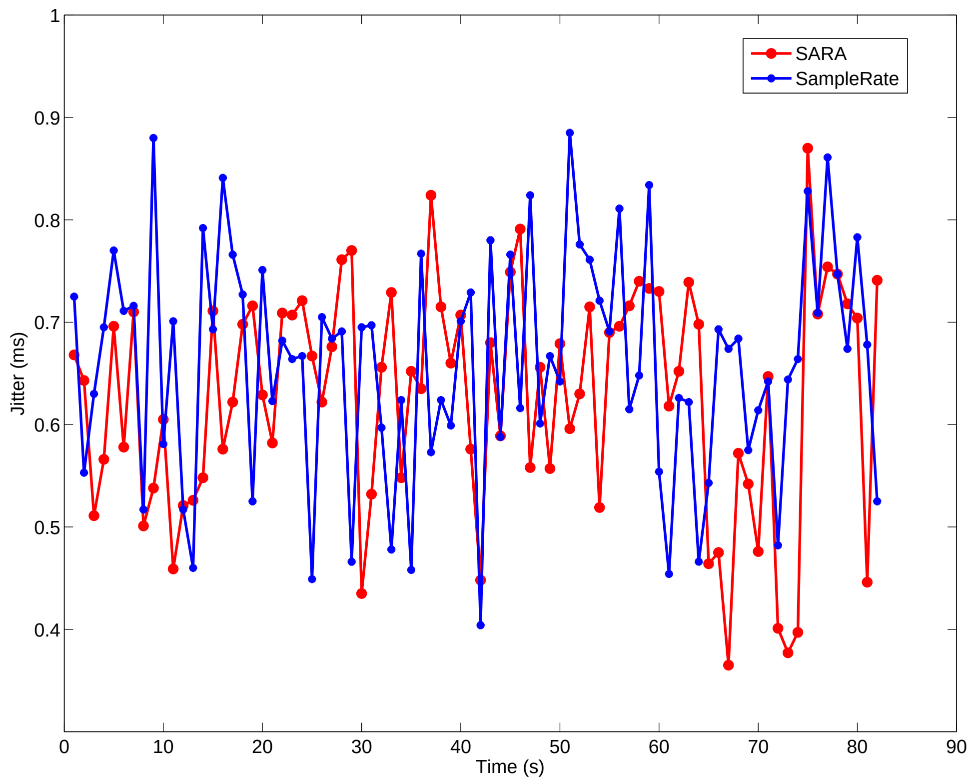

Figure 5.10: Jitter comparison between SARA and SampleRate on the main floor of the UCC.

In addition, the jitter is plotted and SARA and SampleRate are compared in Figure 5.10. Figure 5.11 shows the performance gain of SARA over SampleRate in terms of the average jitter experienced. As can be seen, the jitter is lower than that experienced by SampleRate.

5.3 Congested Scenario

In this section SARA is compared against SampleRate in a congested scenario. The experiment was performed in the Primus Canada Telecommunications Laboratory in the Thompson Engineering Building at UWO. In this experiment, three STAs were

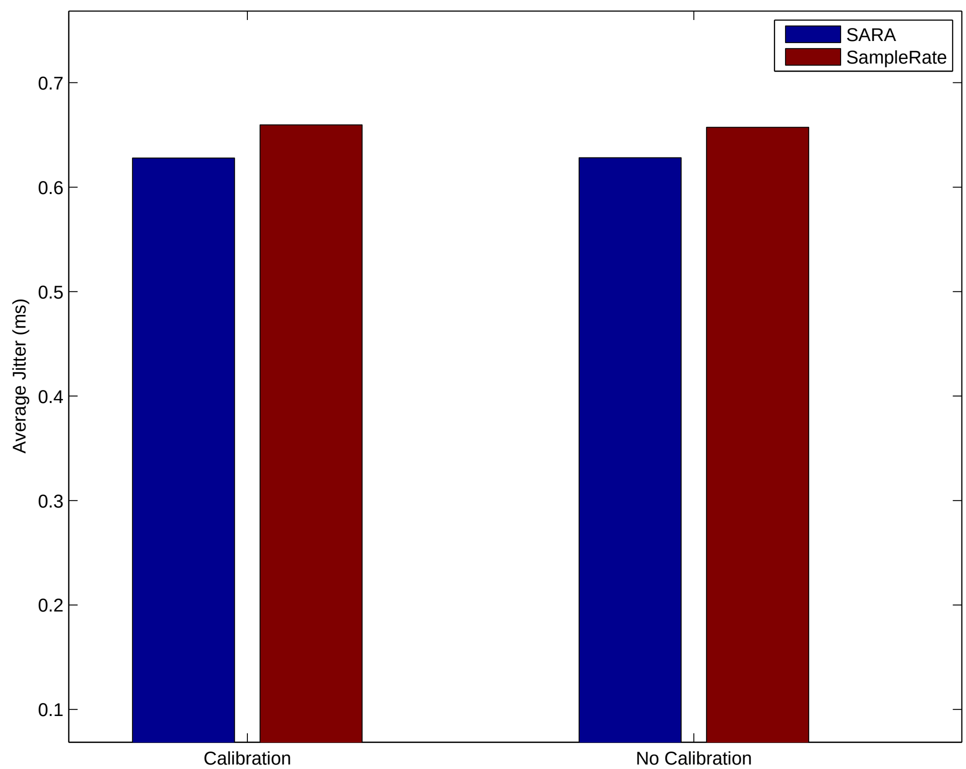

Figure 5.11: Average jitter performance gain comparison between SARA and SampleRate on the main floor of the UCC. (A lower jitter is better.)

associated with a central AP. All three STAs employed the use of jPerf to transmit UDP packets with a bandwidth of 50 Mbps. The test was run three times for SARA and three times for SampleRate with the results averaged. During the test, all STAs are transmitting data to create a congested scenario with collisions occurring amongst the STAs. Figure 5.12 shows the connection diagram for this test. All three STAs and the AP are spread throughout the lab with about two metres separation. In an uncongested scenario, this situation would be very good and throughput would typically be high, however, in the congested scenario loss can be very high due to collisions. However, most algorithms reduce the data rate because they attribute the high loss to channel conditions. In this scenario, it is better to maintain the data rate, rather than reduce it, since a reduction in data rate translates into a longer transmission time, thus increasing the probability of collision. SARA mitigates this effect by using the average RSSI to guide rate selection.

The following figure shows the result of the congestion scenario test. As can be seen in Figure 5.13, SARA shows considerable performance gain in congested scenarios when compared to SampleRate. Figure 5.14 shows the average throughput gain of SARA compared to SampleRate for the congested scenario.

Figures 5.15 and 5.16 show the results of the jitter measurements taken during the congestion scenario. In the congested scenario, SARA performs better than SampleRate with a lower average jitter. As mentioned previously, jitter represents the variation of the latency of packet delivery which affects streaming video and voice. In most typical scenarios a user will be sharing access to the AP with other STAs, so it is important that a RAA performs well or has a low jitter in congested scenarios.

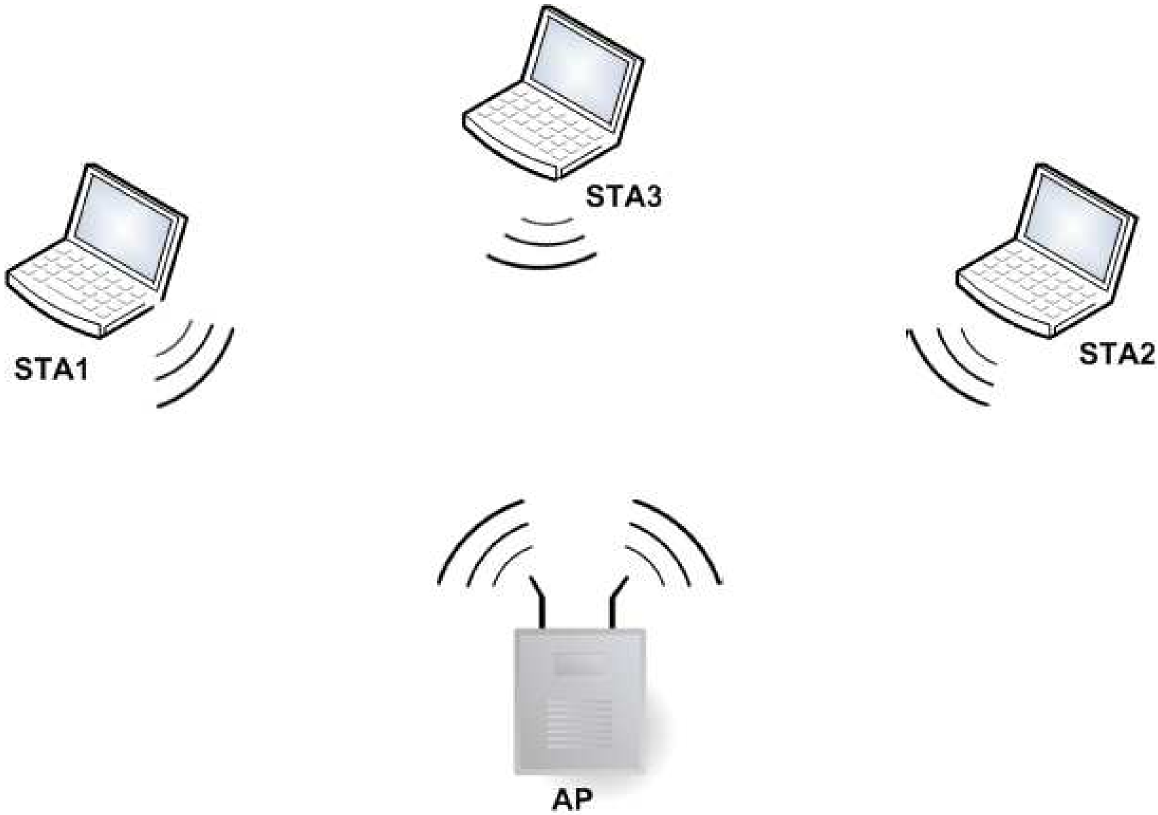

Figure 5.12: Connection diagram used during the RAAs evaluation under a congested scenario.

5.4 Summary

In this chapter, the performance of SARA was evaluated and compared against the standard benchmark SampleRate. Three scenarios were devised to compare the performance of SARA and SampleRate, static scenario, dynamic scenario and congested scenario.

In the static scenario, measurements were carried out on the top floor of the UCC with relatively few obstructions and people moving around. In this scenario SARA showed better performance when compared with SampleRate. Similarly, with the effect of the calibration phase removed, SARA can outperform SampleRate with a still higher throughput. In this scenario relatively good channel conditions are

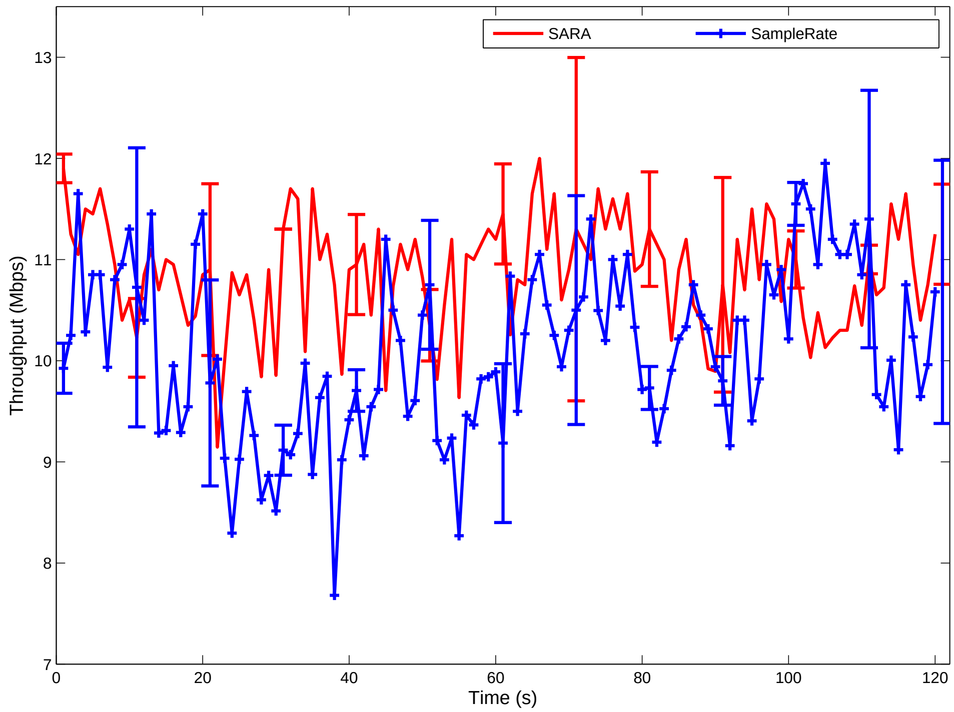

Figure 5.13: SARA versus SampleRate in a congested scenario with two additional STAs contending for the medium.

experienced due to the LOS and lack of movement. In this scenario most algorithms will transmit at or near the optimal data rate. Furthermore, any performance loss of SARA due to the calibration phase can be mitigated if the algorithm is used for a longer period of time, however as mentioned, the calibration phase effect is minimal. Also, in this scenario, the jitter was compared and SARA shows improved performance when compared to SampleRate.

In the dynamic scenario, measurements were carried out on the main floor of the UCC. In this scenario there are many people moving around as well as many wireless devices operating (laptops, cell phones, etc.) causing unknown interference and Doppler effects. In this scenario SARA outperform SampleRate and shows a

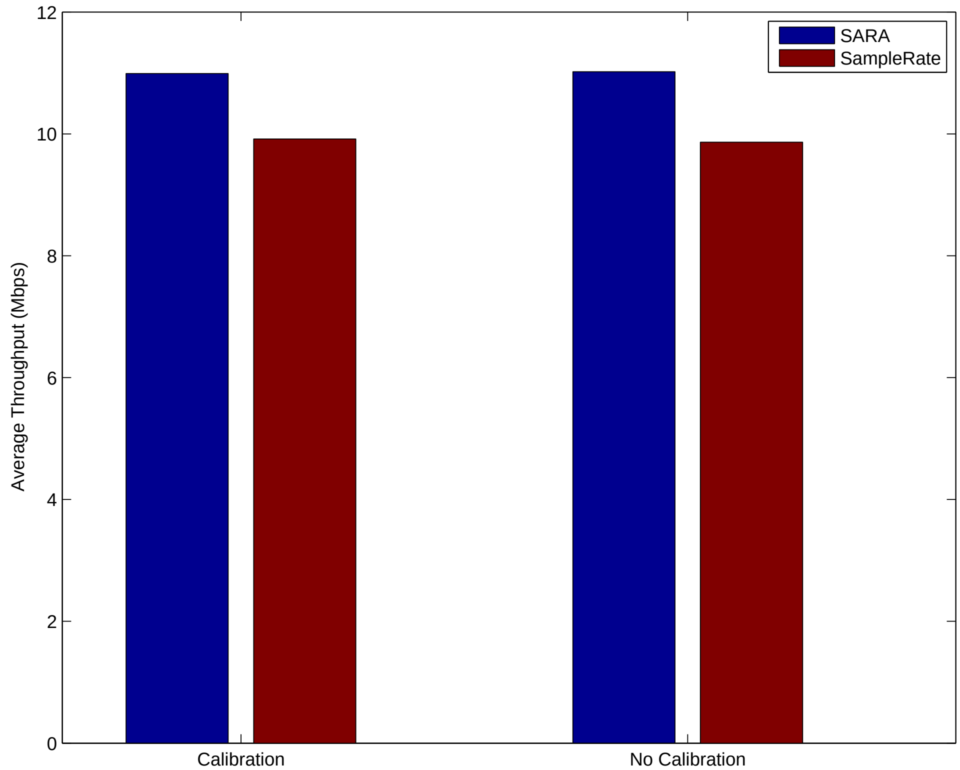

Figure 5.14: Average throughput gain of SARA versus SampleRate in a congested scenario with two additional STAs contending for the medium.

higher throughput with and without the calibration phase considered. In addition, the jitter experienced by SARA is lower than that of SampleRate indicating that in this type of scenario, streaming video and voice would have better performance if using SARA as the RAA. It should be noted that this scenario is considered typical, in that it may be similar to environments where an 802.11 device is typically used such as a library, coffee shop or other public space where people are moving around and other wireless devices are in use causing interference.

In the congested scenario, measurements were carried out in the Primus Canada Telecommunications Laboratory in the Thompson Engineering Building at UWO. In

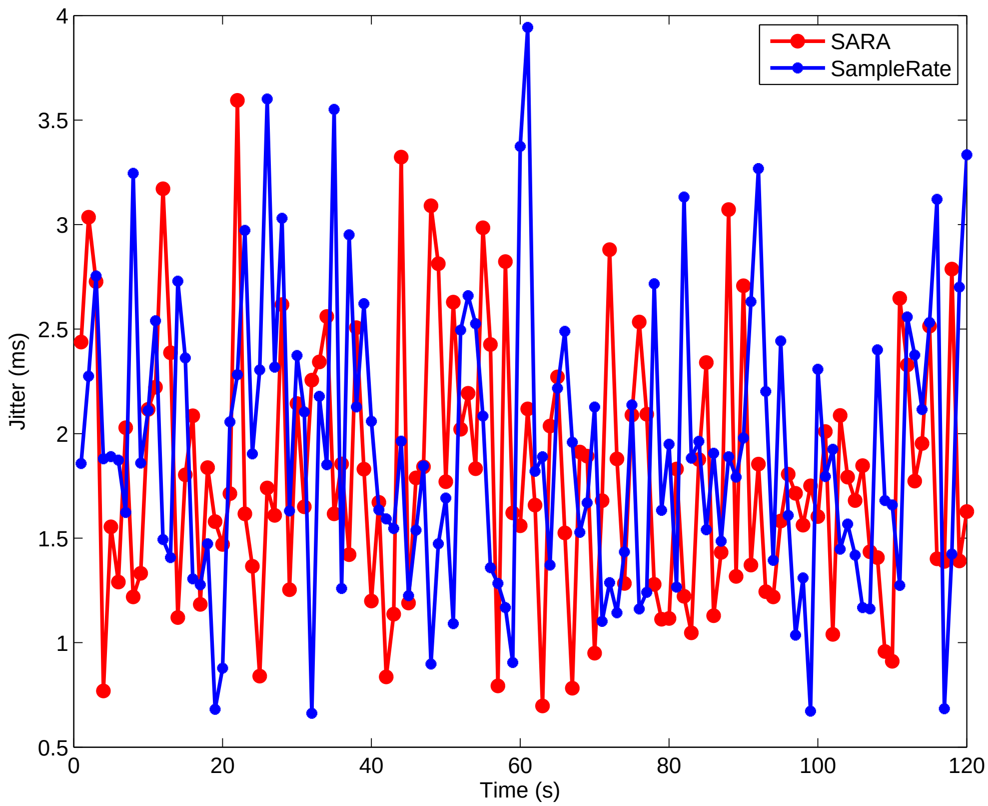

Figure 5.15: Jitter comparison between SARA and SampleRate in the congested scenario.

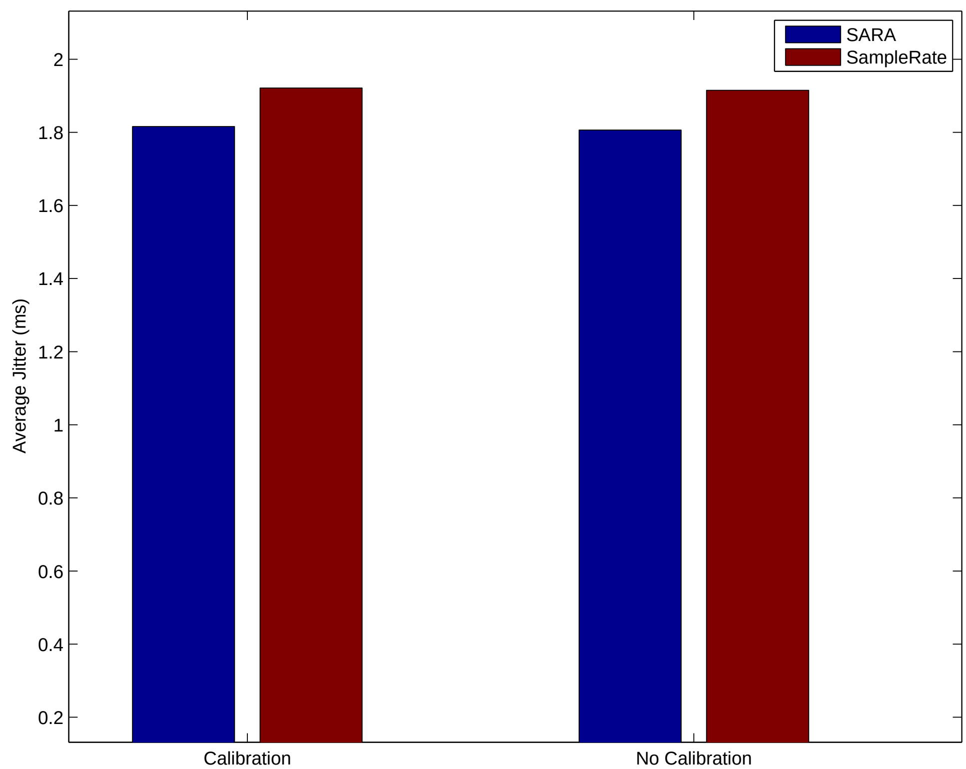

Figure 5.16: Average jitter performance gain comparison between SARA and SampleRate in the congested scenario. (A lower jitter is better.)

this scenario, a total of four devices were used with one device acting as the AP and three other STAs contending for the wireless medium. Out of the three STAs, one STA was used as the test device while the other two were traffic generators. In this scenario, SARA outperforms SampleRate in both throughput and jitter tests.

In summary, SARA is shown to have similar or better performance than the benchmark algorithm, SampleRate. Improved performance is experienced in the static scenario, the dynamic scenario and the congested scenario. The measurements show that SARA performs better than SampleRate in dynamic scenarios, meaning that it can adapt to rapid fluctuations in the environment better than SampleRate. In addition, SARA can mitigate the effects of congested scenarios by monitoring the average RSSI, which remains unaffected by collisions or congestion, whereas SampleRate reduces the data rate when experiencing collisions.

Chapter 6

Minimum Transmit Power Adaptation

In this chapter the details of the proposed approach to finding the minimum transmission power for 802.11 WLANs is presented. The approach uses the binary search method for quickly finding the minimum transmission power and relies on ACKs by the receiver to determine whether transmission was successful or not.

6.1 Minimum Transmission Power using the Binary Search

6.1.1 Algorithm Description

In order to determine the minimum transmission power, a search of the available power levels must be performed to determine at which power level connectivity can be maintained while minimizing power consumption. However, a search of all possible transmission power levels would be wasteful in terms of both power and packet loss. Therefore, an algorithm to quickly determine the minimum transmission power has been developed for this purpose. The algorithm can be described as follows:

1: Send packet at min transmission power.

2: Wait for ACK receipt. If received go to end.

3: Send packet at max transmission power.

4: Wait for ACK receipt. If not received go to end.

5: Send packet at $(max - min)/2$.

6: Wait for ACK receipt. If received go to step 7, otherwise go to step 8.

7: Set the current max to $(max - min)/2$, go to step 9.

8: Set the current min to $(max - min)/2$.

9: If $max - min == 1$, go to end, otherwise return to step 5.

The algorithm is performed by first transmitting a single packet at the lowest transmission power setting. If this transmission is successful, then immediately quit the search and continue transmitting at the lowest transmission power. If the packet is unsuccessful, however, then attempt transmission at the highest transmission power setting. If this transmission is unsuccessful, then immediately quit since there is no transmission power high enough for success. Otherwise, if transmission at the highest power setting is successful then the binary search for the optimal transmission power setting can be started. The lowest transmission power is then set as the minimum and the highest transmission power to the maximum. The next step is to transmit a packet at a power level mid way between the min and max transmission power setting, or $(max - min)/2$. If this transmission is successful, then this power level setting is set to the max transmission power, otherwise, it is set to the min. Continue in this fashion, dividing the transmission power range by two each time, only stopping when the max and min transmission power level differ by a single step. The following flow chart (Figure 6.1) depicts the algorithm graphically.

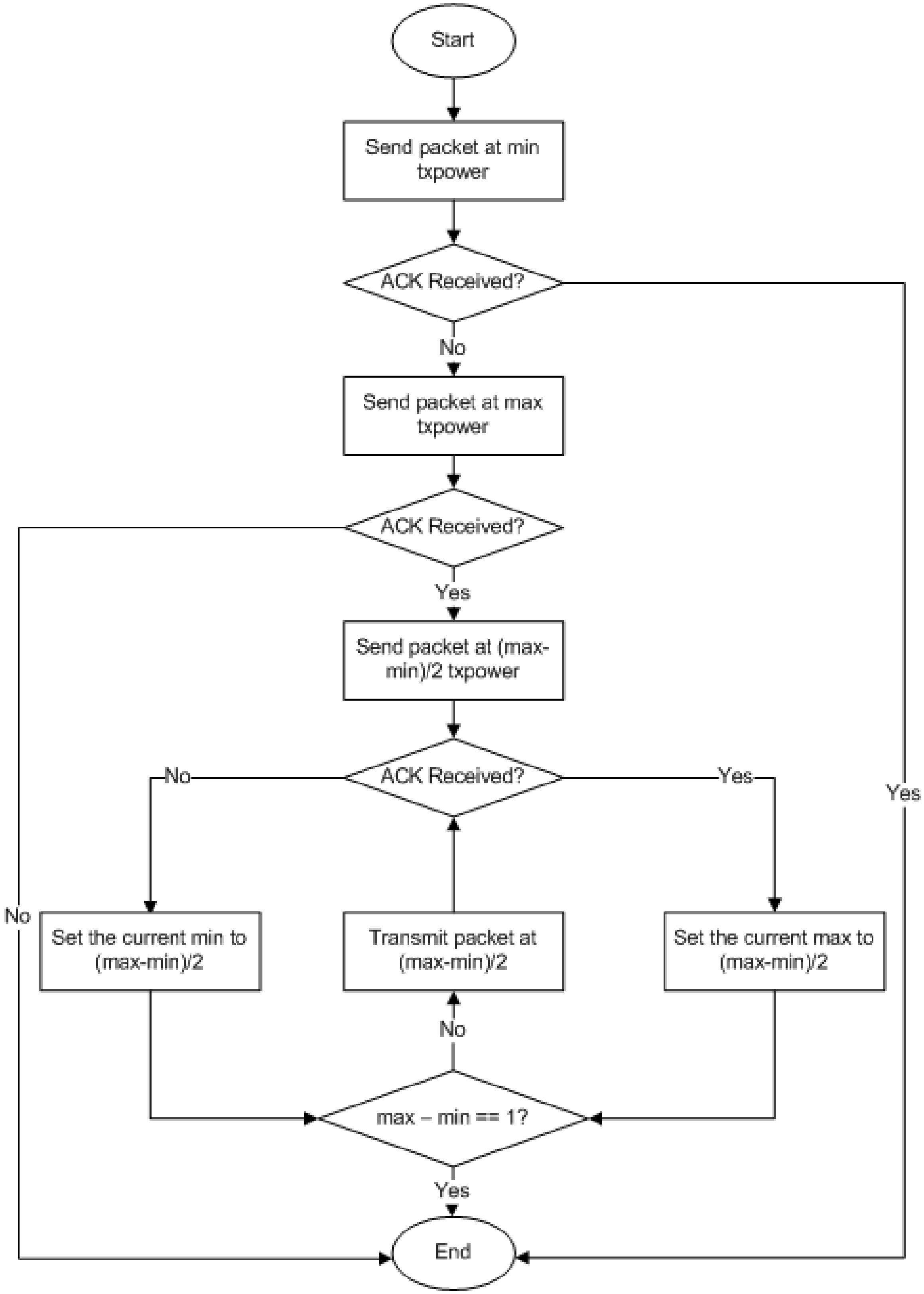

Figure 6.1: Minimum transmission power algorithm.

6.1.2 Energy Consumption Estimation

Once the minimum transmission power has been determined, the obtained power level setting can then be used to calculate the amount of energy used in the transmission of a single packet. The average transmission time, t_D, of a single packet of L_D bytes in length at a data rate r_D in bps can be estimated as

$$t_D(r_D) = \frac{L_D \times 8}{r_D} \tag{6.1}$$

However, one must also account for the ACK packet reply, which has an estimated transmission time given by

$$t_A(r_A) = \frac{L_A \times 8}{r_A} \tag{6.2}$$

where L_A is the length of the ACK packet in bytes and r_A is the ACK data rate. Therefore, the total time required to transmit a data packet successfully on the first try, i.e., ignoring retransmissions and back-off, including the receipt of the ACK packet is

$$t_{TX}(r_D) = DIFS + t_D(r_D) + SIFS + t_A(r_A) \tag{6.3}$$

where DIFS and SIFS are the interframe spaces. The time required to transmit a packet successfully after n retransmissions is then

$$\begin{gathered} t_{TXS}(r_D) = \\ n \times (DIFS + t_D(r_D) + SIFS + t_A(r_A)) \end{gathered} \tag{6.4}$$

The probability of packet error rate (PER) can be approximated by using the following equation

$$p_{PER}(r_D) = 1 - (1 - p_{BER}(r_D))^{L_D \times 8} \tag{6.5}$$

where p_{BER} is the probability of bit-error for a given modulation scheme and signal-to-noise ratio. Since a data packet is retransmitted until either an ACK packet is received successfully at the transmitter or the retransmission limit is reached, the probability that a data packet is successfully delivered after n trials is given by

$$p_S\left(n\right) = p_{PER}^{n-1}\left(1 - p_{PER}\right) \tag{6.6}$$

Therefore, the average time to transmit a packet successfully is

$$\begin{aligned}\overline{t_{TXS}} &= \sum_{n=1}^{N_{max}} t_{TXS}\left(n\right) p_S\left(n\right) \\ &= \sum_{n=1}^{N_{max}} \left[n\left(DIFS + t_D + SIFS + t_A\right)\right. \\ &\qquad \left.\times\, p_{PER}^{n-1}\left(1 - p_{PER}\right)\right]\end{aligned} \tag{6.7}$$

where N_{max} is the retransmission limit and we have omitted the data rate r_D and ACK rate r_A for simplicity. In fact, for the purposes of this algorithm, the data rate r_D and ACK rate r_A remain unchanged during packet transmission and for retransmissions.

The probability that a data packet is not delivered due to the retransmission limit being reached is

$$p_{TXFail} = p_{PER}^{N_{max}} \tag{6.8}$$

The time to transmit a packet after N_{max} retransmissions is

$$\begin{aligned}t_{TXF} =& N_{max} t_{TX} \\ =& N_{max}\left(DIFS + t_D + SIFS + t_A\right)\end{aligned} \tag{6.9}$$

The probability that a data packet is successfully delivered is

$$p_{TXS} = 1 - p_{TXFail} = 1 - p_{PER}^{N_{max}} \tag{6.10}$$

Therefore, the average data packet transmission time at a data transmission rate of r_D bps and a PER of p_{PER} is

$$\begin{aligned} \overline{t_{TX}}\left(r_D\right) = & \frac{1}{1 - p_{PER}\left(r_D\right)^{N_{max}}} \\ & \times \left(\overline{t_{TXS}}\left(r_D\right)\left(1 - p_{PER}\left(r_D\right)^{N_{max}}\right)\right. \\ & \left. + t_{TXF} p_{PER}\left(r_D\right)^{N_{max}}\right) \end{aligned} \tag{6.11}$$

Using the average transmission time $\overline{t_{TX}}\left(r_D\right)$ multiplied by the current power level gives the amount of energy used during the transmission of an average sized packet for a given data rate.

$$\epsilon\left(r_D\right) = \overline{t_{TX}}\left(r_D\right) \times p_{min}\left(r_D\right) \tag{6.12}$$

where $\epsilon\left(r_D\right)$ and $p_{min}\left(r_D\right)$ are the energy consumed and minimum transmission power for each rate r_D, respectively. This will give an indication of which power levels will result in higher throughput and possibly greater energy savings. In calculating the energy consumed, the DIFS and SIFS intervals can be ignored since during these times, no transmission is taking place.

6.2 System Setup and Test Results

All measurements were conducted using the Atheros based CM9-GP from Unex Technologies Corp. The CM9-GP is an IEEE 802.11a/b/g mini-PCI module which possesses the Atheros AR5213 Baseband/MAC and AR5112 5 GHz/2.4 GHz Transceiver. The CM9-GP was tested using the Atheros driver on a platform running the FreeBSD open source operating system. Two devices were used with one configured as an AP and the other configured as a STA. The antennas were removed to eliminate outside interference and cables were used to complete the connection. Additive white Gaussian noise (AWGN) was injected using the SMJ-100A signal generator from Rohde & Schwarz with signal and noise levels monitored using the FPU spectrum analyzer also from Rohde & Schwarz. Figure 6.2 depicts the testing scenario graphically. As shown in Figure 6.2, the AP and STA are connected via cables to a 'T' connector to facilitate the connection with the signal generator and the spectrum analyzer. The signal generator and spectrum analyzer are also connected with a 'T' connector with the signal generator injecting noise into the system. The spectrum analyzer allows for the monitoring of noise levels and corresponding transmission power levels of the 802.11 equipment. The STA is setup to use a static unicast rate to test all 802.11g rates with the maximum retries set to zero to ensure that success or failure of transmission does not rely on retransmissions. Transmission power level and static transmission rate settings were modified using the FreeBSD $ifconfig$ command, which possesses 47 different transmission power level settings ranging from 0 to 23. The minimum transmission power setting is 0 while the maximum transmission level setting is 23. Table 6.1 lists the noise levels tested for each rate.

Using the previously described testing apparatus, tests were carried out to evaluate the proposed binary search method for finding the minimum transmission power.

Data Rate (Mbps)	Noise Levels Tested (dBm)	Data Rate (Mbps)	Noise Levels Tested (dBm)
54	-27, -20, -18	9	-13, -6, 1
48	-27, -20, -15	6	-13, -6, 1
36	-20, -15, -10	11	-13, -6, 1
24	-20, -13, -6	5.5	-10, -3, 4
18	-13, -6, 0	2	-10, -3, 2
12	-13, -6, 1	1	-3, 1, 4

Table 6.1: Data rates and noise levels tested.

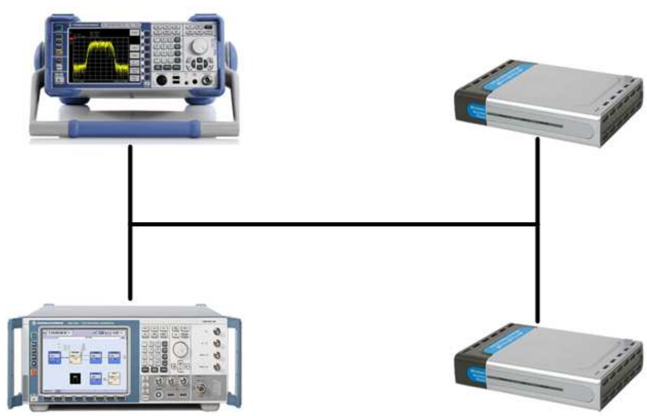

Figure 6.2: Testing scenario setup diagram.

Noise	MCS	Rate (Mbps)	Iterations	Min. Tx Power
-20	QAM	54, 48, 36, 24	7, 8, 8, 7	10.5, 8.5, 4.0, 0.5
-6	PSK	18, 12, 9, 6	7, 8, 7, 8	11.5, 9.5, 9.0, 8.5
-3	DSSS	11, 5.5, 2, 1	7, 7, 8, 8	10.5, 10.5, 9.5, 6.0

Table 6.2: Measurement results.

The MCSs have been grouped according to their modulation schemes. That is, for Quadrature Amplitude Modulation (QAM) we have 54, 48, 36 and 24 Mbps, Phase Shift Keying (PSK) we have 18, 12, 9 and 6 Mbps and DSSS we have 11, 5.5, 2 and 1 Mbps. Furthermore, for each grouping a common noise level has been selected so that a meaningful comparison may be made. Table 6.2 lists the experimental results.

In Figure 6.3, it can be seen that for a noise level of -20 dBm, 54 Mbps has a minimum transmission power level setting of 10.5 which was determined in 7 iterations, 48 Mbps has a minimum transmission power level setting of 8.5 which was determined in 8 iterations, 36 Mbps has a minimum transmission power level setting of 4.0 determined in 8 iterations and 24 Mbps has a minimum transmission power level setting of 0.5 found in 7 iterations. It can also be seen that, for a noise level of -6 dBm, 18 Mbps has a minimum transmission power level setting of 11.5 found in 7 iterations, 12 Mbps has a transmission power level setting of 9.5 found in 8 iterations, 9 Mbps has a transmission power level setting of 9.0 found in 7 iterations and 6 Mbps has a transmission power level setting of 8.5 found in 8 iterations. Finally, at a noise level of -3 dBm, 5.5 Mbps has a transmission power level setting of 10.5 found in 7 iterations, 2 Mbps has a transmission power level setting of 9.5 found in 8 iterations and 1 Mbps has a minimum transmission level setting of 6.0 found in 8 iterations.

It can clearly be seen that the minimum transmission power level setting for all data rates can be found within 7 or 8 iterations. This can be explained by examining the complexity of the binary search algorithm as discussed in Section II. The

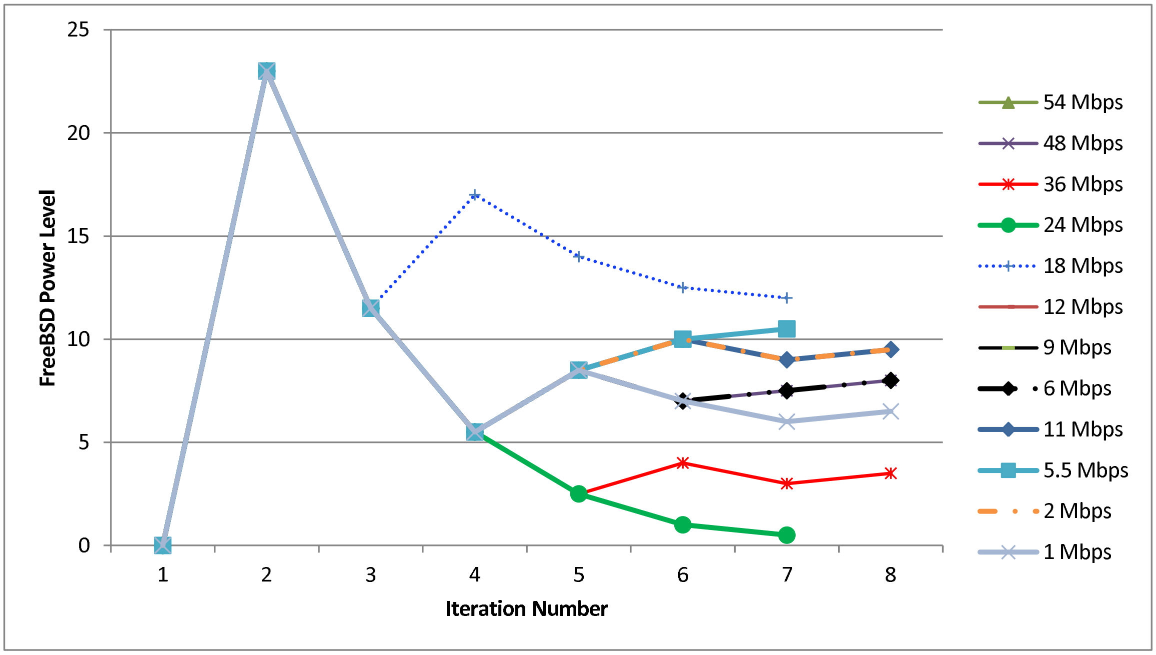

Figure 6.3: Minimum transmission power level setting using the binary search technique. Data rates 54, 48, 36 and 24 Mbps are tested at a noise level of -20 dBm, rates 18, 12, 9 and 6 Mbps are tested at a noise level of -6 dBm and rates 11, 5.5, 2 and 1 Mbps are tested at a noise level of -3 dBm.

worst-case performance of the binary search algorithm is given as $O(log_2(n))$. For our experiments on the Atheros platform, we have 47 available transmission power level settings, then $log_2(47) = 5.554589$, which when taking the ceiling of, gives $\lceil log_2(47) \rceil = 6$. However, each experiment actually requires around 7 or 8 iterations due to the inability to divide some transmission power level settings exactly by 2. The $ifconfig$ command and Atheros hardware can only accept transmission power level settings in 0.5 increments. Thus, in this case, the worst-case performance is 8 iterations.

Using the minimum transmission power found from the method, a calculation using MATLAB has been carried out to estimate the amount of energy consumed based on the average transmission time for a data packet derived previously. The minimum transmission power is multiplied by the average transmission time including

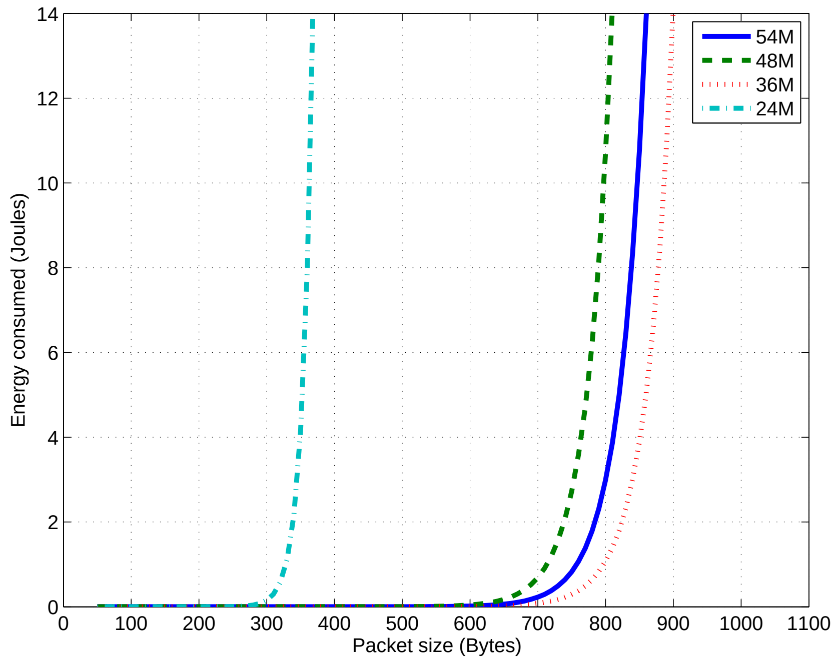

Figure 6.4: Energy consumed for QAM modulated rates.

retransmissions to determine the energy consumed. The packet size is varied from 50 to 1500 Bytes and the energy consumed is plotted.

As can be seen in Figure 6.4, for packets smaller than approximately 300 Bytes, all rates perform the same. However, it can also be seen that for packets larger than 300 Bytes, the 24 Mbps data rate starts to consume a larger amount of energy due to it's lower speed and resulting longer transmission time. For larger packets greater than 700 Bytes we can see that the 36 Mbps data rate is the best performing rate since, it can combine speed with low power requirements to minimize the energy consumed. For medium sized packets we can see that 54 or 48 Mbps may be a good choice if the additional throughput is required.

In Figure 6.5 it can be seen that 12 Mbps has the highest energy consumption once the packet size reaches about 250 Bytes. The lowest energy consuming rate is 9

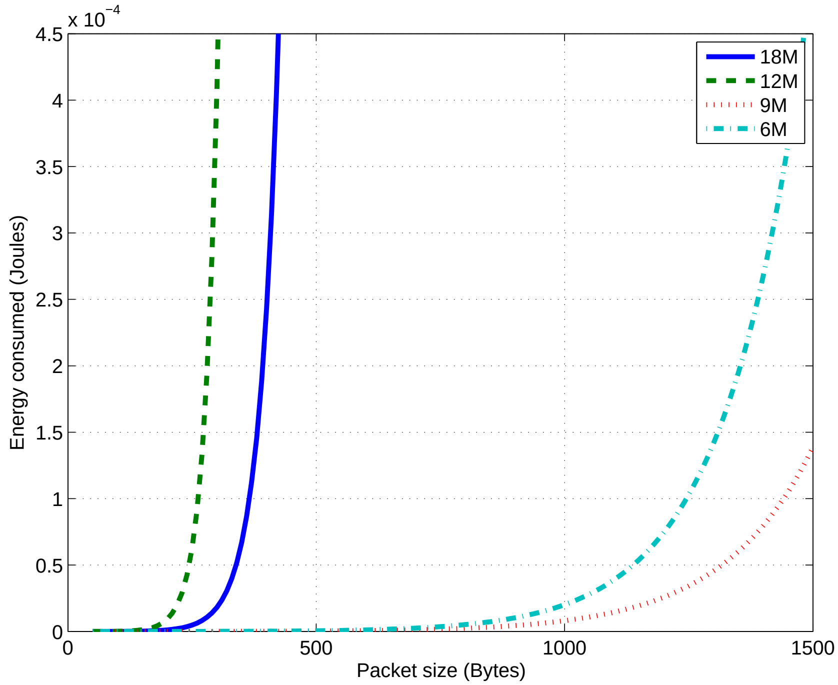

Figure 6.5: Energy consumed for PSK modulated rates.

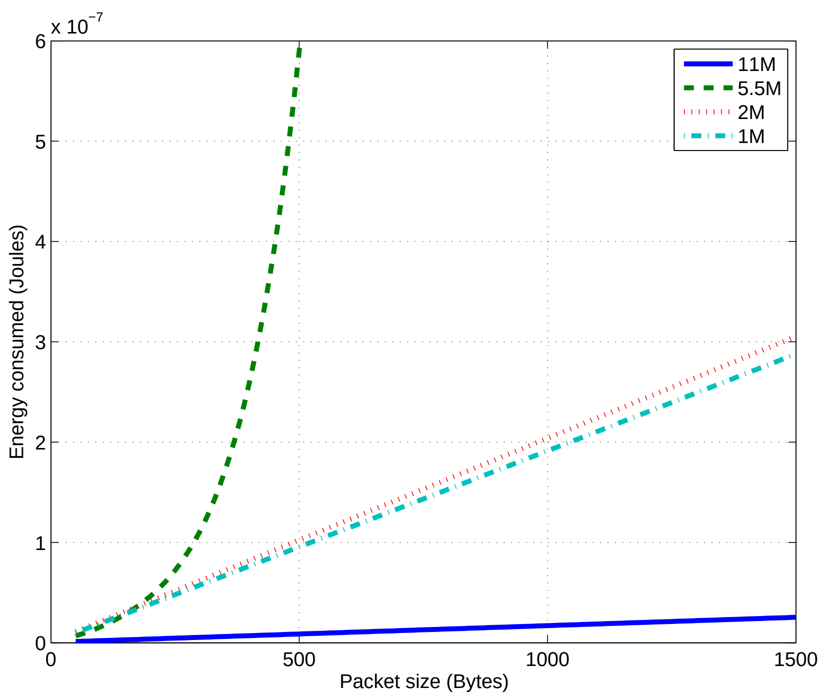

Figure 6.6: Energy consumed for DSSS modulated rates.

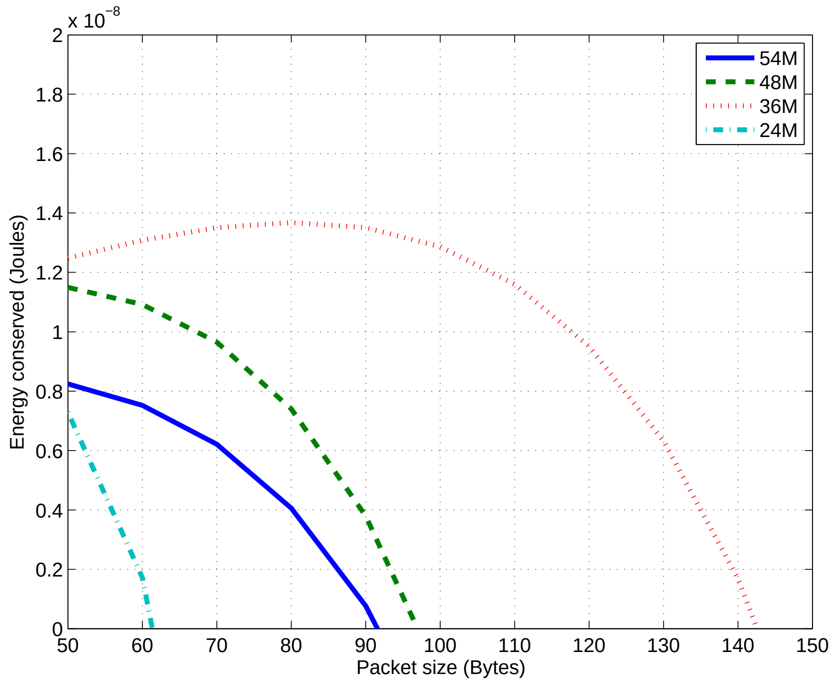

Figure 6.7: Energy conserved for QAM modulated rates.

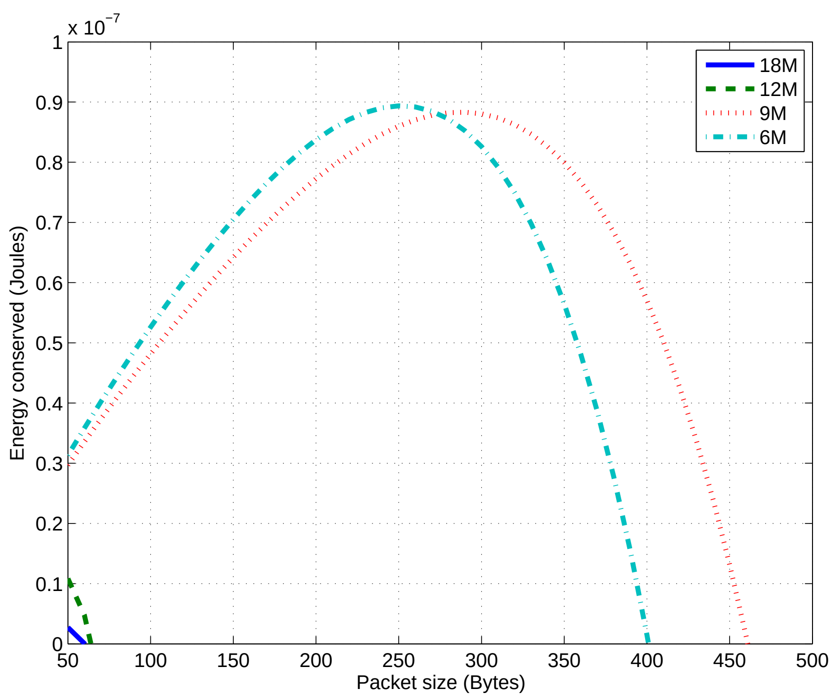

Figure 6.8: Energy conserved for PSK modulated rates.

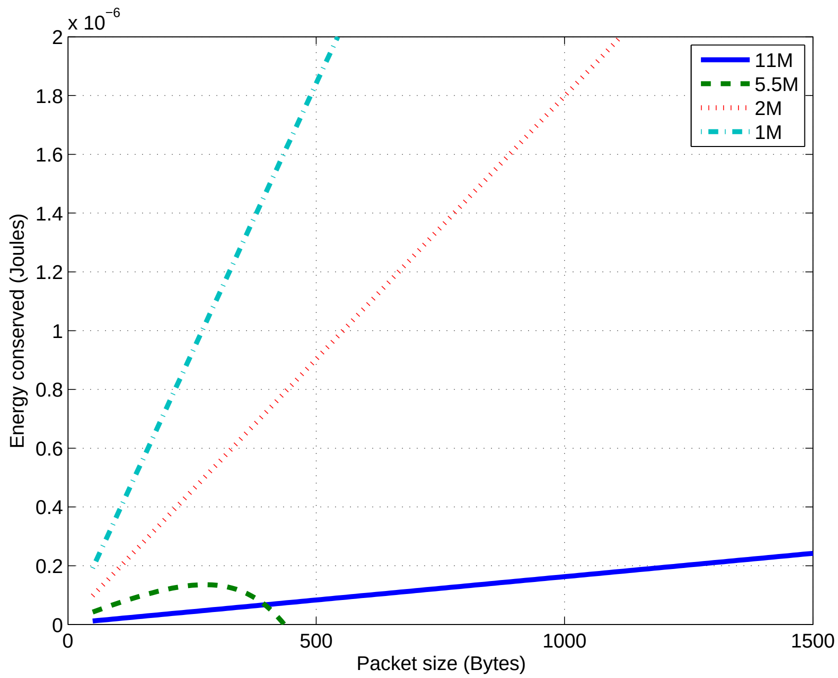

Figure 6.9: Energy conserved for DSSS modulated rates.

Mbps for large packet sizes. For medium sized packets 18 Mbps may be chosen, however, 6 Mbps should not be chosen since one can use 9 Mbps to get higher throughput and lower energy consumption.

In Figure 6.6 it can clearly be seen that 5.5 Mbps consumes a large amount of energy and that 11 Mbps should be chosen over all other rates if energy consumption is a factor. This figure also shows that 1 and 2 Mbps are much too slow and have a long transmission time thus, consuming more energy than 11 Mbps. Furthermore, it can also be seen that 11 Mbps provides excellent throughput with low energy consumption.

In Figures 6.7, 6.8 and 6.9 it can be seen the energy conserved. Here the difference between the energy used at the *max* transmission power and the energy used at the *min* transmission power has been calculated to show the energy conserved.

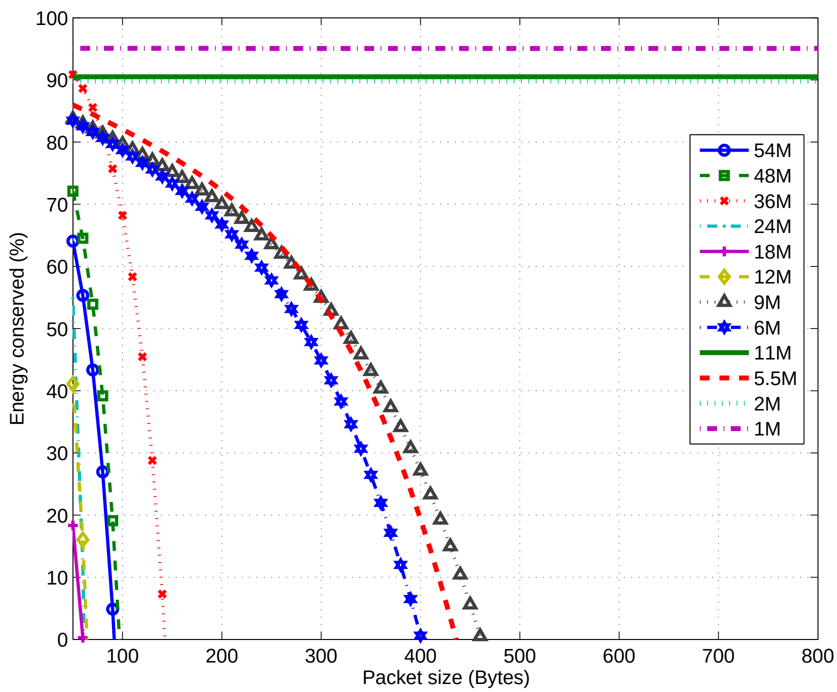

Figure 6.10: Ratio of energy conserved to the energy consumed using the maximum transmit power.

As can be seen from the figures, it is clear that for larger packet sizes, it becomes difficult to conserve energy due to the lower SNR at the minimum transmission power, which causes retransmissions and thus longer transmission time. However, once the minimum transmission power is found, the transmission power for a given data rate can be increased if larger packet sizes are required.

In Figure 6.10 the ratio of the energy conserved to the energy consumed using the maximum transmission power has been plotted. This gives a clear indication of the packet size required for each rate to obtain a high ratio of energy conservation. The result is that for an increase in packet size, an increase in transmission power may be necessary. That is for power control algorithms, the packet size should be included in the determination of minimum transmission power.

Chapter 7

Conclusion

This thesis presented a new RAA named Situation-Aware Rate Adaptation (SARA) and a new technique to determine the minimum transmission power for IEEE 802.11 WLANs. SARA employs a calibration phase to collect PER estimation which is then used to shift a RSSI table which maps the RSSI to data rates. The algorithm then uses the average RSSI to select a data rate based on the shifted RSSI table. Using the RSSI in rate selection allows the algorithm to react quickly to changes in the environment as well as avoid unnecessary data rate changes in congested scenarios. In order to ensure the algorithm is operating optimally, the PER is monitored during normal operation to ensure the rate selected using the average RSSI is providing the highest throughput. The PER gives the algorithm stability and ensures that the data rate selected is maximizing the throughput. Using a calibration phase allows the algorithm to become situation aware by shifting the RSSI table based on the current PER. SARA also recalibrates if the environment changes by detecting a change in the RSSI by monitoring the sample variance of the RSSI.

SARA determines the amount of interference and channel conditions during the calibration phase. It is because of this environmental awareness that the algorithm can perform optimally since the data transmission rate table can be adjusted to take advantage of each individual situation as it arises. Current RAAs lack situation awareness and cannot detect the environment, forcing them to rely on the limited feedback available in the form of the RSSI, PER and RTT.

Evaluation of SARA was performed in static, dynamic and congestion scenarios and is shown to have improved performance when compared to SampleRate which is considered the benchmark RAA. Additionally, a technique for determining the minimum transmission power is developed and evaluated. The technique uses a binary search to traverse through the transmission power levels to determine the minimum transmission power. The technique is shown to conserve battery power while maintaining QoS requirements.

SARA shows improved performance of up to 15% when compared to SampleRate due to it's ability to calibrate to the current environment and rapidly select data rates. In addition, using the average RSSI to select the data rate allows SARA to be adaptable to changing conditions which provides an ability to take advantage of good situations when they occur. SampleRate takes one to ten seconds to adapt to changing conditions which prevents the algorithm from opportunistically changing the data rate when conditions allow. As the performance results show in the static scenario, SARA shows higher throughput and increased stability. The static jitter test also shows that SARA has lower jitter than SampleRate which indicates that SARA delivers packets with low latency variation. In this static scenario, the channel conditions are good which makes it difficult for any algorithm to gain a significant advantage over others since most RAAs will converge towards similar data rates. In the dynamic scenario, the environment shows higher variation due to the large number of people and objects moving around in the test location. In this dynamic scenario, SARA also shows higher throughput when compared to SampleRate. The variation of SARA is higher than SampleRate which indicates that SARA is constantly following the environmental variations due to the movements of people and objects by selecting different data rates, whereas, SampleRate is consistently choosing similar data rates. SampleRate chooses from a subset of data rates which have been chosen by the algorithm to transmit at, however, the criteria to choose the data rate limits the number of data rates available. SampleRate selects data rates that have not experienced four consecutive losses in the last ten seconds and their lossless transmission time without retries must be less than the current data rate's. In poor conditions, this will limit the number of data rates that are eligible, thus preventing SampleRate from choosing the correct data rate. SARA also shows lower jitter than SampleRate in the dynamic scenario. In the congested scenario, SARA shows higher throughput than SampleRate as well as lower jitter. SARA uses the average RSSI to choose the data rate which gives the algorithm a slight resiliency to collisions. Packet collisions cause the PER or ETT to increase which causes algorithms such as SampleRate to have reduced performance due to their reliance on these metrics. Since SARA also uses the PER in addition to the average RSSI, SARA cannot be completely resilient to packet collisions.

Transmit power control can be used to minimize the transmission power during times of strong signal conditions or low noise and interference situations. Using an adaptive transmit power control allows the energy consumption to be optimized by adjusting according to current environmental situations. Optimizing the energy consumption results in increased battery life for power hungry smartphones, tablet computers and also laptops. The evaluation of the transmit power control algorithm shows that the minimum transmit power can be determined in as little as seven packet transmissions. Typical data transmissions consist of hundreds of thousands of data packets, therefore, the transmit power control algorithm can find the minimum transmit power with a very small number of packet transmissions when compared to the typical number of packet transmissions.

7.1 Future Work

There are several topics related to the work presented in this thesis that may be considered for future work. These topics include combining SARA with the minimum transmission power technique to optimize both rate selection and transmission power as well as further improving SARA by optimizing the calibration phase to reduce it's impact on the performance of SARA. The power optimization could be included in the calibration phase to determine a minimum transmission power necessary to maintain a given QoS. Furthermore, additional data rate tables can be developed based on different situations. As an example, if the calibration determines that the environment is favourable, an aggressive data rate table can be used, whereas, if the environment is poor, a more conservative data rate table can be used. Developing the data rate tables can also be improved. The data rate table used by SARA was developed by using the results of measurements in indoor scenarios, however, other data rate tables may be developed by performing measurements in additional environments and scenarios. Another modification to SARA could be to completely change algorithms during run time based on calibration or re-calibration. As an example, consider a favourable scenario where the current channel conditions show low noise and interference with relatively few packet losses. SARA could recalibrate and use a very aggressive algorithm for determining the data rate, or even switch to a fixed rate in this situation. Alternatively, in unfavourable conditions, SARA could choose a conservative algorithm that favours lower but more robust data rates. In this way, SARA could replace the entire rate selection algorithm, choosing completely different algorithms for different situations.

References

[1] Wikipedia, "Orthogonal frequency-division multiplexing — wikipedia, the free encyclopedia," http://en.wikipedia.org/w/index.php?title=Orthogonal_frequency-division_multiplexing&oldid=519020484, 2012, [Online; accessed 28-March-2012].

[2] *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Std. 802.11, 2007.

[3] E. Krouk and S. Semenov, Eds., *Modulation and Coding Techniques in Wireless Communications.* Chichester, U.K.: John Wiley and Sons, Ltd., 2011.

[4] S. Panichpapiboon, G. Ferrari, and O. Tonguz, "Optimal transmit power in wireless sensor networks," *Mobile Computing, IEEE Transactions on*, vol. 5, no. 10, pp. 1432–1447, Oct. 2006.

[5] S. Narayanaswamy, V. Kawadia, R. S. Sreenivas, and P. R. Kumar, "Power control in ad-hoc networks: Theory, architecture, algorithm and implementation of the compow protocol," in *European Wireless Conference*, 2002, pp. 156–162.

[6] G. L. Stuber and Y. G. Li, Eds., *Orthogonal Frequency Division Multiplexing for Wireless Communications.* New York: Springer, 2006.

[7] M. S. Gast, *802.11 Wireless Networks, The Definitive Guide.* Santa Rosa, U.S.: O'Reilly Media, 2002.

[8] J. Mikulka and S. Hanus, "Complementary code keying implementation in the wireless networking," in *Systems, Signals and Image Processing, 2007 and 6th EURASIP Conference focused on Speech and Image Processing, Multimedia Communications and Services. 14th International Workshop on*, June 2007, pp. 315–318.

[9] X. Chen, P. Gangwal, and D. Qiao, "Ram: Rate adaptation in mobile environments," *Mobile Computing, IEEE Transactions on*, vol. 11, no. 3, pp. 464–477, March 2012.

[10] D. Qiao, S. Choi, A. Jain, and K. Shin, "Adaptive transmit power control in ieee 802.11a wireless lans," in *Vehicular Technology Conference, 2003. VTC 2003-Spring. The 57th IEEE Semiannual*, vol. 1, April 2003, pp. 433–437.

[11] S.-Q. Wang and J.-N. Nie, "An approach of optimizing power in wsn with random topology under rayleigh channels," in *Wireless Communications, Networking and Mobile Computing, 2008. WiCOM '08. 4th International Conference on*, Oct. 2008, pp. 1–5.

[12] K. Ramachandran, R. Kokku, H. Zhang, and M. Gruteser, "Symphony: Synchronous two-phase rate and power control in 802.11 wlans," *Networking, IEEE/ACM Transactions on*, vol. 18, no. 4, pp. 1289–1302, Aug. 2010.

[13] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. New York: McGraw-Hill, 2002.

[14] J. C. Bicket, "Bit-rate selection in wireless networks," Master's thesis, Massachusetts Institute of Technology, Massachusetts, U.S., 2005.

[15] Q. Xia and M. Hamdi, "Smart sender: a practical rate adaptation algorithm for multirate ieee 802.11 wlans," *IEEE Transactions on Wireless Communications*, vol. 7, no. 5, pp. 1764–1775, May 2008.

[16] M. Lacage, M. H. Manshaei, and T. Turletti, "Ieee 802.11 rate adaptation: a practical approach," in *Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, ser. MSWiM '04. New York, NY, USA: ACM, 2004, pp. 126–134.

[17] "Bit-rate selection algorithms," http://madwifi-project.org/wiki/UserDocs/RateControl, the MadWifi Project.

[18] M. Khan and D. Veitch, "Smartrate: A new dynamic rate adaptation algorithm for 802.11 wireless networks," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a*, June 2011, pp. 1–10.

[19] G. Judd, X. Wang, and P. Steenkiste, “Efficient channel-aware rate adaptation in dynamic environments,” in *Proceedings of the 6th international conference on Mobile systems, applications, and services*, ser. MobiSys ’08. New York, NY, USA: ACM, 2008, pp. 118–131.

[20] G. Holland, N. Vaidya, and P. Bahl, “A rate-adaptive mac protocol for multi-hop wireless networks,” in *Proceedings of the 7th annual international conference on Mobile computing and networking*, ser. MobiCom ’01. New York, NY, USA: ACM, 2001, pp. 236–251.

[21] S. H. Y. Wong, H. Yang, S. Lu, and V. Bharghavan, “Robust rate adaptation for 802.11 wireless networks,” in *Proceedings of the 12th annual international conference on Mobile computing and networking*, ser. MobiCom ’06. New York, NY, USA: ACM, 2006, pp. 146–157.

[22] B. Sadeghi, V. Kanodia, A. Sabharwal, and E. Knightly, “Oar: an opportunistic auto-rate media access protocol for ad hoc networks,” *Wirel. Netw.*, vol. 11, no. 1-2, pp. 39–53, Jan. 2005.

[23] J. Pavon and S. Choi, “Link adaptation strategy for ieee 802.11 wlan via received signal strength measurement,” in *Communications, 2003. ICC ’03. IEEE International Conference on*, vol. 2, 2003, pp. 1108–1113.

[24] J. Zhou, M. Jacobsson, E. Onur, and I. Niemegeers, “A novel link quality assessment method for mobile multi-rate multi-hop wireless networks,” in *Consumer Communications and Networking Conference, 2009. CCNC 2009. 6th IEEE*, Jan. 2009, pp. 1–5.

[25] S.-H. T. Mathias Pauli, Udo Wachsmann, “Quality determination for a wireless communications link,” Patent US 7 231 183, 06 12, 2004.

[26] S. Tsai and A. C. K. Soong, “Effective-snr mapping for modeling frame error rates in multiple-state channels,” http://ftp.3gpp2.org/TSGC/working/2003/2003-05-SanDiego/TSG-C-2003-05-San%20Diego/WG3/WG3%20Call,%202003.04.29/C30-20030429-010_Ericsson_Effective_SNR_Mapping.PDF, 2003.

[27] "Initial results on link-level performance modeling for system-level evaluation with coloured interference," http://www.quintillion.co.jp/3GPP/TSG_RAN/TSG_RAN2004/TSG_RAN_WG1_RL1_2.html, 2004.

[28] A. Tamimi, "Exponential effective signal to noise ratio mapping (eesm) computation for wimax physical layer," Master's thesis, Washington University, 2007.

[29] R. Giuliano and F. Mazzenga, "Dimensioning of ofdm/ofdma-based cellular networks using exponential effective sinr," *Vehicular Technology, IEEE Transactions on*, vol. 58, no. 8, pp. 4204–4213, Oct. 2009.

[30] H. Song, R. Kwan, and J. Zhang, "Approximations of eesm effective snr distribution," *Communications, IEEE Transactions on*, vol. 59, no. 2, pp. 603–612, Feb. 2011.

[31] A. Papoulis and S. U. Pillai, *Probability, Random Variables and Stochastic Processes*. New York: McGraw-Hill, 2002.

[32] E. Westman, "Calibration and evaluation of the exponential effective sinr mapping (eesm) in 802.16," Master's thesis, The Royal Institute of Technology (KTH), 2006.

[33] L. Szczecinski, A. Alvarado, and R. Feick, "Distribution of max-log metrics for qam-based bicm in fading channels," *Communications, IEEE Transactions on*, vol. 57, no. 9, pp. 2558–2563, Sep. 2009.

[34] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*, 1st ed. Cambridge, U.K.: Cambridge University Press, 2005.

[35] D. Qiao, S. Choi, and K. Shin, "Goodput analysis and link adaptation for ieee 802.11a wireless lans," *Mobile Computing, IEEE Transactions on*, vol. 1, no. 4, pp. 278–292, Oct-Dec 2002.

[36] L. Stewart and J. Healy, "An introduction to freebsd 6 kernel hacking," http://caia.swin.edu.au/reports/070622A/CAIA-TR-070622A.pdf, 2007.

Curriculum Vitae

Name:	Jay Nadeau
Place of birth:	London, Ontario, Canada
Year of birth:	1978
Post-secondary Education and Degrees:	2010–2012, Candidate, Master of Engineering Science The University of Western Ontario London, Ontario, Canada 2004–2010, Bachelor of Engineering Science The University of Western Ontario London, Ontario, Canada 2004–2010, Bachelor of Science The University of Western Ontario London, Ontario, Canada 2001–2004, Diploma in Computer Programming Fanshawe College London, Ontario, Canada
Honours and Awards:	Ontario Graduate Scholarship (OGS-M) 2011–2012 MacKay-Lassonde Award in Computer Engineering 2009–2010
Teaching Experience:	SE3314–Computer Network Applications The course covers advanced concepts in computer networking and data communications The University of Western Ontario Spring 2011 and Spring 2012 ES1036–Programming Fundamentals for Engineers The course covers the fundamentals of Object-Oriented

analysis, design and implementation using C++
The University of Western Ontario
Spring 2012

Publications:

[1] J. G. Liu, X. Wang, J. Nadeau, and H. Lin, "Modulation Classification Based on Gaussian Mixture Models under Multipath Fading Channel," to appear in *GlobeCom 2012 Signal Processing for Communications Symposium*, Anaheim, U.S.A., Dec. 2012.

[2] V-H. Pham, X. Wang, J. Rahman, and J. Nadeau, "Channel Prediction-Based Adaptive Power Control for Dynamic Wireless Communications," in *Proc. 73rd Vehicular Technology Conference*, Budapest, Hungary, May 2011, pp. 1–5.

[3] V-H. Pham, X. Wang, and J. Nadeau, "Long Term Cluster-Based Channel Envelope and Phase Prediction for Dynamic Link Adaptation," in *Proc. IEEE Communication Letters*, vol. 15, no. 7, 2011, pp. 713–715.

[4] J. G. Liu, X. Wang, J. Nadeau, and P. Ho, "Blind Parameter Estimation for OFDM Interception Receiver with Iterative Cyclostationary Analysis," in *Proc. Military Communications Conference*, 2011, pp. 2211–2215.

[5] V-H. Pham, X. Wang, J. Nadeau, and J.Y. Chouinard, "Cluster-based Time-domain Channel Prediction for Dynamic Wireless Communications," in *Proc. GlobeCom Workshops*, 2010, pp. 59–63.