Experimental Study on the Performance of Rate Adaptation Algorithm in IEEE 802.11g Networks

by

Dong Xia

A thesis
submitted to the Victoria University of Wellington
in fulfilment of the
requirements for the degree of
Master of Engineering
in Networking Engineering.

Victoria University of Wellington
2013

Abstract

IEEE 802.11 technology provides a low-cost wireless networking solution. In the last few years, we have seen that the demand for high-bandwidth wireless local area networks increases rapidly, due to the proliferation of mobile devices such as laptops, smart phones and tablet PCs. This has driven the widespread deployment of IEEE 802.11 wireless networks to provide Internet access. However, wireless networks present their own unique problems. Wireless channel is extremely variable and can be affected by a number of different factors, such as collisions, multipath fading and signal attenuation. As such, rate adaptation algorithm is a key component of IEEE 802.11 standard which is used to vary the transmission data rate to match the wireless channel conditions, in order to achieve the best possible performance. Rate adaptation algorithm studies and evaluations are always hot research topics. However, despite its popularity, little work has been done on evaluating the performance of rate adaptation algorithms by comparing the throughput of the algorithm with the throughput of the fixed rates.

This thesis presents an experimental study that compares the performance of MikroTik rate adaptation algorithm and *Minstrel* rate adaptation algorithm against fixed rates in an IEEE 802.11g network. MikroTik and *Minstrel* rate adaptation algorithm are most commonly used algorithm around the world. All experiments are conducted in a real world environment in this thesis. In a real world environment, wireless channel conditions are not tightly being controlled, and it is extremely vulnerable to interference of surrounding environment. The dynamic changes of wireless channel conditions have a considerable effect on the performance of rate

adaptation algorithms. The main challenge of evaluating a rate adaptation algorithm in a real world environment is getting different experiment behaviours from the same experiment. Experiment results may indicate many different behaviours which due to the leak of wireless environment controlling. Having a final conclusion from those experiment results can be a challenge task.

In order to perform a comprehensive rate adaptation algorithm evaluation. All experiments run 20 times for 60 seconds. The average result and stand deviation is calculated. We also design and implement an automation experiment controlling program to help us maintain that each run of experiment is following exactly the same procedures.

In MikroTik rate adaptation algorithm evaluation, the results show in many cases that fixed rate outperforms rate adaptation. Our findings raise questions regarding the suitability of the adopted rate adaptation algorithm in typical indoor environments. Furthermore, our study indicates that it is not wise to simply ignore fixed rate. A fine selection of a fixed rate could be made to achieve desired performance.

The result of *Minstrel* rate adaptation evaluation show that whilst *Minstrel* performs reasonably well in static wireless channel conditions, in some cases the algorithm has difficulty selecting the optimal data rate in the presence of dynamic channel conditions. In addition, *Minstrel* performs well when the channel condition improves from bad quality to good quality. However, *Minstrel* has trouble selecting the optimal rate when the channel condition deteriorates from good quality to bad quality.

By comparing the experimental results between the performance of rate adaptation algorithms and the performance of fixed data rate against different factors, the experiment results directly pointed out the weakness of these two rate adaptation algorithms. Our findings from both experiments provide useful information on the design of rate adaptation algorithms.

Acknowledgments

Foremost, I would like to express my gratitude to my supervisor Dr. Qiang Fu for the continuous help and support of my master study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better supervisor and mentor for my master study.

Besides my supervisor, I would like to thank the rest of my lecturers: Prof. Winston Seah, Dr. Ian Welch, Dr. Kris Bubendorfer, and Andy Linton, for their encouragement, insightful comments, and help.

I would also like to thank my fellow lab mates: Joanthan Hart, Casimirraj Subramanian, Mike Lin, Zakwan Al-Arnaout and Nan Liu, for the discussions and for all the fun we have had in the last one year. Also I thank technical staff Mark Davies, Roger Cliffe, and Radek Hes from school of engineering and computer science, for supplying all my experimental equipment.

Last but not the least, I would like to thank my parents and my wife, for supporting me throughout my life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

People keep looking for a low-cost, high throughput, and portable Internet access. In the year 1991, NCR Corporation/AT&T invented 802.11 wireless networks[3] (also known as, Wi-Fi). Sicne then, the demand for high bandwidth wireless local area networks has been increased rapidly, due to the proliferation of mobile devices such as laptops, smart phones, smart TV and so on. An IEEE 802.11 network usually contains at least one Access Point (AP) and one or more client nodes. Each AP broadcasts beacons every 100ms (sending information like, Service Set Identifier (SSID), Network name). Clients use these beacons to decide whether or not to connect to the AP. IEEE 802.11 networks use an unlicensed radio spectrum. The frequency bands are 2.4, 3.6 and 5 GHz. Nowadays, Many IEEE 802.11 products are available in the market. Unlike cellular networks, the same IEEE 802.11 devices work in different countries around the world without any modification to the devices. They can be used to extend the coverage of an existing network or to deploy a new network with much less costs.

However, wireless networks also present their own unique problems. Wireless channels are extremely variable, and it can be affected by a number of different factors, such as interference from other wireless devices, multipath fading and signal attenuation. Radio spectrum is a limited resource in most counties around world, therefore, the number of unlicensed free channels are limited. Most countries allow channels 1 through to 11, some countries may allow channels 10 and 11 only, where else there are other countries, that may allow all 14 channels[4][5]. Due to the continuous growth of wireless networks and the limited wireless channels, IEEE 802.11 networks are becoming continuously congested. The congestion of wireless networks causes serious interference among wireless devices within ranges. There are also some new challenges that need to be tackled, such as the use of new building materials, which can alter channel conditions. Therefore, it is essential that the wireless MAC layer is modified to be able to quickly adapt to the changes in the wireless channel conditions.

As such, one of the key components of the IEEE 802.11 network is the rate adaptation algorithm. Rate adaptation algorithm allows wireless devices to adapt data rates to the channel conditions. For example, a high data rate should be used in order to achieve fast transmission when wireless channel conditions are good. However, if wireless conditions are bad, high data rates are more susceptible to bit errors, meaning that more packets get corrupted. Therefore, a low data rate should be used to transmit packets in order to reduce packet losses. Rate adaptation algorithm is therefore an optimization problem that attempts to determine the optimal rate for transmission based on the current wireless channel condition, in order to provide the best possible performance.

Rate adaptation algorithm is critical to the performance of IEEE 802.11 networks. There have been several studies to develop effective rate adaptation algorithms to take advantage of the low-cost and portable networks [6][7][8][9][10][11] [12][13][14][15][16][17] [18]. All of these studies have pointed out that the performance of the wireless network depends on how accurately the rate adaptation algorithm can predict the channel conditions to transmit the next packet. Based upon the metrics used to predict channel conditions, rate adaptation algorithms can be grouped into three categories: ACK packet based approaches, signal-to-noise ratio based approaches, and bit error rate based approaches. Based on how much emphasize is given towards the importance of efficiency in performance of wireless networks in a rapidly changing environment, it is vital to research on increasing the effectiveness of rate adaptation algorithms. Nevertheless, a good understanding of how rate adaptation algorithm performs in comparison with fixed rate is a fundamental step towards the design of more effective rate adaptation algorithms. Moreover, it has a practical value to the configuration of IEEE 802.11 networks. Unfortunately, in the literature these issues have not been clearly studied.

1.1 Research Question

This thesis presents the results of two experimental studies which compare the performance of rate adaptation algorithm and fixed rates. Both experimental studies used a MikroTik router as Access Point (AP). We choose two most commonly used rate adaptation algorithms (MikroTik and *Minstrel*) for our study. In the first study the AP was installed with a MikroTik routerOS operating system. Integrated with the operating system is MikroTiks proprietary rate adaptation algorithm. In the second study, the AP was installed with a Linux based operating system called OpenWRT. *Minstrel* rate adaptation algorithm is the default setting for the OpenWRT operating system. Studies take into account various factors such as power level, dynamic wireless channel condition and distance between client and AP. The following lists the research questions of the thesis.

- How effective would a commercial rate adaptation algorithm be in IEEE 802.11g networks?

- How effective would *Minstrel* rate adaptation algorithm be in an IEEE 802.11g networks?

- What are the potential problems which would cause *Minstrel* to perform less efficiently than expected?

1.2 Research Methodology

Data rate adaptation algorithm studies can be classified into three major categories: simulation based studies, emulation based studies, and field based studies. Every approach has its unique advantages and disadvantages.

Simulation based studies use a simulation program to observe the operations and the behaviour without performing real operations. The advantage of using simulation program to study rate adaptation algorithm is having a totally controlled environment and also been much less time consuming. Wireless channel conditions can be interfered by many external sources. Having a totally controlled environment can provide repeatable experiment results for the user. However, the similarity of simulated environment to a real world environment depends on the user′s inputs and how they control the environment. No matter how well the simulation program is configured, it is still not able to fully replicate a real wireless environment. Artificiality and limited experiment behaviour are two disadvantages of using simulation based studies.

Emulation Based Studies use physical devices to obtain a controlled emulated environment while performing real operations. Emulation based study reduces the artificiality of the experiment by observing real operations. The emulated environment is also created through a user′s inputs. Therefore, experiment results still cannot fully represent the situations which happen in a real environment due to the complexity and diversity of wireless conditions.

Field based studies use physical devices to perform real options in a real world environment. It cannot have a tight control of the environment, but it provides user the advantage of being far less artificial than the other two experimental methodologies. It requires user to repeat the same experiment many times to summarise one final conclusion. Therefore, getting experiment results from field based studies requires more time than the other two methodologies.

The complexity and diversity of wireless environment brings many challenges to the design of effective rate adaptation algorithm. However, the complexity and diversity of wireless environment reflects the key problem of rate adaptation algorithm design and implementation. Simulation and emulation based studies take this complexity and diversity away from user to explore the real issues of target algorithms in a wireless environment. Therefore, it is important to study the performance of rate adaptation algorithms based on field experiment. This thesis is reflected upon field based experimental studies.

1.3 Contributions

The contributions of this research are:

- Designed and implemented an automation testbed controller program. In order to have accurate and comparable result from experiments, it is important to keep all the experimental procedure identical. The automated testbed controller ensures that all experiments follow the same experimental procedure. It is also able to simultaneously control and manage experiment devices at the same time.

- Evaluated the performance of MikroTik rate adaptation by comparing the performance of the rate adaptation and fixed rate, taking into account various factors. Results show in many cases that fixed rate outperforms MikroTik rate adaptation algorithm, especially when high fixed rates are used. The performance gain by fixed rate is even greater if low power level is used. This could include the scenarios where smaller network coverage is desired to accommodate more networks and thus more users.

- Carefully studied and documented the source code of *Minstrel* rate adaptation algorithm. *Minstrel* rate adaptation algorithm is one of the most popular algorithms. Many IEEE 802.11 devices use it. However, there is lack of information that explains how *Minstrel* rate adaptation algorithm works in detail. Therefore, it is important to create the documentation that explains how *Minstrel* works step by step. Our documentation will benefit the future study of design rate adaptation algorithms.

- Evaluated the performance of *Minstrel* rate adaptation algorithm by comparing the performance of the rate adaptation and fixed rate in both uplink and downlink situations in a real world 802.11g testbed. The comparison was conducted with various factors taken into account, including transmission power, channel dynamic, Access Point location, and datagram size, etc. The results show that while *Minstrel* performs well in many cases (particularly with good or stable channel conditions), the algorithm has poor performance compared to fixed rates in the presence of poor or highly dynamic channel conditions.

1.4 Publication

The following list are published papers as part of this research study.

- *An Experimental Study on Performance Comparison of Rate Adaptation and Fixed Rate in IEEE 802.11g*[19] was published in Australasian Telecommunication Networks & Applications Conference (ATNAC) in 2011.

- *On the Performance of Rate Control Algorithm Minstrel*[20] was published in IEEE International Symposium on Personal, Indoor and Mobile Radio Communication (PIMRC) in 2012.

- *Evaluation of the Minstrel Rate Adaptation Algorithm in IEEE 802.11g WLANs*[21] was published in IEEE International Conference on Communications (ICC) in 2013.

1.5 Overview of Chapters

Chapter 2 **Background** provides related background knowledge of IEEE 802.11 technologies: MAC layer, Physical layer, and rate adaptation algorithms. It is important to understand these knowledge before moving onto next chapter. This chapter outlined three main wireless interference, and how how IEE 802.11 mechanisms maintain the performance and reliability of wireless communications. It also outlines three different types of rate adaptation algorithms, ACK based approaches, SNR based approaches, and BER based approaches.

Chapter 3 **Related Work** describes what other researchers have done in rate adaptation algorithm research areas. With our literature reviews, we classify these works into three groups based on research methodologies. In this chapter, we compared our approaches with other people's approaches. The differences between our work and other works were highlighted in the chapter.

Chapter 4 **Experimental Methodology** outlines three relevant experiment methodologies, simulation based studies, emulation based studies, and field based studies. Every methodology has its unique pros and cons. After we carefully compared these methodologies, we decided to carry out our study based field based approach. This chapter explains the reasons behind our choice.

Chapter 5 **Experiment Design** describes how the testbed and the automatic experiment controller were designed to fulfil the requirements of the study. Chapter 4 has described the pros and cons of our experiment methodology. This chapter lists issues of field based approach and some other issues, such as the interference from our controller program. Solutions are proposed to resolve all listed issues.

Chapter 6 **MikroTik Experiment Study** covers the performance comparisons to MikroTik rate adaptation algorithm vs. fixed rate against different factors such as the distance between sender node and receiver node, AP transmission power level and RTT for TCP traffic. Findings are listed in this chapter.

Chapter 7 **Theoretical Study of *Minstrel*** explores *Minstrel* rate adaptation algorithm by explaining the working process of *Minstrel* and key factors from the study of source code. A potential issue was discovered from our theoretical study.

Chapter 8 ***Minstrel* Experiment Study** explores *Minstrel* rate adaptation algorithm in detail by comparing the performance of *Minstrel* with the performance of fixed rate. The experiment has two parts, the downlink transmission experiment and uplink transmission experiment. Different factors such as static channel condition, dynamic channel condition and mobility, are considered to be experiment parameters. Our experiment proves the existence of the potential issue which is raised in Chapter 7 (Theoretical Study of *Minstrel*).

Chapter 9 **Conclusion** summarises the thesis, answers to the research questions, and provides a listing of the contributions, and directions for future work.

Chapter 2

Background

This chapter describes the background knowledge and feature of IEEE 802.11 wireless standard that are relevant to this thesis. IEEE 802.11 standards cover both the Medium Access Control (MAC) layer and the Physical Layer[22]. Section 2.1 provides basic information of network layers. Features of IEEE 802.11 networks are introduced in section 2.2. In section 2.3, an overview of rate adaptation algorithms is described.

2.1 Layers of Network

A layered architecture in computer networking provides a simpler, cheaper and more reliable solution for a large and complex system. The network layered system makes network more adaptable for changes in the implementation and prevent communication errors across the network. The data link layer is the second layer in the seven layers Open Systems Interconnection Basic Reference Model[23] (known as OSI model) as shown in Figure 2.1.

In IEEE 802 local area networks, the data link layer is described in more detail with Logical Link Control (LLC)[24] layer and Medium Access Control (MAC)[25] layer. The LLC protocol controls the network flows, frame synchronization, and error checking. It is able to work with all of the IEEE

802 MAC layers, such as IEEE 802.3[26][27] , IEEE 802.11[22], etc. The MAC layer handles communication with physical layer, it controls channel access which makes it possible for multiple nodes to communicate within a multiple access network that incorporates a shared medium.

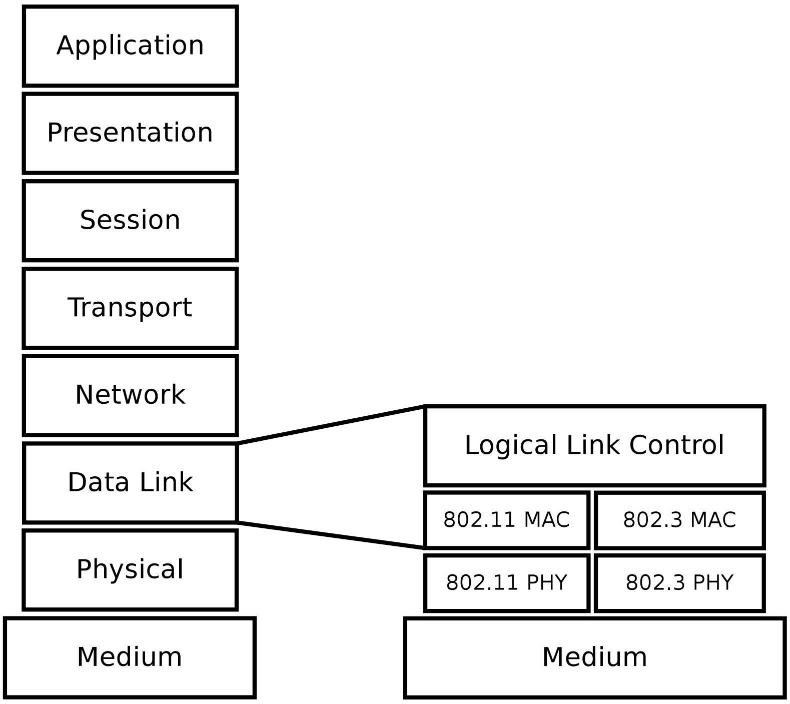


Figure 2.1: OSI 7 layer model and Data Link SubLayers

2.2 IEEE 802.11 Networks

IEEE 802.11[22] standards are also known as Wifi. The project was initialled and defined by the IEEE 802.11 working group[28] in 1990, and the first draft standard was released in 1997. Since 1997, IEEE 802.11 has had five subsequent amendments (See Table 2.1). The architecture of IEEE 802.11 networks has six main components: Station (STA), Wireless Access Point (AP), Independent Basic Service Set (IBSS), Basic Service Set (BSS), Distribution System (DS), and Extended Service Set (ESS), as shown in Figure 2.2. In this architecture, some of those components like STA and AP are mapped directly to hardware devices so that a STA can also be used as an AP. In an ESS wireless network, the smallest components are the STAs and APs. When multiple STAs connect to the same AP, the resulting network is called a BSS. An ESS is created by interconnecting the APs of multiple BSSs via the DS, as shown in Figure 2.2 on the left. ESS wireless networks are mostly used in an environment where the DSs are available. An IBSS wireless network is another type of 802.11 architectures. Unlike ESS wireless networks that are constrained by the availability of the DS. An IBSS wireless network consists of at least two STAs without any assisting from DSs, as shown in Figure 2.2 on the right. An IBSS network is more easily deployed in a low coverage of network infrastructures.

802.11 protocol	Bandwidth (MHz)	Data rate per stream (Mbit/s)	Modulation
-	20	1, 2	DSSS
a	20	6, 9, 12, 18, 24, 36, 48, 54	OFDM
b	20	5.5, 11	DSSS
g	20	6, 9, 12, 18, 24, 36, 48, 54	OFDM, DSSS
n	20	7.2, 14.4, 21.7, 28.9, 43.3, 57.8, 65, 72.2	OFDM
	40	15, 30, 45, 60, 90, 120, 135, 150	
ac	80	433, 867	QAM
	160	867, 1771, 3553, 7096	

Table 2.1: IEEE 802.11 Network Standard

IEEE 802.11 standards cover both physical layer and MAC layer; they also define how each packet is transmitted from one wireless node to another. For the purpose of this thesis, 802.11 physical layer and MAC layer are the most important background knowledge, and it is also important to understand the correlation between the wireless interference and these two layers.

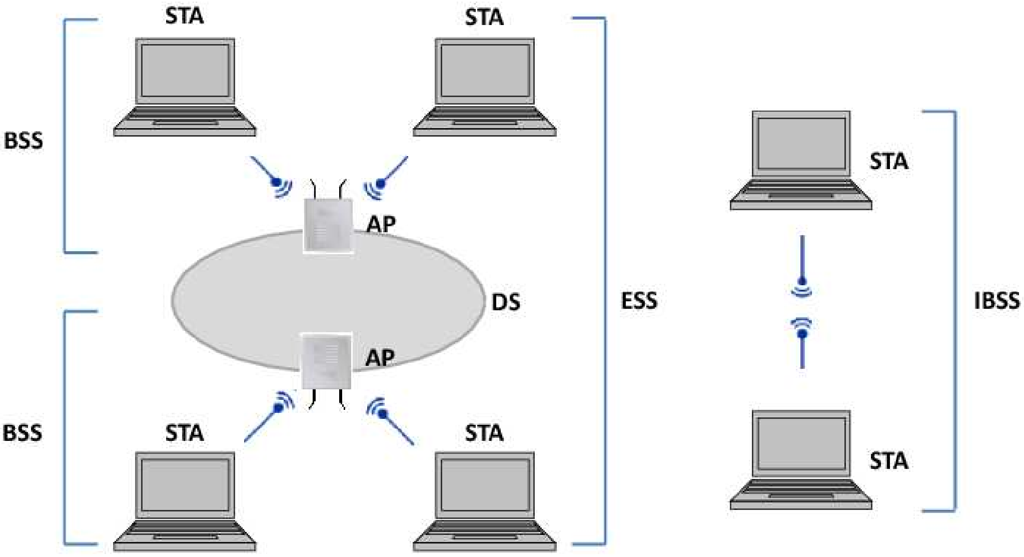


Figure 2.2: IEEE 802.11 Architecture

2.2.1 Wireless Interferences

Wireless interference is when a device transmits signals on a radio band, the transmission clashes to other radio bands on the airwaves. The transmission clash causes a receiver receives error signals. Wireless interference is a major issue to wireless communication, and it can be classified into three categories: wireless collisions[29], hidden terminals[30] and multipath fading[31].

- **Wireless Collisions** are caused by multiple devices waiting for the channel to become idle and simultaneously transmit signals. Wireless collisions could get worse if data frames were too large[32].

- **Hidden Terminal** is caused by a device that believes the channel is free, and starts to transmit without successfully detecting another transmission is already in progress.
- **Multipath fading** is caused by random amplitude and phase fluctuation at the receiver[33].

Wireless interference is highly dynamic and random. It causes the reliability issues in wireless networks. The reliable communication in wireless networks is often measured by the average bit error rate (BER). BER is the ratio of the number of errors in the total number of transferred bits during a time interval. In order to improve the reliability of IEEE 802.11 networks, mechanisms such as Forward Error Corrections (FEC)[34], Automatic Repeat Request (ARQ)[35], Multiple Data Rates, Rate Adaptation are used at both MAC layer and physical layer. The goal of FEC and ARQ mechanisms s the guarantee of packet delivery. They do not care neither the performance of the network nor the efficiency of the transmission. However, rate adaptation mechanism cares not only success of transmissions, but also performance of the network.

2.2.2 802.11 Physical Layer

In order to have a better understand of rate adaptation mechanism, it is important to understand the physical layer of IEEE 802.11. Modulation techniques and data coding are the most important parts of the 802.11 physical layer. The simple definition of modulation is to convert a digital information symbol into a continuous low frequency waveform, and this low frequency signal is carried over by a high frequency signal for its transmission. The bandwidth efficiency and implementation complexity are two major concerns for designing modulation schemes. The efficiency of bandwidth is directly impacted on the performance of wireless communication. Direct Sequence Spread Spectrum (DS/SS)[36] radios is a modulation technique. It is used by the original IEEE 802.11 standard. Orthogonal Frequency Division Multiplexing (OFDM) radios[37] is another modulation technique which used in 802.11a[38], 802.11g[39], 802.11n[40] and 802.11ac standards[41]. OFDM uses four different modulation schemas. They are Binary phase-shift keying (BPSK)[42], Quadrature phase-shift keying (QPSK)[43], 16 Quadrature amplitude modulation (QAM)[44] and 64 QAM modulation schema[44]. OFDM has much better bandwidth efficiency than DS/SS.

Figure 2.3 shows the theoretical throughput of BPSK, QPSK, 16 QAM, and 64 QAM modulation schemas in relation with signal-to-noise- ratio (S/N), assuming additive Gaussian white noise (AGWN)[1], based on IEEE 802.11g networks. These are four modulation schemas in order from the most bandwidth efficiency to least efficiency, with 1, 2, 4, 6 bits per symbol respectively. Different modulation schemas also have a S/N for which it is most efficient. They are resistant to different levels of S/N. For example, if the S/N was 10 dB, then only BPSK could be used for transmitting data. However, if the S/N was 20 dB, then 16 QAM could achieve the best performance.

Table 2.2 shows a summary of the modulations and data coding for the data rate used in IEEE 802.11g. Data coding is a mechanism that improves the reliability of wireless communications. It is mostly referred to error correcting code (ECC), and it is used primarily for FEC mechanism to protect data sent over in a poor quality wireless link. When a device transmits k bits of data, the k bits of data is encoded by ECC with a coding rate k/n, where n is the total transmitted bits after encoding for every k bits of data. The data rates are the product of multiplying the data coding rate, the number of bits per symbol, and the number of symbols per second. Figure 2.3 describes that different modulations are resistant to different levels of S/N, with the different data coding rates and modulation schemas, the lower data rates always have higher reliability and better resistance to the channel noise, and each data rate has its bandwidth efficiency.

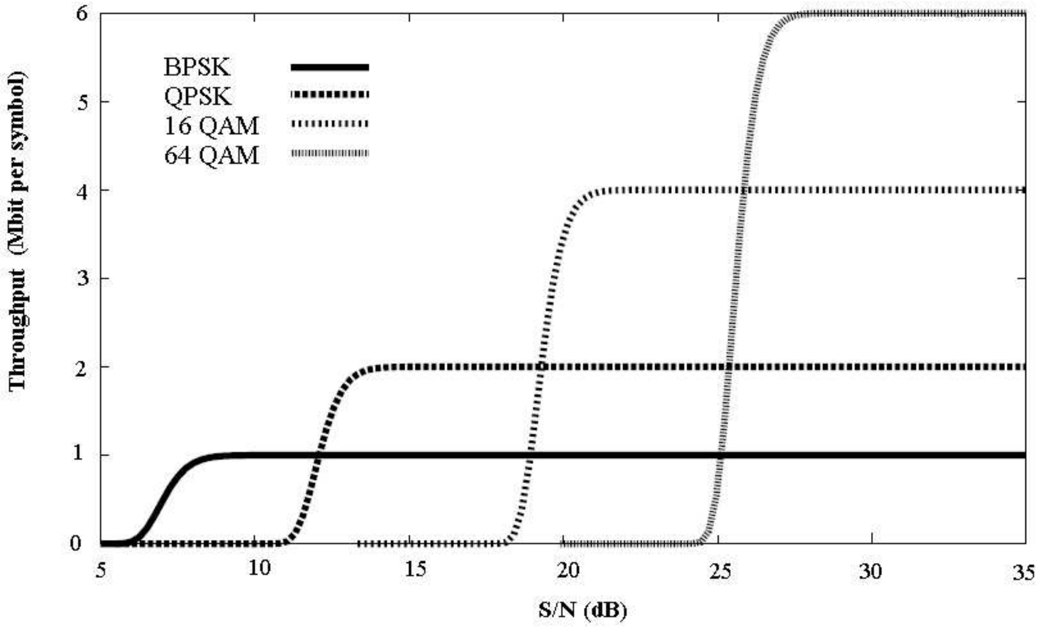


Figure 2.3: Theoretical BER versus S/N for modulation schemes[1]

2.2.3 802.11 MAC Layer

Medium Access Control (MAC)

Medium access control mechanism[45] is one of most important feature of 802.11 networks. It provides a physical-layer multi-rate capability[46]. It also prevents multiple nodes from accessing the same channel at the same time by using Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA)[47]. Every node starts listening on a desired channel to detect whether the channel is ideal or busy before it transmits a packet. If the channel is busy, the node waits for the channel until channel becomes idle then an additional back-off time (Distributed Inter-Frame Space (DIFS) period and a random time within a back-off window). If the channel is idle during the back-off time, the node transmits the packet and reset the back-off window size to minimum. Otherwise, it doubles the back-off win-

Spreading Method	OFDM							
Modulation	BPSK		QPSK		16 QAM		64 QAM	
Bits Per Symbol	1		2		4		6	
Mega-Symbols Per Second	12							
Mega Bits per Second	12	12	24	24	48	48	72	72
Data Coding Rate (k/n)	1/2	3/4	1/2	3/4	1/2	3/4	2/3	3/4
Data rate (Mbit/s)	6	9	12	18	24	36	48	54

Table 2.2: IEEE 802.11 g Data Rate

dow size, waits until the channel is idle and then a further new back-off time[48][49].

Forward Error Collection (FEC)

Forward Error Collection[34] is another mechanism to reduce the impacts on multipath fading. Section 2.2.2 has described that Error Correcting Code (ECC) is added at each sending packet. FEC mechanism uses ECC to detect a limited number of errors that occur in the receiving packet, and to recover the error packet without MAC layer retransmission. FEC improves the reliability of wireless communication and reduces numbers of retransmissions.

Automatic Repeat Request (ARQ)

Automatic Repeat Request[35] is also known as Automatic Repeat Query. ARQ is a MAC layer retransmission mechanism for reliable transmission in wireless network. It is a secondary error control method if FEC mechanism fails to recover the error packet. If a node receives a correct packet, the receiver node sends an ACK back to sender node. On the other hand, if a packet lost or FEC could not recover the packet, then the receiver node would send a NACK back to the sender node. The sender node retransmit the error packet.

Data Rate Adaptation

Section 2.2.2 explained different data rate has different bandwidth efficiency. In an IEEE 802.11 network, wireless communications suffer from a wide range of sources (see Section 2.2.1). The channel quality varies quickly from time to time. A single data rate cannot fit into all different levels of channel quality to have maximum bandwidth efficiency. Therefore, sender nodes have to adapt data rates. Data rate adaptation mechanism automatically selects a most efficient data rate according to wireless channel quality. Data rate adaptation mechanism is a key component in IEEE 802.11 networks.

2.3 Rate Adaptation Algorithm

Rate adaptation algorithms is one of most important mechanism in the data link layer to the performance of IEEE 802.11 networks. IEEE 802.11 specifications use different modulation and coding schema to mandate multiple transmission data rates at the physical layer. Rate adaptation algorithms value the condition of wireless channel and determine a data rate which performs the best. Therefore, a rate adaptation algorithm is an optimization problem that attempts to determine the optimal data rate for transmission based on the current condition of the wireless channel, in order to provide the best possible performance. The effectiveness of the rate adaptation algorithm in selecting optimal data rates depends on how accurately it can predict the channel conditions.

In this section, rate adaptation algorithm is classified into three major categories by the type of metrics used to predict channel conditions: ACK packet based, Signal-to-noise ratio based, and Bit Error Rate based mechanism.

ACK packet Based Approaches

ACK packet based rate adaptation algorithm is the most popular approach nowadays. It adapts data rates according to the result of success or failure of transmissions. Loss ratios and average transmission time for each data rates are recorded to predict which data rate is able to perform the best in the wireless condition. ACK packet based approaches do not require IEEE 802.11 standard modification. So these algorithms can be easily deployed in existing 802.11 networks. ARF[50], RRAA[51], AARF[52], AMRR[52], ONOE[53], SampleRate[54], Minstrel[55] and PID[56] are typical examples of ACK packet based rate adaptation algorithms.

Auto Rate Fallback (ARF) was the first rate adaptation algorithm to be published in 1997[50]. It has been adapted into open-loop rate adaptation schemes, such as [57], [58]. For the ARF algorithm, a sender node records the number of successfully and failure transmissions to determine whether the current data rate is an optimal rate. ARF increases its data rate and resets the timer when either the sender node receives 10 successfully ACK packets or the timer expires. On the other hands, ARF drops its data rate when the sender node has two consecutive fail transmissions or unsuccessful transmission right after the data rate increased. Some IEEE 802.11 devices are still using ARF algorithm because of its simplicity.

The ARF algorithm brings two major problems. It cannot adapt data rate fast enough if the channel conditions change quickly. Due to ARF requires 10 successful transmissions increases data rate, but only requires 1 or 2 fail transmissions to reduce data rate. The second problem is when channel conditions change very slowly. Fox example, ARF has selected an optimal data rate. However, it will keep trying to use a higher data rate every 10 successful transmissions. It increases many unnecessary attempts and reduces the throughputs.

Adaptive Auto Rate Fallback (AARF) algorithm was designed based on ARF. Author claims that it provides both short-term and long-term adaptation. AARF algorithm has a continuously changing threshold to reflect the channel conditions. The threshold is determined by using a Binary Exponential Backoff (BEB)[26]. The upper bound of the threshold is 50. The initial value of the threshold is 10. Data rate is switched back to the previous lower rate, and the threshold is set to two times of the number of consecutive successful transmissions when the probing packet is not able to transmit successful. When number of successful transmission equals to the number of threshold, the data rate is switched to a higher rate. If there is two consecutive failed transmission, then the threshold is reset to its initial value. The author claims that AARF algorithm improves the performance when the wireless channel conditions change very fast or very slow.

Adaptive Multi Rate Retry (AMRR) is the first algorithm that introduces retry chain concept into rate adaptation algorithm. A retry chain contains four pairs of data rate and retry counter ($r0/c0$, $r1/c1$, $r2/c2$, $r3/c3$). The retry counter is the maximum number of retransmission that allows a data rate to be used for transmitting a packet. The data rate $r3$ is always set to the lowest data rate. Data rate $r1$ and $r2$ are set to the immediately lower available rates after rate $r0$. Determining data rate $r0$ is based on the same set of ideas of AARF algorithm. It uses of a BEB to adapt the success threshold.

Signal-to-noise Ratio Based Approaches

RBAR[59], CHARM[60], and FARA[61] are typcial examples of SNR based approaches. SNR based approaches use the RTS/CTS mechanism[62] to calculate SNR to determine an optimal data rate. Selected data rate for each data frame is calculated at the receiver node, and sends back to the sender node via CTS control frame.

Receiver-based Auto Rate (RBAR) algorithm uses the RTS/CTS mechanism in the DCF of the IEEE 802.11 standard, it is also applicable to other RTS/CTS based protocols such as SRMA[63], MACA[64], MACAW[62] and FAMA[65]. The receiver node calculates the data rate to be used for the next incoming packet based on the SNR of the receiving RTS. The selected rate is sent back to sender node in the CTS packet.

The RBAR algorithm has three problems. The algorithm requires the RTS/CTS protocol, however most IEEE 802.11 networks turn the RTC/CTS protocol off by default. The algorithm also assumes that the SNR of each receiving packet is available at the receiver node. It is not always true. Some of IEEE 802.11 devices estimate the SNR by measuring the energy level from start of the reception of a packet and during the reception of the packet. RBAR cannot be deployed in existing 802.11 networks because it requires to interpret the RTS and CTS frames and the format of the data frames.

Channel-aware Rate Adaptation (CHARM) algorithm uses the Reciprocity Theorem[66] to predict wireless channel conditions. The algorithm has four components: information gathering, path loss predication, rate SNIR threshold estimation, and rate selection. CHARM algorithm adds new information in beacons, probe requests, and probe responses to inform other nodes about the transmission power level and noise level. At the receiver node, CHARM algorithm uses EWMA to predict path loss (equivalently SINR) at the receiver node. CHARM algorithm has a table that holds default values for the SINR threshold for the different data rates. The SINR threshold table gets update by observing the successful ratio of transmission. Before, a sender node sends a packet, it uses a SINR threshold table to determine a set of data rates. The algorithm also have the same problem as other SNR based rate adaptation algorithm. It requires wireless devices to turn the RTC/CTS mechanism on. Moreover, an existing wireless network has to be modified when the CHARM algorithm is deployed.

Frequency-aware Rate Adaptation (FARA) algorithm is similar to channel bonding[67] and load balancing[68][69]. The algorithm is a receiver based algorithm. It calculates the optimal data rates on each sub band, and feeds it to the sender node in ACK packets. Every ACK packet uses a 2-bit field to represent: stay on current data rate, move up/down to the next higher/lower data rate.

Bit Error Rate Based Approaches

BER based approaches is the least popular approach. It uses a more fine-grained metric to predict optimal data rate for each data frame. SoftRate[70] is the typical example of BER based approach. SoftRate Algorithm has three components: BER predication, computing optimal thresholds, and data rate selection. SoftRate uses the relationship between SNR and BER to predict optimal data rates. SoftRate calculates a range of BER for each data rate. If BER is at a range of BER for a data rate, then the data rate is selected for transmission.

2.4 Summary

This chapter has explained that wireless channel conditions dynamically change all the time. There are three major types of wireless interferences: wireless collisions, hidden terminal, and multipath fading. We have also explained key mechanisms (Medium Access Control (MAC), Forward Error Collection (FEC), and Automatic Repeat Request (ARQ)) that improve the reliability of wireless communications. Multi-data-rate is another important mechanism in IEEE 802.11 networks. It helps wireless devices working more efficient in different wireless channel conditions. However, wireless channel conditions change dynamically; rate adaptation algorithms help wireless devices to select a data rate which provides best possible throughputs by predict channel conditions. The chapter has also described three types of rate adaptation algorithms: (ACK Based Approaches, SNR Based Approaches, and BER based Approaches). Examples are provided to show how they work.

Chapter 3

Related Work

The topic of rate adaptation algorithm has been heavily researched [6][7][8] [9][10][11] [12][13][14][15][16][17] [18]. The rate adaptation algorithm itself can be classified into three categories: ACK packet based approaches, SNR based approaches, and BER based approaches. The study of rate adaptation algorithm is classified into three approaches by their research methodology: simulation based studies, emulation based studies and field based studies. The simulation based study uses a simulation program to observe operations in a simulated wireless environment. The emulation based studies uses physical devices to obtain experiment results in a controllable platform while performing real operations. The field based studies carry out in a real wireless environment with performing real operations.

SampleRate[54] is an algorithm that is frequently compared in these studies. However, none of these studies look at the performance of the newer *Minstrel* rate adaptation algorithm, except the work in [71]. Moreover, while the comparison among different rate adaptation algorithms is very useful, most of these studies do not directly compare against fixed rates. In spite of the fact that the comparison among of rate adaptation algorithms can point out which algorithm performs better, it cannot have a direct indication of how well these algorithms can adapt to an environment. Comparing rate adaptation against fixed data rate is an important step in further developments in the design of effective rate adaptation algorithms.

As this thesis aims to study the two most common rate adaptation algorithms, a survey and an analysis of the three research methodologies are presented in the following sections of the chapter. It should be noted that there is no literature or any evidence of a comprehensive study done on the comparison of the performances of the MikroTik and the *Minstrel* rate adaptation algorithm against the performance of fixed rate.

3.1 Simulation Based Studies

Sadeghi, B. [72] introduce the Opportunistic Auto Rate (OAR) protocol to enhance the performance of IEEE 802.11a[38] and 802.11b[73] media access protocols. It also compares the performance of OAR with RBAR[59] by using ns-2 simulator[74] in Ad Hoc Networks[75]. The Fast Fading model of Equation is used in their experiments. The Ricean Fading model is used to simulate a packet level of short time-scale fading[76]. Experimental procedure follows [49]. Three different topologies were taken in to consideration for this paper: Fully Connected Topologies, Asymmetric Topologies, and Random Topologies. For the Fully connected topologies, all nodes were within radio range of each other. The experiment shows that aggregate throughput of both algorithms increases as the number of flows increases. However, OAR has 51% more throughput gain than RBAR. In the asymmetric topologies, all nodes are not within transmission range, i.e., node A and B are connected, and Node B and C are connected, but Node A and C are not connected. Both OAR and RBAR achieve throughput gains approximately 60% and 35% respectively. In the random topologies, the aggregate throughput of OAR increases from less than 4 Mbps to 4.4 Mbps when the number of nodes increases from 10 to 40. On the other hand, the increase of the number of nodes does not have a significant impact on the aggregate throughput of RBAR. It remains at approximately around 2.8 Mbps. The paper is more focused on the topology of Ad Hoc network itself. It does not consider the rate adaptation algorithm performance impact on wireless channel conditions.

The work in [77] evaluates ARF[50] and CARA in five scenarios by using ns-2 simulator[74] with the Ricean fading model[76]. The first scenario is One-to-One Topology. In this topology, AP continuously transmits packets to the client. This experiment is focused on transmission distances. The result shows the performance of both algorithms decrease as the transmission distance increases. Moreover, CARA achieves comparable throughput with ARF. The second scenario is Star Topologies with varying number of contending station. The purpose of studying this scenario is for the understanding the collision impact on the performance of rate adaptation algorithms. The author notes that ARF cannot identify collisions, and the performance anomaly[78] is occurred. CARA significantly outperforms ARF in this scenario. The third scenario conducts Line Topology with random data frame sizes and random station positions. Once again, CARA performs approximately 20% to 80% better than ARF. The fourth scenario is Random Topology with time-varying wireless channel. In this experiment, author simulate 50 different scenarios with 10 contending stations. CARA performs significantly better than ARF. However, the result also shows that CARA also experiences lower aggregate throughput which is caused by its failure to detect collisions. The paper compares the performance of ARF and CARA, and it also considers many different factors such as number of contending stations, data frame sizes, and time-varying wireless channel. However, ARF was the first rate adaptation algorithm. It was invented in 1997. At that time, there was two data rates available to the users. Nowadays, new IEEE 802.11g/n standards have 8 data rates with much faster sending rates. Therefore, ARF is an out of date data rate adaptation algorithm. Moreover, the proposed testbed is based on IEEE 802.11b network, the standard is invented in 1999. Nowadays, most of the wireless networks are using IEEE 802.11g/n standards. Moreover, the work does not consider the performance impact on the dynamic changes of channel conditions.

Lacage, M.[52] propose an Adaptive ARF (AARF) rate adaptation algorithm to enhance the performance for low latency systems. In this paper, it compares the performance of ARF, RBAR and AARF. The simulation environment is described in [79], and it uses the ns-2 simulator[74]. The distances between the sender node and the receiver node is considered as a key experiment parameter. The results show that ARF has the lowest performance, and AARF with RTC/CTS enable has comparable performance results to RBAR. However, the main advantage of AARF is that it does not require RTC/CTS enabled. As a result, it can achieve much better performance than RBAR. The paper also presents another rate adaptation algorithm, AMRR. The AMRR rate adaptation algorithm is evaluated against the RBAR and one of the original Madwifi rate adaptation algorithms in a simulated environment and also within a real wireless environment. The transmission distance is taken as one of the key parameters for this experiment. The results show that the AMRR rate adaptation algorithm outperforms the Madwifi and RBAR. The transmission distance is one of the factors which caused the change in the wireless channel conditions. However, the author does not focus on evaluating the other factors (such as the transmission power, dynamic channel conditions) which also have a significant impact on the wireless channel conditions.

3.2 Field Based Studies

The work in [80] propose a rate adaptation algorithm called Robust Rate Adaptation Algorithm (RRAA) which was compared with SampleRate, AARF, and ARF in a real IEEE 802.11a/b network environment with various settings, such as static/mobile clients, with/without hidden stations. In order to conduct repeatable experiments, all experiments in the work are done during midnight. The experiment conducts a stationary server node, and a constant moving client node during the experiment. The purpose of this experiment is to evaluate how rate adaptation algorithms response to the dynamics of the wireless channel conditions. The channel conditions either get deteriorated when client nodes move away from the server node, or get improved when client nodes move closer to the server node. In reality, the wireless channel conditions are constantly changing between good and bad. For our experiments, we have set our channel conditions to be highly dynamic to fit the real wireless environment.

Ancillotti, E. [81] have evaluated three different rate adaptation algorithms including SampleRate. There is some comparison done between rate adaptation algorithms and fixed rates. Yet, the focus of the paper is on the correlation between the number of senders and the rate adaptation algorithms in wireless mesh networks. The evaluation did however show in a particular scenario that the performance of some of the fixed rates was better than the evaluated rate adaptation algorithms. The work in [29] proposed Wireless Congestion Optimized Fallback (WOOF), a rate adaptation algorithm that attempts to identify congestion related packet losses. Along with other solutions including SampleRate, the scheme was to evaluate against the factors such as network load and the number of clients. There are also many other solutions which use SampleRate for performance evaluation, however, the newer *Minstrel* algorithm is seldom used for comparison.

3.3 Emulation Based Studies

Camp et.al[71] use an emulation platform (Wireless Open-Access Research Platform (WARP)) to evaluate three rate adaptation algorithms (ARF, RRAA, RBAR) and two of their own extended rate adaptation algorithms (RRAA with the A-RTS extension, RBAR with the OAR extension). OAR is presented in [72]. All experiments are conducted with a Spirent Communication Channel Emulator (SR5500). The purpose of having an emulator is for simulating different wireless channel conditions. The device is able to manually adding interferences to a single channel with a time interval (from 0.1 ms to 100 ms). The maximum average interface noise is -40 dBm. However, all emulation experiments are based on three out of date rate adaptation algorithms and 2 extended algorithm. The contribution to nowadays wireless networks is limited due to these factors.

The most relevant work done is in [82][2], as it is the only previous evaluation of *Minstrel* to my knowledge. This work evaluates four rate adaptation algorithms found in the MadWifi driver, which are ONOE, Adaptive Multi Rate Retry (AMRR), SampleRate and *Minstrel*. The evaluation uses a wired testbed with co-axial cables[83] to emulate wireless channels. This is a clever solution because it allows the algorithms to be evaluated in an environment that is free of interference, and consequently is also able to produce repeatable results. This evaluation shows that *Minstrel* performs far better than the other algorithms. Figure 3.1 and Figure 3.2 show the throughput performance comparison between four rate adaptation algorithms from the paper [2]. In figure 3.1, the author compares the performance of these four rate adaptation algorithms against the path loss. Once the path loss was set, the wireless channel condition remained the same. According to author's experiment, it is easier to see that *Minstrel* rate adaptation algorithm and SampleRate have reasonably good performance in a static channel. In figure 3.2, the paper evaluates the impact of interference duration on these four rate adaptation algorithms. In this experiment, the wireless channel dynamically changes. *Minstrel* rate adaptation algorithm has much better performance than SampleRate and other two algorithms when channel conditions keep changing.

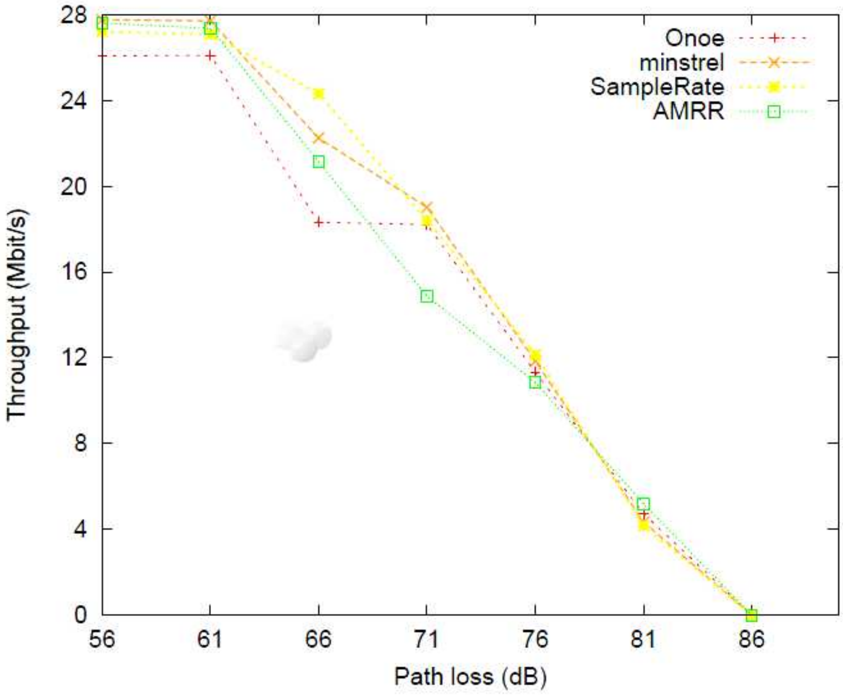


Figure 3.1: Rate Adaptation Performance vs path loss (received signal strength)[2]

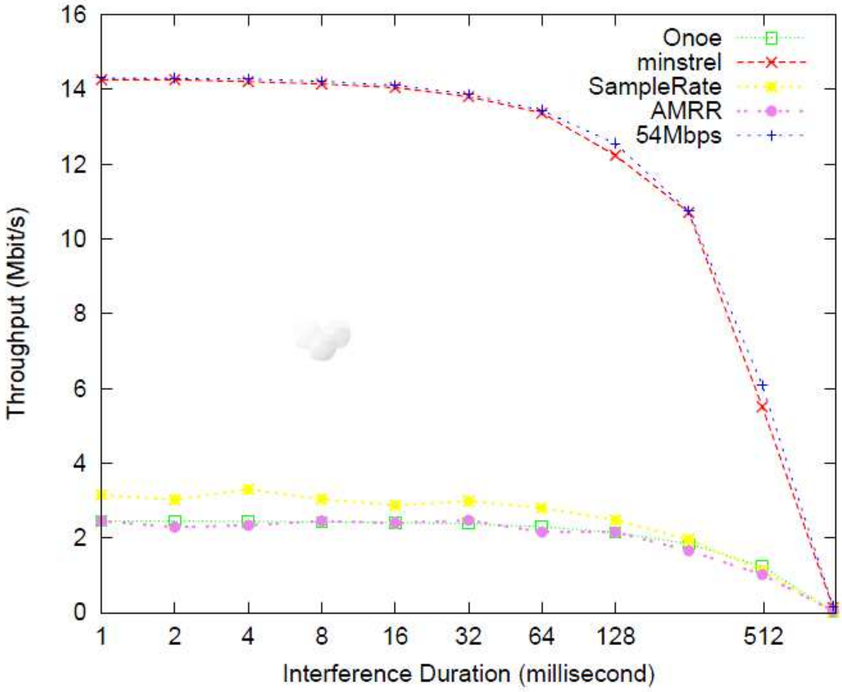


Figure 3.2: Rate Adaptation Performance vs Interference duration[2]

3.4 Summary

Despite the popularity and apparent superiority of *Minstrel* over other older rate adaptation algorithms, there has been very little study of *Minstrel* in the literature. In addition, little work in the literature has been done on direct comparisons of rate adaptation algorithms with fixed rates. These factors motivated us to compare the performance of *Minstrel* against fixed rates in a real world 802.11 testbed. Therefore, other schemes such as SampleRate are not included in this thesis. However, the work does not show how *Minstrel* performs against fixed rate, and the wired testbed cannot fully emulate the real-world characteristics.

Chapter 4

Experimental Methodology

This chapter describes three related experiment methodologies and it explains in detail what method is used in this study. Section 4.1 gives an overview of this study. Section 4.2 highlights the characteristics of each methods. Section 4.3 is summary of the chapter, it explains the methodology we used in this study, and the reason behind our choice.

4.1 Overview of Study

MAC layer rate adaptation algorithm is one of the key components of IEEE 802.11 protocol. The performance of the protocol mainly depends on how well rate adaptation algorithm can adapt according to the dynamic wireless environment. The dynamic changes of a wireless environment are caused by a number of factors, such as interference from other wireless devices, multi-path fading and signal attenuation. MikroTik rate adaptation algorithm and *Minstrel* rate adaptation algorithm are two of commonly used rate adaptation algorithms, nowadays. The purpose of this study was to fulfil and understand the performance and potential issues of MikroTik rate adaptation algorithm and Minstrel rate adaptation algorithm.

4.2 Experiment Methods

There are three existing experiment methods which are relevant to this study; they are Simulation Based Study, Emulation Based Study and Field Experiment Based Study. Each method has its unique characteristics.

4.2.1 Simulation Based Study

A simulation based study uses simulation program to observe operations without performing the real operations in a real environment. A simulation based study provides a user a totally controlled environment. In a simulated environment, user is able to adjust any of the individual variables and observe the impact of changing a variable. Moreover, a simulation based study provides user repeatable experiment results. Compared to the two othe methods, it is the most least time consuming methodology.

On the other hand, a simulation program replicates an environment by taking users inputs; and these qualities of inputs determine the similarity of the environment. However, having high quality of inputs are still not enough to fully replicate a real environment, especially a complex environment such as wireless channel. Because of this weakness, a simulation based study has two main disadvantages: Artificiality, and Limited Experiment Behaviour.

4.2.2 Emulation Based Study

An emulation based study uses physical tools to obtain a controlled environment with performing real operations. An example which is relevant to this study is paper[84][82][2]. The author uses a wired testbed to emulate wireless channel conditions. Except for the emulated experiment environment, the reset of experiment setting works no different to a field based experiment study. Emulation based study reduces the artificiality of the experiment by observing real operations. However, the emulated environment is also depended on the user's input variables. Experimental results still cannot fully represent the situations which happen in a real environment.

4.2.3 Field Based Study

A field based study uses physical devices to perform real options in a real world environment. It cannot have a tight control of the environment, however it provides a user the advantage of being far less artificial than the other two experimental methodologies. Because of the loose environmental control, the experiment result can be containing a lot of different experimental behaviour. Yet, to understand and analyse these experimental behaviours can be a challenging job. It requires user to repeat the same experiment many times to get one final summarised conclusion. It is the most time consuming experiment methodology compared than the other two methods.

4.3 Proposed Methodology - Field Based Study

Three of these experimental methodologies have their advantages and disadvantages in some ways (Table 4.1). Simulation based study and emulation based study provides user controllable testing environment and short experiment cycle. The complexity of the wireless channel conditions being many challenges to the design and implementation of a rate adaptation algorithm. Even though, simulation programs and emulation approach makes it difficult to replicate a complex wireless environment, these two methodologies takes away the complexity to explore the real causes behind the experimental outputs. In order to produce a comprehensive experiment study, it is important to have results that are able to cover most of the experimental behaviours. A field based experiment has its natural advantage of covering most of the behaviours. For the purpose of this thesis,

	Simulation Based Study	Emulation Based Study	Field Based Study
Artificiality	High	Medium	Low
Limited Behaviour	High	High	Low
Time Consuming	Low	Medium	High
Devices Cost	Low	High	Medium
Repeatable Experiment	Yes	Yes	No

Table 4.1: The Comparison of Experimental Methods

- **Artificiality** - The experiment results are not typical of real life situations. Most experiments are not able to fully conduct a complex situation such as a complex wireless environment. The complexity of a simulated environment is depended on user input settings.
- **Limited Experiment Behaviour** - Behaviour in the simulation and emulation based study is very narrow in its range. By controlling the situation so precisely, behaviour may be very limited.

we have chosen a field based approach. The main downside of conducting field based study is the difficult of obtaining respectable experimental results. To resolve the issue, all the experiments in this thesis study run 20 times, and each run takes 60 seconds to complete. The average result and stand deviation is calculated. The multiple runs reduced the side effect of field based experiment. In this research, the source code of *Minstrel* rate adaptation algorithm is also studied in detail which enhanced the theoretical understanding.

Chapter 5

Experiment Design

This chapter focuses on describing the design of experiment procedure, experiment measurement, and the implementation of experiment automation controlling system. Section 5.1 describes the testbed platform. Section 5.2 describes the measurement methods and tools used in this study. Section 5.3 lists factors that can interfere experiment results, and provides two solutions which can reduce interferences of getting accurate results.

5.1 Testbed Platform

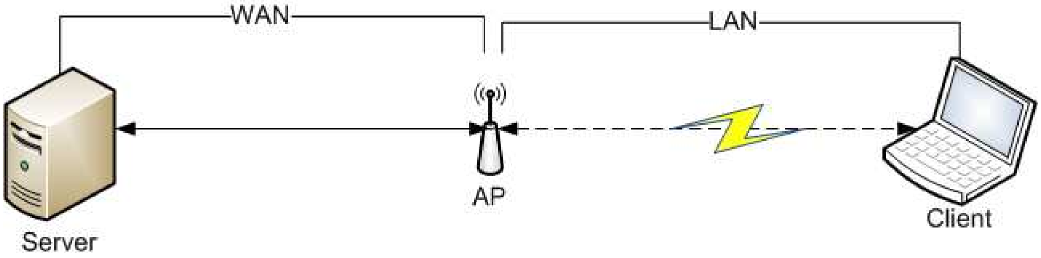


Figure 5.1: Testbed network topology

Figure 5.1 shows the network topology of the testbed. Wireless cards used in this study containing the Atheros AR5414 2.4/5 GHz chipset[85] in the AP and the laptop (Client). The testbed used 802.11g standard and was set to use channel 11, which is centred at 2.462 GHz frequency[86]. 802.11g protocol provides eight data rate options (Table 2.1). The AP is based on a MikroTike RB411AH router and runs the MikroTik routerOS 5.0rc10 operating system in the first experiment (Section 6) and the Linux-based OpenWrt[87] backfire[88] operating system in the second experiment (Section 8). The client PC is a Dell Latitude E6510 series Laptop. Both the AP and the laptop use the *Minstrel* rate adaptation algorithm, and the default transmission power is 17 dBm. The server PC is a Dell Optiplex GX620 desktop. Both the desktop and laptop run Ubuntu 10.04 which uses version 2.6.23 of the Linux kernel.

This real-world evaluation is conducted in a teaching and office building on a university campus. Figure 5.2 shows the floor plan of the campus building. The campus building is constructed of a mix of wooden, concrete and glass walls. The dark/blue lines in Figure 4.2 represent glass walls. Spot *CAP* is the location of a Campus Access Point which is out of our control for the experiments. Spot *R1* and *R2* in the figure on the left represent the two locations where the receiver node is placed. The distance from *AP* to *R1* and to *R2* is 1 meter and 20 meters respectively. The reasons for choosing the two locations are explained in chapter 6. The figure on the right is the floor plan of *Minstrel* field study (Chapter 8). Spot *R1* and *R2* represent the two locations where the laptop is placed. In this experiment, the laptop acted as a receiver node and a sender node.

5.2 Measurement Methods

Performing a wireless measurement is a difficult and challenging task. The quality of the performance measurement depends on the accuracy of experiment measurement and the quality of collected experiment data.

Active measurement method is used for this study. The idea of active

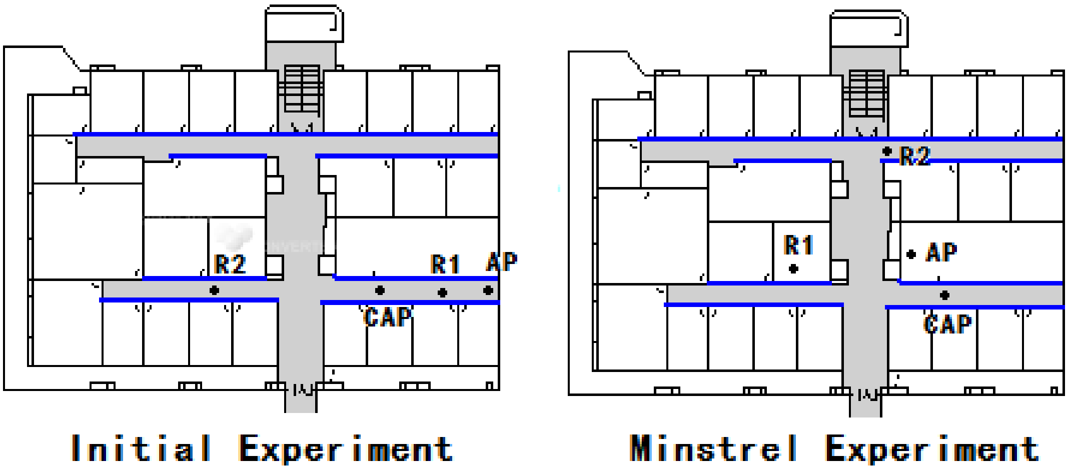


Figure 5.2: Testbed floor plan

measurement is injecting known packets into the network and analysing the packets at the end of receiver node. *iperf*[89] is an end-to-end active network performance measurement tool written in C++, which is able to generate UDP and TCP traffic to measure the throughput between a client and server. It allows the user to set various parameters that can be used for testing a network. The results are not sufficient for rate adaptation algorithm studies, as the experiment results from *iperf* does not provide any wireless link layer information such as the data rate is selected by rate adaptation algorithm or the number of attempts for each data rates is used to transmit data. In order to improve the quality of collected experiment data, statistics about the link layer must be obtained.

The second measurement method used in this study is called ”ratestats”. It uses different measurement concept to an active measurement. Instead of inspecting packets at the end of receiver node, target information is able to be viewed at the sender node as data packets are transmitted. This is achieved by collecting low overhead data from the rate adaptation algorithm sending process. The captured data is independent from the rate adaptation algorithm that can be used to analyse performance. Figure 5.3 shows the components of the ath5k wireless driver. At the bottom of the diagram is the physical wireless card. In this study, Atheros AR5414 chipset was used. On the top of the physical wireless card is the Atheros HAL called OpenHAL[90], the following layer is ath5k wireless driver for Atheros wireless cards, and the next layer is rate adaptation algorithms . These components are implemented in the kernel space which is located at the middle in the diagram. At the top of the diagrams the file system which is accessible from user space. The *Minstrel* rate adaptation algorithm output statistics are stored in ratestate in the file system. However, this method requires kernel access. In the first experiment, we are not able to get ratestate information from kernel because the MikroTik routerOS is a close source operating system.

The combination of active measurement method and ”ratestats” measurement method, the collected data is able to be coarse-grained and fine-grained analysed, so that the content of collected data is much richer which helps improving the quality of the study.

5.3 Data Collection

The quality of collected experiment data can also be influenced by many factors. In this section, we describe how data is collected with minimum interferences by many factors.

Start Time	Monday	Tuesday	Wednesday	Thursday	Friday
8:00 am	Test #1	Test #5	Test #9	Test #13	Test #17
10:00 am	Test #3	Test #7	Test #11	Test #15	Test #19
1:00 pm	Test #2	Test #6	Test #10	Test #14	Test #18
3:00 pm	Test #4	Test #4	Test #12	Test #16	Test #20

Table 5.1: Experiment Time Routine

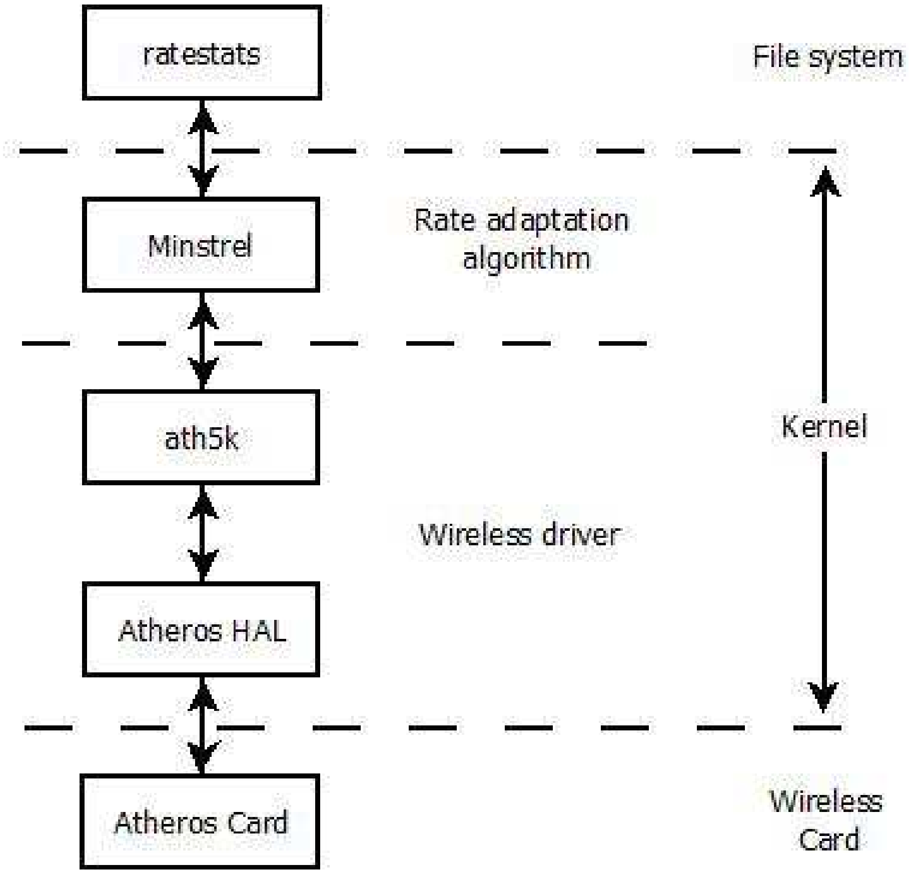


Figure 5.3: Visualise Component of the ath5k wireless drive

One of key interfering factors is caused by the dynamic of wireless channel conditions. All the experiments in this study were carried out in a real environment. The wireless channel conditions dynamically change from time to time. It causes a major concern about the experiment repeatability. To perform an accurate comparable and repeatable experimental study of a wireless link, multiple tests must be obtained to reduce the impact on the external interference. In this thesis study, each experiment repeatedly runs for 20 times and each run takes for 60 seconds to complete. In addition, all the experiments strictly followed a time routine (see Table 5.1). By following the routine improved the comparability among the different set of experiment.

5.4 Automation Controller

It can be a challenge to precisely and accurately run all the tests and to simultaneously give instructions to all the devices. It is important to have an automation controller that is able to precisely control and manage all the devices, and to collect necessary data with the minimum interference to experiment results. An automation program can minimize human error, however it also brings two major internal interferences. The source of two major internal influence factors come from the automation controller itself, and transmitting controlling massages between experiment devices.

In a wired experiment, the overhead of controlling messages can be relatively small to the capacity of wired link bandwidth. However wireless transmission has much larger transmission overhead because of high packet loss ratio, especially when bad channel conditions take place during the transmission. The overhead of controlling messages is relatively large to wireless networks. Transmitting any controlling messages can have huge impact on the quality of collected data no matter what measurement methods used in the experiment. Therefore, transmitting controlling messages is prohibited during the experiments. In this study, the testbed automation controller is carefully designed to resolve the impact on transmitting controlling messages. The design of the testbed automation controller will be explained in the later section.

Many programming languages can be used for controlling and managing these experiment devices, such as Java, C#, etc. To perform an accurate evaluation of rate adaptation algorithm, the programming language should be light weight on process power consumption, so that it will not cause additional interference to the experiment. However, Java, C# and other high level programming languages are too heavy for this experiment study which can potentially interfere the experiment results. As the reason above, the solution was to use a script programming language to implement the automation controller. Bash scripting[91] was the choice for controlling measurement applications and collecting experiment data due to its light weight. To be able to remotely control all the devices, bash script alone is not enough to satisfy the needs of the study. Expect scripting[92] is another choice to remotely controlling all the devices for this study. With the combination of bash and expect scripting, we are able to simultaneously control and manage experiment devices.

Design of Testbed Automation Controller

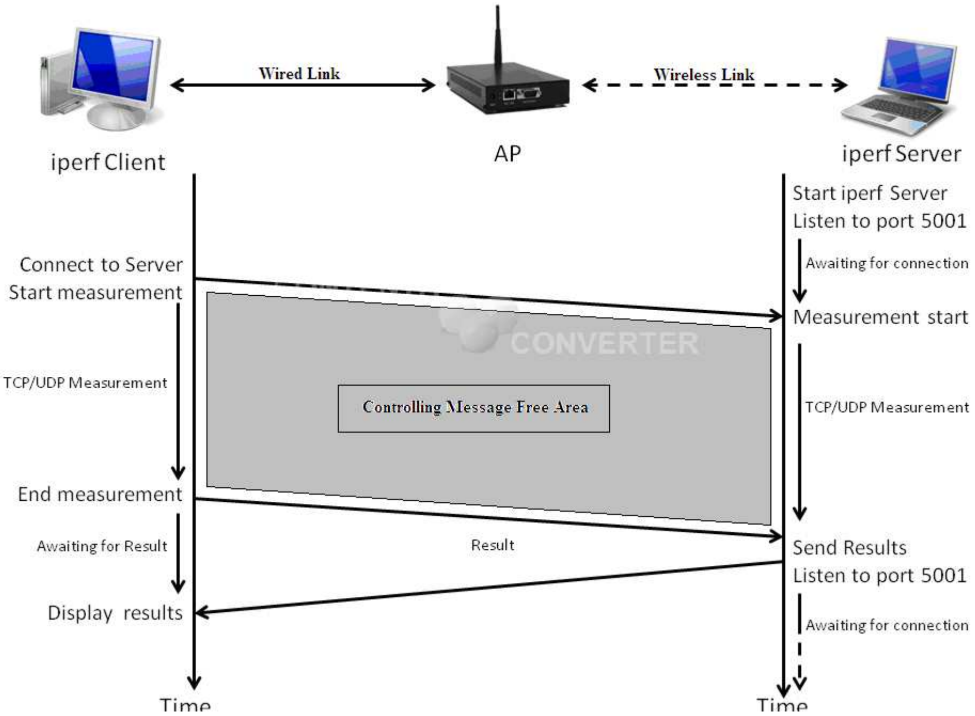


Figure 5.4: *iperf* Working Procedure

In this thesis study, *iperf* is the choice for the active measurement method. *iperf* is an end-to-end network performance measurement tool written in

C++. It is made up by *iperf* server and *iperf* client. Figure 5.4 shows an example of *iperf* working procedure. To run *iperf* measurement, the *iperf* server should start running first by executing the following: *iperf -s*. It allows server to listen on TCP or UDP port 5001. After the *iperf* server started running, *iperf* client can be executed by executing the following: *iperf -c*. The performance measurement starts once the *iperf* client sends the first packet, and the *iperf* server receives the packet. After the *iperf* server receives the last packet from the client, it calculates the throughput, packet loss ratio (UDP experiment only), etc. The result is sent back to the client. *iperf* users are able to set TCP window size, UDP send rate, the duration of transmission and etc.

In order to reduce experiment interference from internal interruption, experiment controlling messages do not allow to be transmitted between AP and the receiver node during experiment measurement period which shows grey area in Figure5.4. Therefore, all the controlling messages have to be transmitted before experiments start or after experiments finish.

Testbed automation controller has four key components, User API, Experiment Initiation, Experiment Execution and Data Collection. Figure 5.5 shows four key controller components and six sub-components. Four key controller components are executed on a computer which has a wired link to the AP, due to larger bandwidth on wired link which means less likely interfere to the experiment. API is a central command center of automation controller program. It provides a GUI interface that allows user to select different experiment parameters such as AP transmission power, duration of experiment, etc. In addition, API also instructs other components running on sender node or receiver node. However, these controlling messages can be key internal interference sources. Therefore, other six sub-components are executed in the sender node or the receiver node at outside of experiment measurement period which has indicated in Figure 5.4. In the entire process, there are few important steps which are required to pay extra attention; they are Experiment Processors Termination, and Data Transfer. In order to have an accurate experiment result, it is important to keep all the internal experiment condition same. Therefore, Experiment Processors Termination is executed before and after experiment to ensure that the experiment result will not be interfered by previous experiment. Data Transfer is another important step in the controlling process. It transfers recorded data back to the main PC for further analysing. However, data transfer can occupy a huge amount of bandwidth which causes interference to the experiment. Therefore, transferring collected data in between experiment measurement is prohibited.

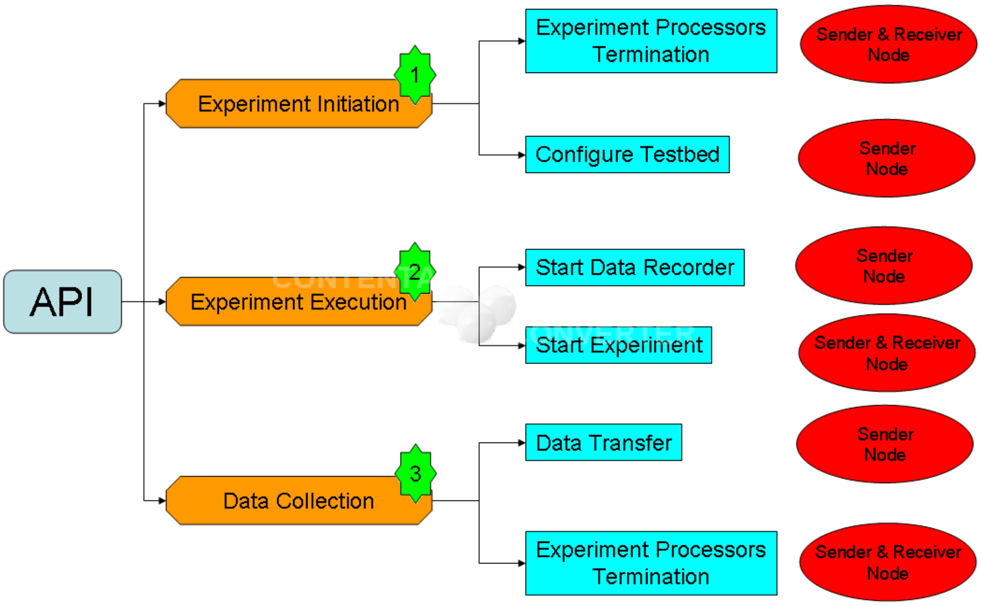


Figure 5.5: Automation Controller Working Procedure

5.5 Summary

Evaluating rate adaptation algorithm in a real-world environment is always a challenge to many researchers. Experiment results can be easily corrupted by surrounding environment and experiment tools. To reduce interference from internally and externally, we carefully select our experiment tools and implement a controller program to manage all experiment devices. In order to have a conclusive and repeatable result, multiple testes are preceded. In addition, all experiments strictly follow a time routine.

Chapter 6

Mikrotik Experiment Study

MikroTik wireless router is widely used in New Zealand and around the world. Many organisations and individuals use the operating system provided by MikroTik. Integrated with the operating system is MikroTik's proprietary rate adaptation algorithm. The router uses rate adaptation as the default setting. Considering the popularity of the Mikrotik wireless router, findings in this study can provide useful information to the user on the performance MikroTik rate adaptation algorithm and the design of rate adaptation algorithm. This section shows the performance comparison between Mikrotik rate adaptation algorithm and fixed rate with UDP and TCP traffic. Our published paper [19] is based on this experiment. The focus of the experiment was on four factors that can have an impact on the performance of the rate adaptation and fixed rate: UDP sending rate, AP transmission power level, AP/client distance and RTT for TCP traffic. Section 6.2 shows how experiment factors were selected in this experiment.

6.1 MikroTik Rate Adaptation Algorithm

MikroTik does not provide any specifications about the algorithm. After consulting with MikroTik technical support, it was noted that the data rate is increased or decreased depends upon the transmission failure or success. Three sequential transmission failures decrease the data rate until it reaches the lowest rate. If the three failures are already on lowest rate, then the transmission for these destination is suspended until it runs out of on-fail-retry-time[93]. On the other hand, three sequential successful transmissions increase the data rate until it reaches the highest rate.

6.2 Experiment Factors

UDP sending rate, AP power level and distances between AP and client are considered as key experiment factors in this experiment. In this section, we conduct few initial tests to select the parameters of the experiment factors. The reason of selecting these parameters are explained in this section.

6.2.1 UDP Sending Rate

UDP traffic does not respond to packet losses. There are two types of packet losses: buffer overflow and wireless corruption. A high UDP sending rate is used to investigate the scenarios that tend to have significant buffer overflow packet losses at the AP. A low UDP sending rate is used to investigate the scenarios that would not have significant buffer overflow losses. High/Low UDP sending rates are defined below (High/Low are relative terms):

- High UDP Sending Rates Sending rates equal data rate (e.g. if the data rate is 54Mbps, the UDP sending rate should be set to 54Mbps).

- Low UDP Sending Rates Sending rates equal 2/5 of data rate (e.g. if the data rate is 36 Mbps, the UDP sending rate should be set to 14 Mbps). The reason why the sending rate is set to 2/5 of data rate is explained in Table 6.1

Data Rate	Packet Loss Ratio			
	High Sending Rate		Low Sending Rate	
	1 M	20 M	1 M	20 M
54 Mbps	54.97%	55.71%	6.41%	1.83%
48 Mbps	55.12%	53.59%	4.91%	4.57%
36 Mbps	47.91%	50.87%	3%	2.19%
24 Mbps	40.19%	45.37%	0.02%	0.75%
18 Mbps	34.31%	32.8%	0.32%	0.1%
12 Mbps	27.99%	27.84%	0.01%	0%
9 Mbps	27.44%	25.33%	0.03%	0%
6 Mbps	19.73%	23.83%	0%	0%

Table 6.1: IEEE 802.11 g Data Rate

Table 6.1 shows the average percentage of packet losses with High-/Low UDP sending rates using fixed rates. The client node is placed at R1 and R2 (Figure 5.2) with the APs power level set to 17 dBm. In the table, the highest packet loss ratio is 56.97% (less than 60%). Therefore, we expect that a sending rate at 40% (2/5) of data rate would have substantially reduced buffer overflow packet losses. Indeed, the table shows that Low UDP sending rate has very low packet loss ratio while High UDP sending rate creates a lot of buffer overflow packet losses at the AP. Based on this observation we conclude that High UDP sending rate tries to fully utilise channel capacity while Low UDP sending rate tries to utilise channel capacity without creating a lot of buffer overflow losses.

6.2.2 Power Level and AP/Client Distance

In this experiment, a combination of different transmission power levels and AP/Client distances were used to emulate bad/good channel conditions. There were two transmission power levels were selected: High power level (17 dBm) and Low power level (8 dBm). 17 dBm is the default transmission power.

Based on the two power levels, we did some initial tests to help select the locations to place the client node. We did tests with AP/client distance at 1 meter, 5 meters, 10 meters, 15 meters and 20 meters. Figure 6.1 and 6.2 show channel utilisation of fixed rate with High UDP sending rate on each of the eight data rates. In Figure 6.1 the power level is set to 8dBm while in Figure 6.2 the power level is set to 17dBm. Figure 6.1 clearly shows that there is no clear performance difference between AP/client distance at 1m, 5m, 10m and 15m. However, at 20 meters the performance is substantially dropped. In Figure 6.2, there is even no clear performance difference between AP/client distance at 1m and 20m. With power level at 8dBm, it was also observed that the client node had frequent connection losses, when placed beyond 20 meters away. Therefore, we choose locations R1 (1m) and R2 (20m) to place the client node for the experiments.

Table 6.2 lists the experiments with UDP traffic. The experiments are split into two parts: High UDP Sending Rates experiments and Low UDP Sending Rates experiments, with a combination of power levels (17/8 dBm) and AP/client distance (R1/R2). Table 6.3 lists the experiments with TCP traffic.

	AP Power	Sending Rate	Client Location
Exp1	8 dBm	High	R1
Exp2	8 dBm	High	R2
Exp3	17 dBm	High	R1
Exp4	17 dBm	High	R2
Exp5	8 dBm	Low	R1
Exp6	8 dBm	Low	R2
Exp7	17 dBm	Low	R1
Exp8	17 dBm	Low	R2

Table 6.2: UDP Experiments

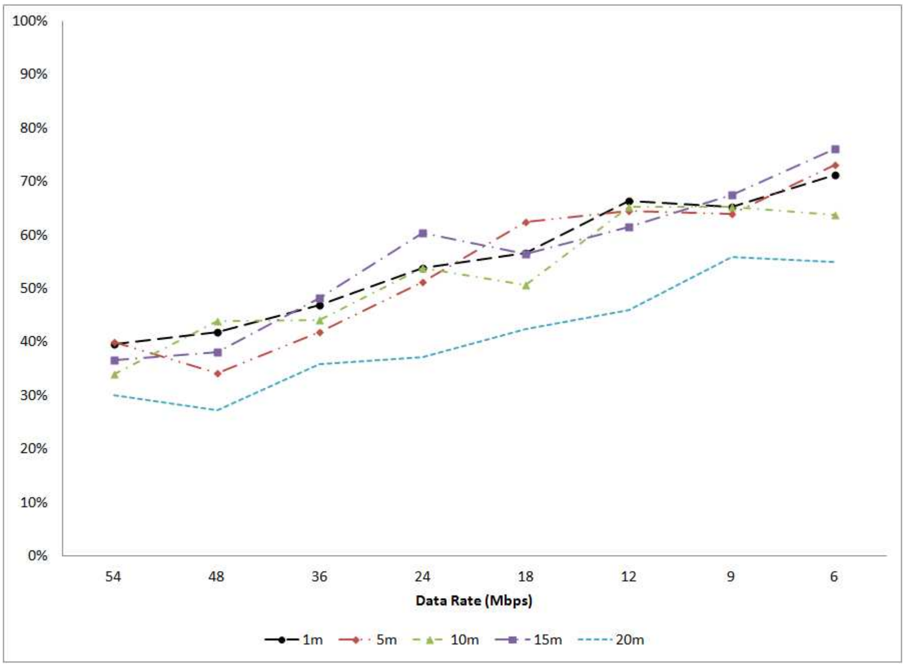


Figure 6.1: Channel Utilisation (8dBm, High UDP)

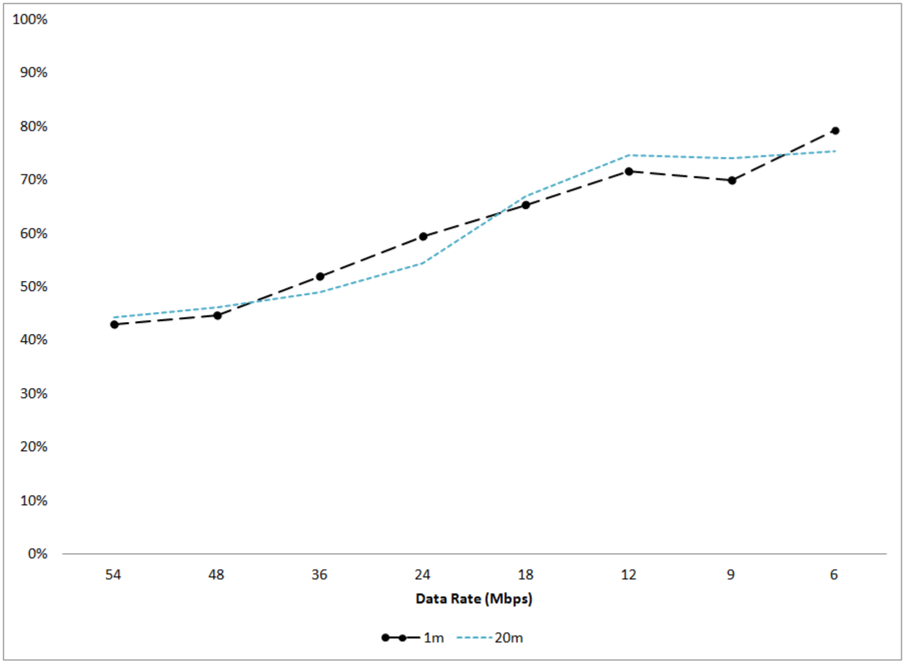


Figure 6.2: Channel Utilisation (17dBm, High UDP)

	AP Power	RTT (ms)	Client Location
Exp1	8 dBm	10, 50, 100, 200	R1
Exp2	8 dBm	10, 50, 100, 200	R2
Exp3	17 dBm	10, 50, 100, 200	R1
Exp4	17 dBm	10, 50, 100, 200	R2

Table 6.3: TCP Experiments

6.3 UDP - High UDP Sending Rate

6.3.1 Average Throughput

Figure 6.3 and 6.4 compares the throughput performance of fixed rate and Mikrotik rate adaptation, based on experiments Exp 1 and Exp 3. It shows that fixed rate outperforms Mikrotik rate adaptation as data rate increases. The rate adaptation performs slightly better than fixed rate when the data rate used by fixed rate is low. This is understandable - rate adaptation has the freedom to use high data rates while fixed rate can only use the selected data rate. However, when fixed rate is not limited by low data rates, it performs significantly better than rate adaptation. The results from experiments Exp 2 and Exp 4 show the similar performance.

From Figure 6.3 and 6.4, it has also been noticed that the standard deviation of Mikrotik rate adaptation with High power level (Figure 6.4) is much higher than the one with low power level (Figure 6.3). The cause could be due to the interference from a nearby AP. The campus has an AP at spot CAP as shown in Figure . The AP at CAP adjusts its power level dynamically according to the surrounding noise levels. When the Mikrotik wireless router increased the power level, the CAP AP also increased its power level. As a result, there might be greater interference between the APs.

Figure 6.5 shows the throughput ratio between the Mikrotik rate adaptation and fixed rates with low and high power levels at location R1. Again,

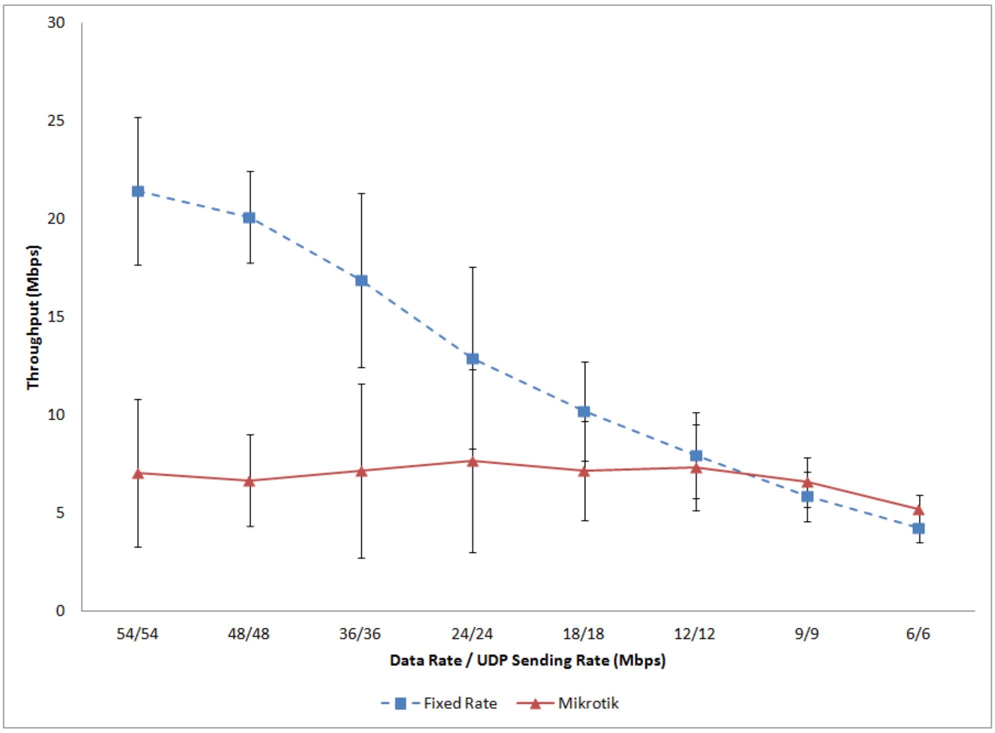


Figure 6.3: Throughput performance - power level: 8 dBm, client location: R1

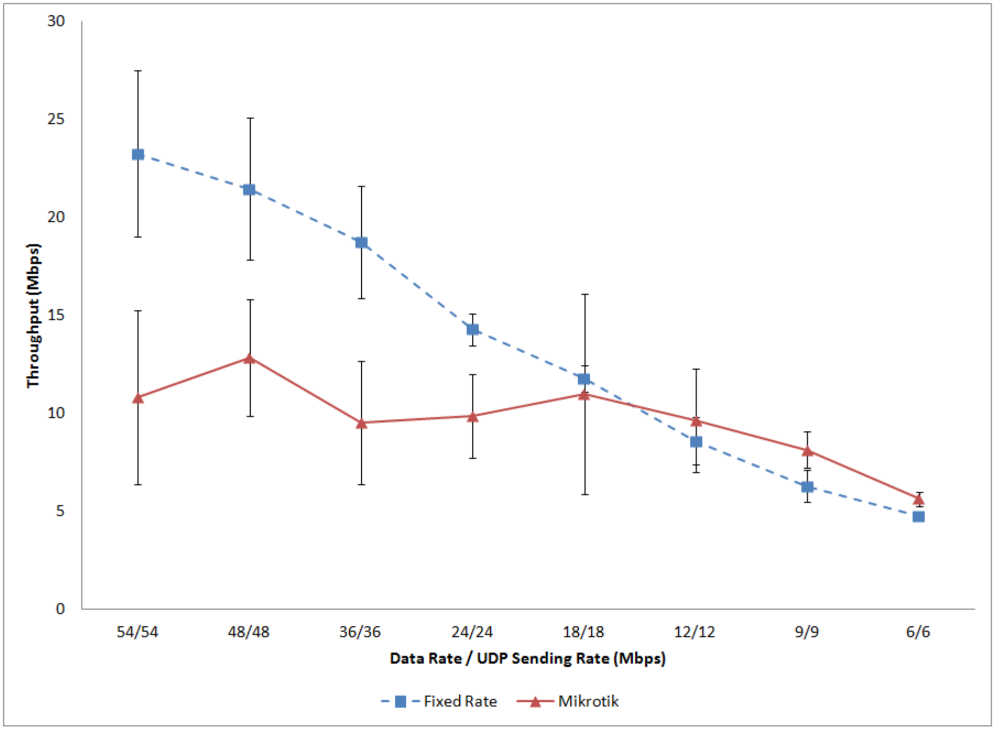


Figure 6.4: Throughput performance - power level: 17 dBm, client location: R1

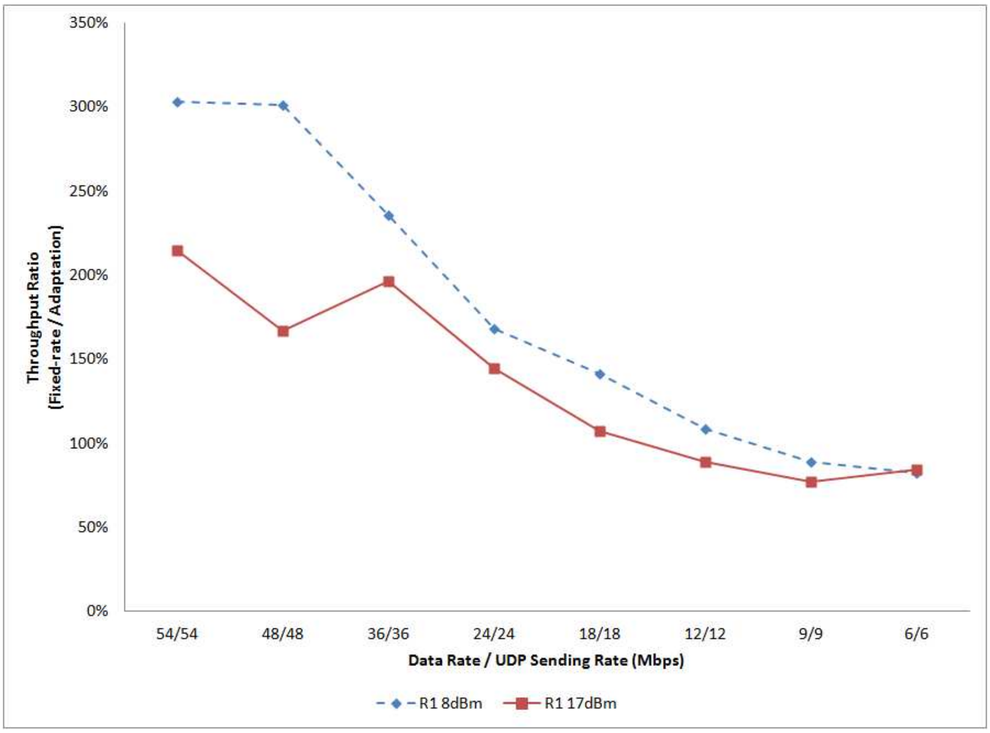


Figure 6.5: Throughput ratio - client location: R1

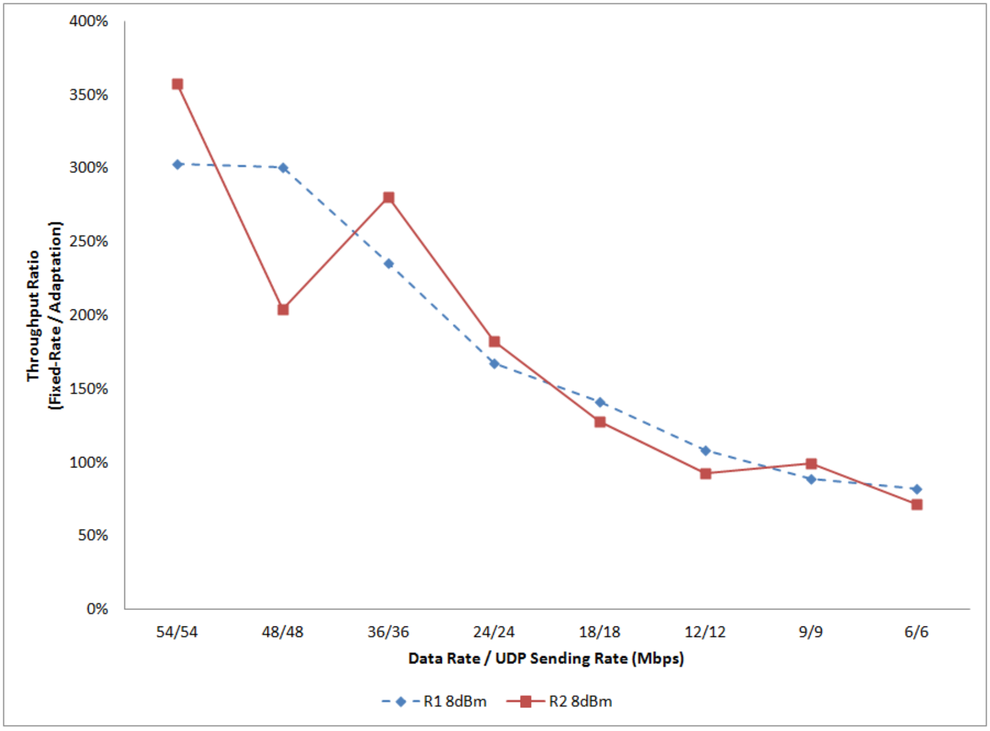


Figure 6.6: Throughput ratio - power level: 8 dBm

it shows that fixed rate significantly outperforms the Mikrotik rate adaptation on most of the eight data rates (up to 300%). The figure also shows that the performance gain by fixed rate over the rate adaptation is more significant with low power level. This is also confirmed by the experiments at location R2 (not shown in this paper).

Figure 6.6 shows the throughput ratio between fixed rate and the rate adaptation with low power level (8 dBm) at locations R1 and R2. In contrast to Figure 6.5, it shows that between locations R1 and R2 there is no much difference on the performance gain, even though the throughput performance is better at location R1 than at location R2 (not shown in this paper). The similar observation is confirmed by the results with high power level (17 dBm) (not shown in this thesis).

Figure 6.7 shows the throughput performance of rate adaptation. There is no much difference between different UDP sending rates, when the sending rate is not low. The throughput is slightly lower when the sending rate is low. This is not surprising - low sending rate, low throughput. However, the lack of throughput difference between different sending rates leads to a conjecture conclusion that the Mikrotik rate adaptation algorithm studied in the experiments could be too conservative.

In terms of power levels, better throughput performance is observed with high power level. This is not surprising . However, the performance with high power level is not as stable as with low power level. As mentioned earlier, this could be due to the interference for the AP at spot CAP. In terms of AP/client distance (R1/R2), with low power level R1 has the better performance. With High power level there is no clear performance difference between R1 and R2 - the distance difference between R1 and R2 appears not big enough to show significant performance difference.

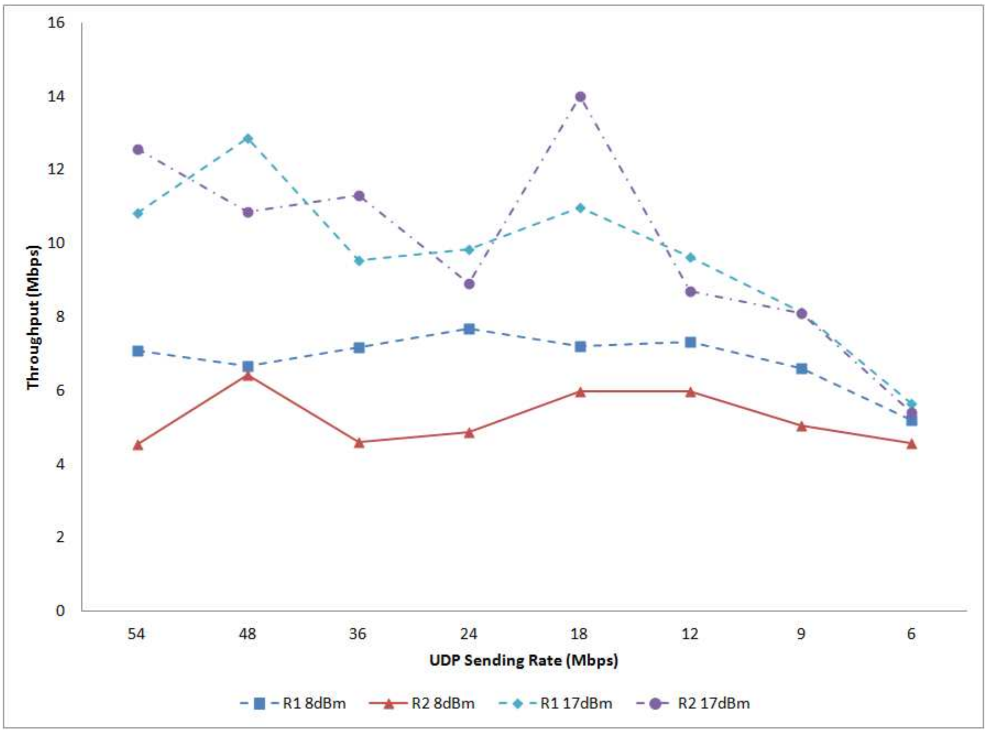


Figure 6.7: Throughput performance - Mikrotik rate adaptation

6.3.2 Maximum and Minimum Throughput

Figure 6.8 and 6.9 show the maximum and minimum throughput of rate adaptation and fixed rate obtained from experiment Exp 1. The corresponding average throughput of the experiment is shown in Figure 6.3, which clearly shows fixed rate outperforms rate adaptation in most cases. However, According to the maximum and minimum throughput in Figure 6.8 and 6.9, except for the two highest data rates (54 and 48Mbps) there is no much difference between the two. In a matter of fact, some experiments show that rate adaptation outperforms fixed rate in many cases (not shown in this paper). Also considering the significant standard deviation, it is hard to predict for a particular instance whether fixed rate is going to outperform rate adaptation. In this section, this study does not intend to make conclusions regarding who outperforms who in terms of minimum and maximum throughput. Indeed, the results from the experiments show conflicting observations (not shown in this paper). However, one thing is clear that due to the unpredictable nature of wireless channels the performance varies significantly from instance to instance.

6.4 UDP - Low UDP Sending Rates

6.4.1 Average Throughput

This section shows the results from the four experiments with Low UDP sending rate (Exp 5 to Exp 8). The goal is to investigate the performance of fixed rate and rate adaptation when buffer overflow losses are not an issue or at least not a major issue. This section also compares with high UDP sending rate.

Figure 6.10 and 6.11 compares the throughput performance of fixed rate and rate adaptation at location R2 with Low and High power levels, respectively. Again, similar to High UDP sending rate fixed rate outperforms rate adaptation as data rate increases. Recall that, with High UDP sending rate, rate adaptation outperforms fixed rate when data rate is low. However, with Low UDP sending rate both mechanisms have the similar performance when data rate is low. This is because with Low UDP sending rate the benefit that rate adaptation can use high data rate becomes less significant. Even when a high data rate is used there is no much data to transmit. The other experiments have the similar observation (not shown in this thesis).

Figure 6.13 shows the throughput ratio between fixed rate and rate adaptation with Low and High power levels at location R1. Again, it shows that the performance gain by fixed rate over rate adaptation is more significant with Low power level and High data rate. Figure 6.12 shows the throughput ratio with Low power level at locations R1 and R2. Again, it shows that between locations R1 and R2 there is no much difference on

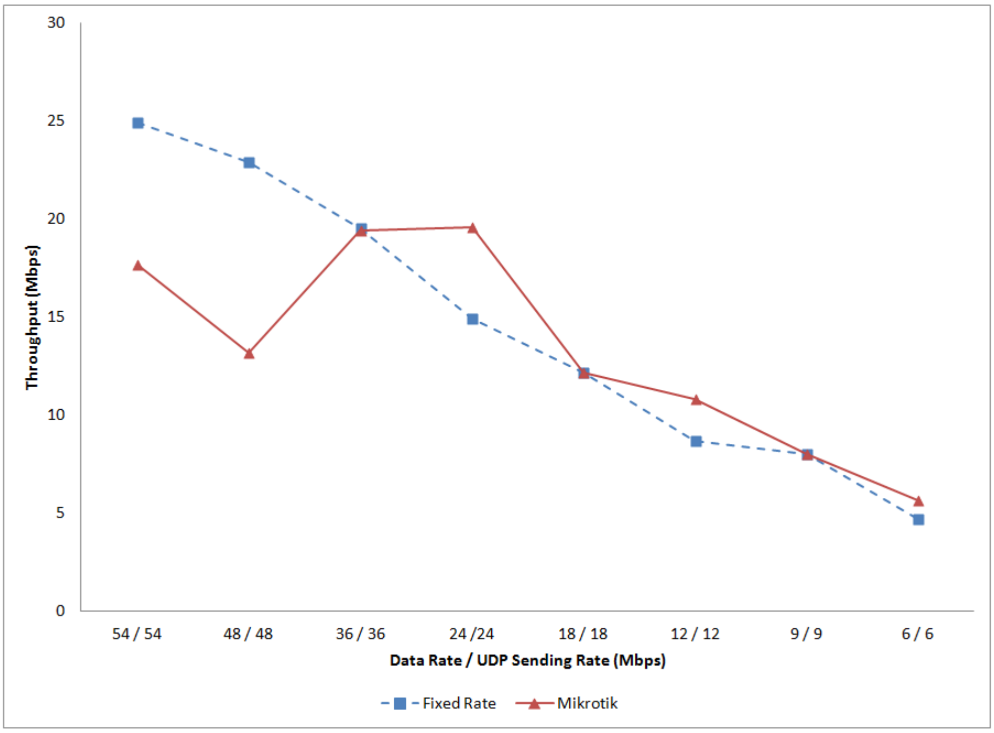


Figure 6.8: Maxium UDP Throughput

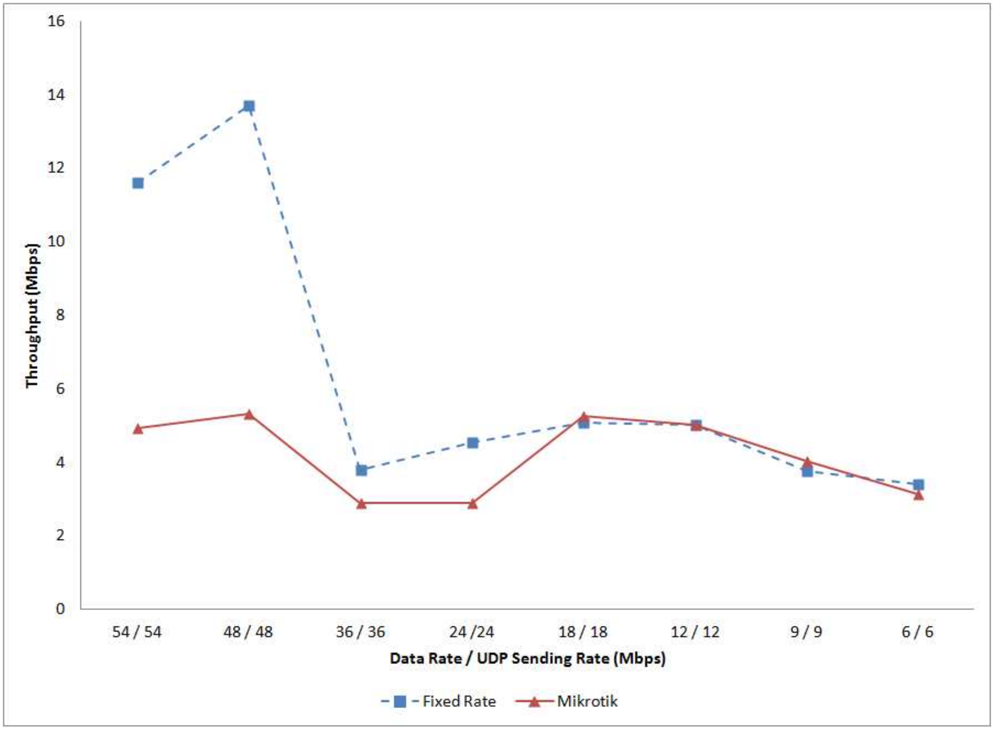


Figure 6.9: Minimum UDP Throughput

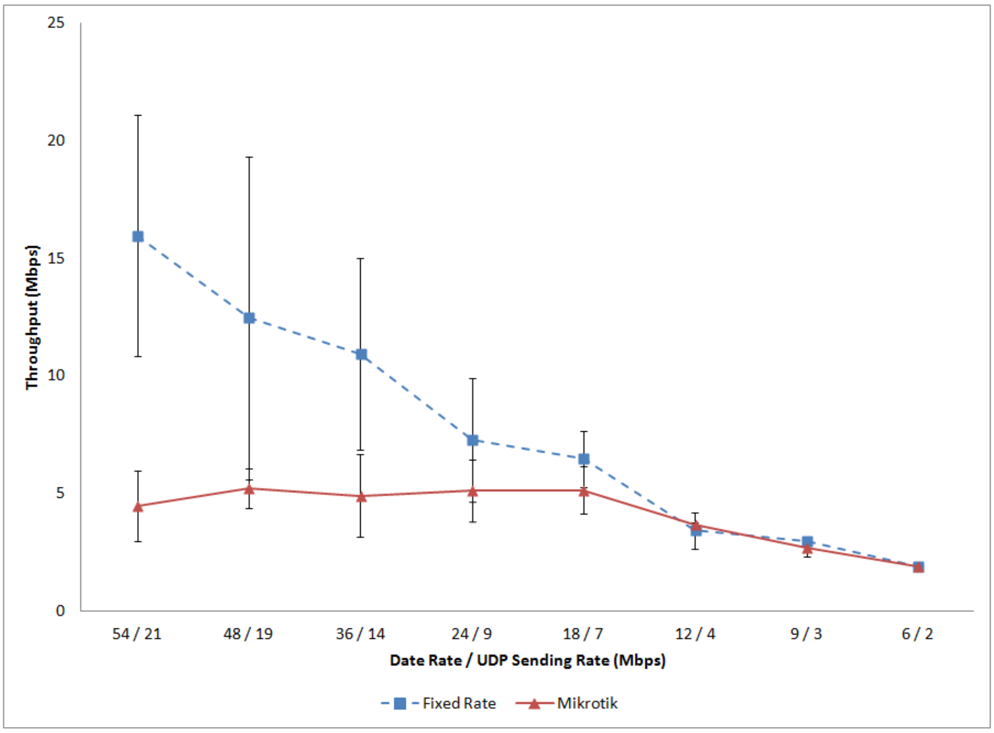


Figure 6.10: Throughput performance - power level: 8 dBm, client location: R2

the performance gain by fixed rate. The similar findings are observed in the other experiments (not shown in this thesis).

6.4.2 Average Delivery Ratio

Figure 6.14 and 6.15 shows the delivery ratio of UDP packets in relation to sending rate and data rate. Delivery ratio measures how much data has been delivered in relation to the amount of data that has been sent. It shows that the delivery ratio increases as sending rate / data rate decreases. In most cases, fixed rate outperforms rate adaptation. With High UDP sending rate and high data rate, rate adaptation has a delivery ratio as low as under 20% while fixed rate performs significantly better. With Low UDP sending rate, fixed rate can achieve a delivery ratio close to

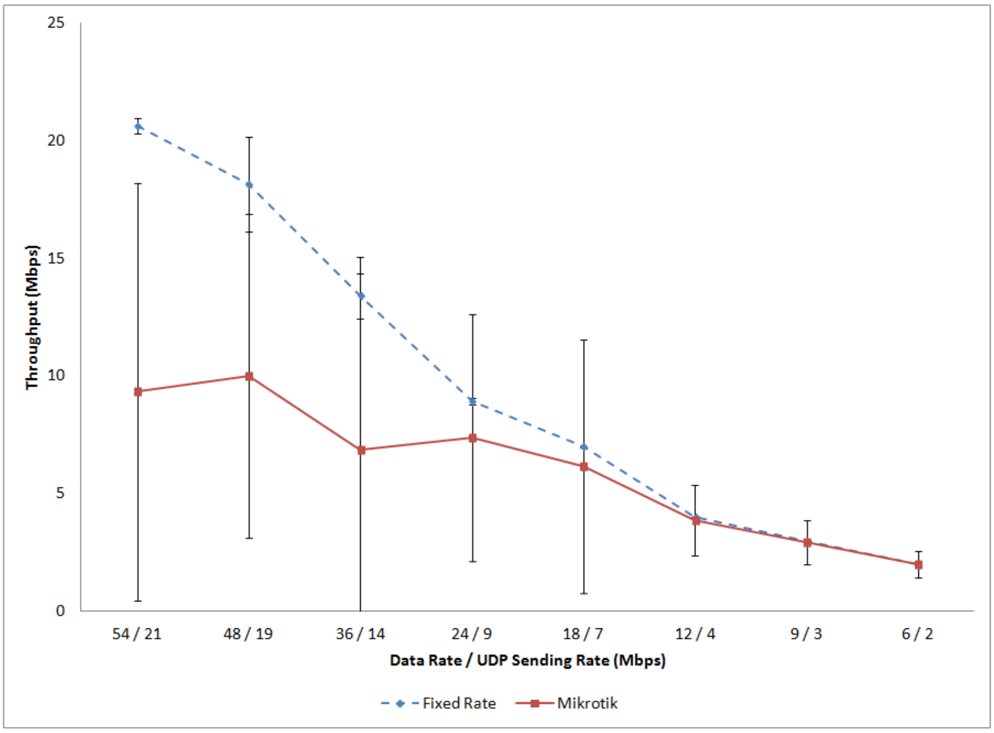


Figure 6.11: Throughput performance - power level: 17 dBm, client location: R2

100% in many cases. In contrast, rate adaptation can only achieve comparable performance when the sending rate is low. For example, in Figure 6.14 using high data rate fix rate can achieve a delivery ratio more than 90%. In the meantime, rate adaptation can only achieve a delivery ratio around 30%.

6.5 TCP

In this section, we describe the comparison of the performance of rate adaptation and fixed rate with TCP traffic. There are four experiments as shown in Table 6.3. In each of the four experiments, four different RTTs (10, 50, 100 and 200 ms) are used to emulate small to large network en-

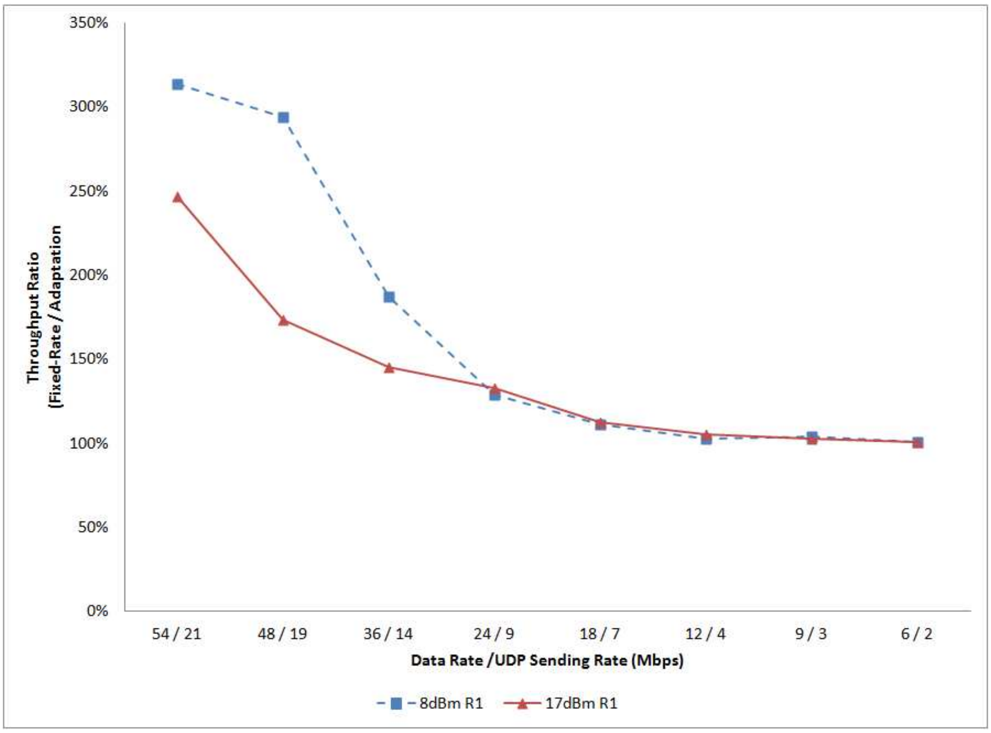


Figure 6.12: Throughput ratio - client location: R1

vironments. In iperf the maximum window size is set to 256 KB. It is observed that all the four experiments have the similar performance. Figure 6.16 shows the throughput performance from experiment Exp 4 with High power level and client node placed at location R2. It shows that fixed rate outperforms rate adaptation in most cases. Only in the cases of low data rate, rate adaptation outperforms fixed rate.

In terms of RTT, the figure shows that the throughput of fixed rate decreases as RTT increases. This is due to the fact that TCP throughput is capped by the maximum TCP window size or the maximum achievable TCP sending rate. Table 6.4 shows the maximum achievable TCP sending rate associated with each of the four RTTs.

The throughput performance is limited by two factors: data rate (channel capacity) and maximum achievable TCP sending rate. In the follow-

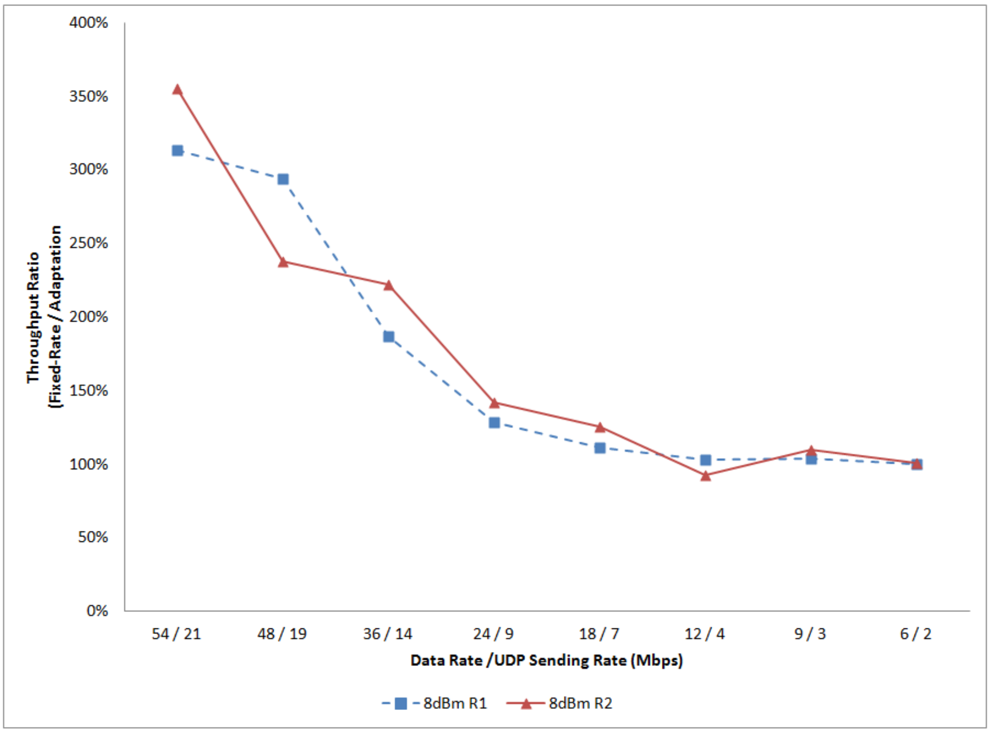


Figure 6.13: Throughput ratio - power level: 8 dBm

ing, we illustrate how these two factors affect throughput performance. Figures 6.17 and 6.18 show the ratio of throughput to data rate and the ratio of throughput to maximum achievable TCP sending rate, with data rates 54Mbps and 6Mbps, respectively.

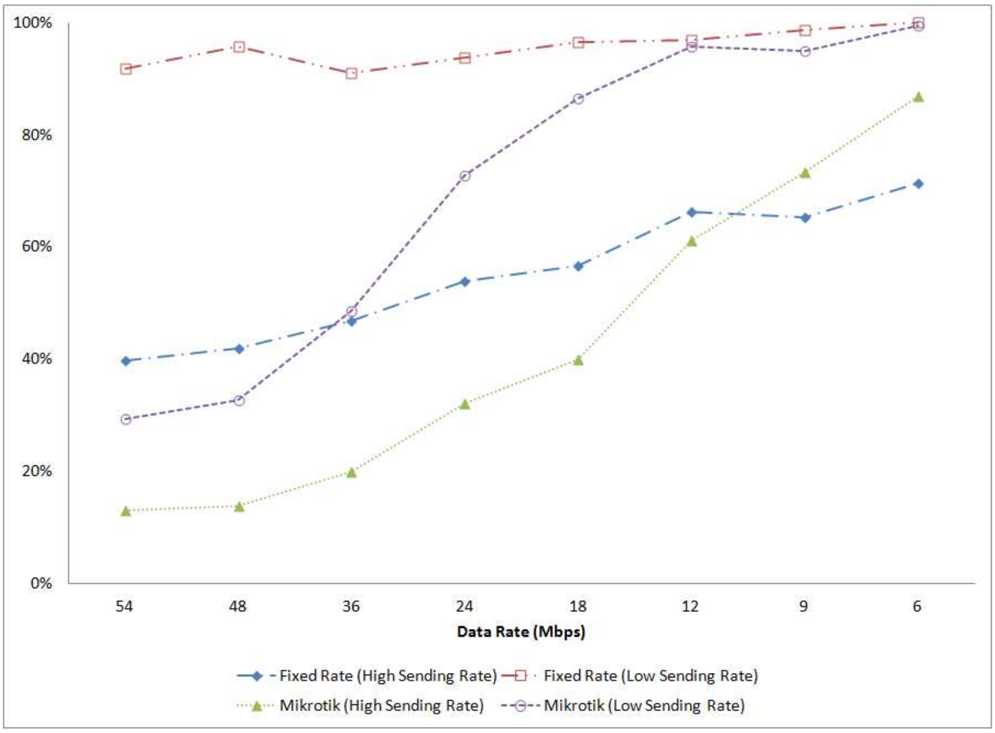


Figure 6.14: Throughput ratio - client location: R1

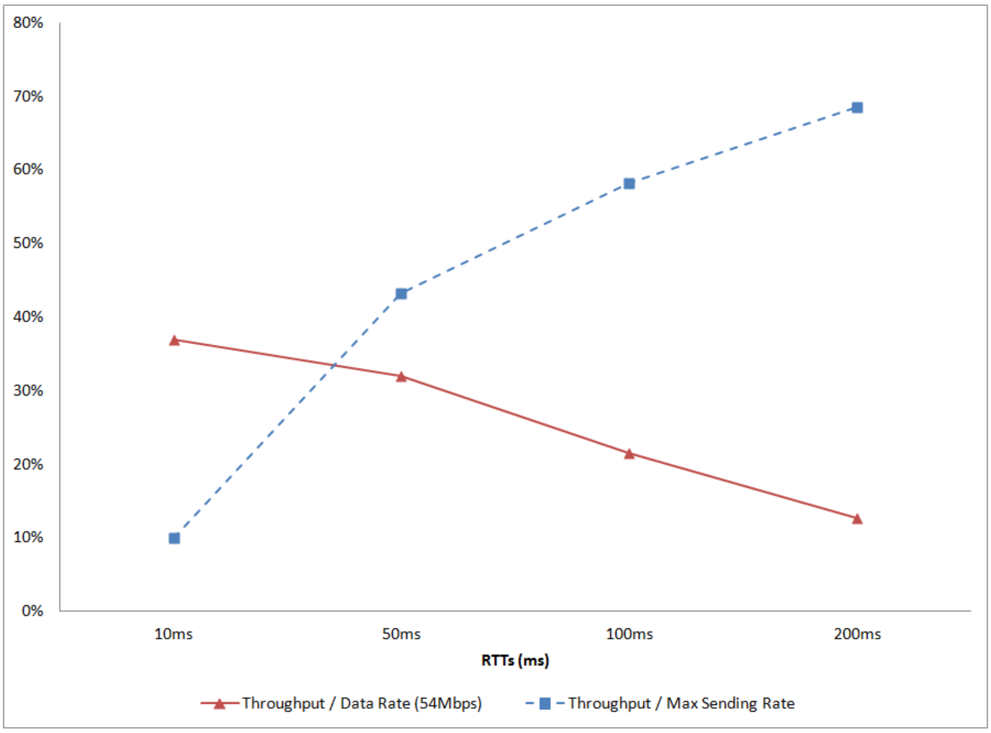


Figure 6.17: Throughput to data rate ratio (54 Mbps) and throughput to maximum sending rate ratio

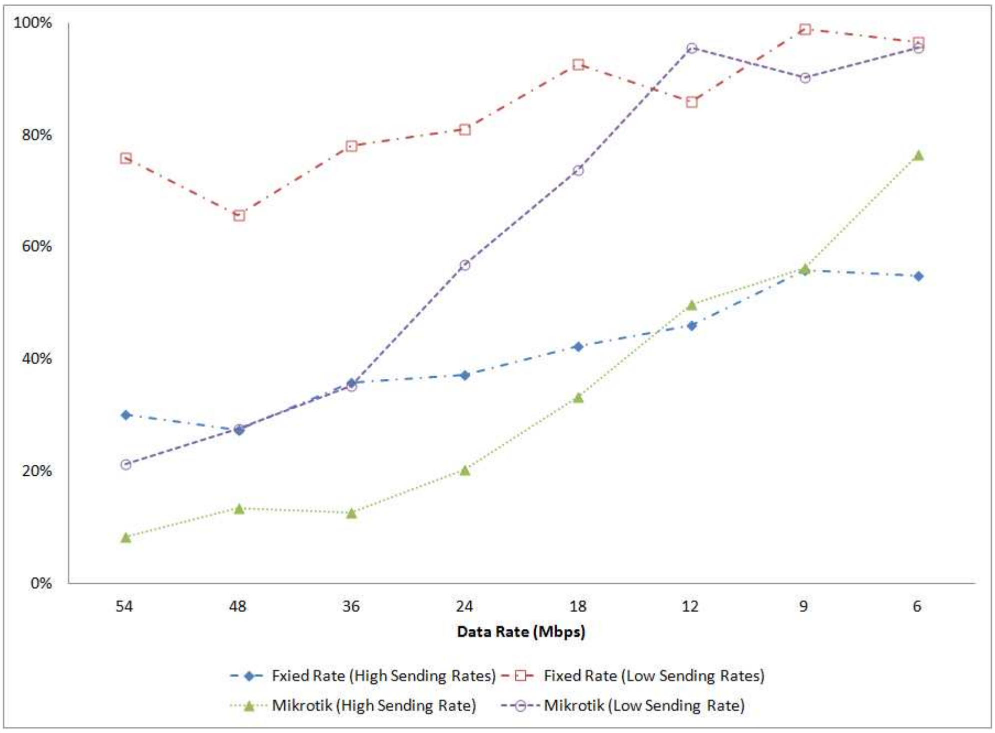


Figure 6.15: Throughput ratio - power level: 8 dBm

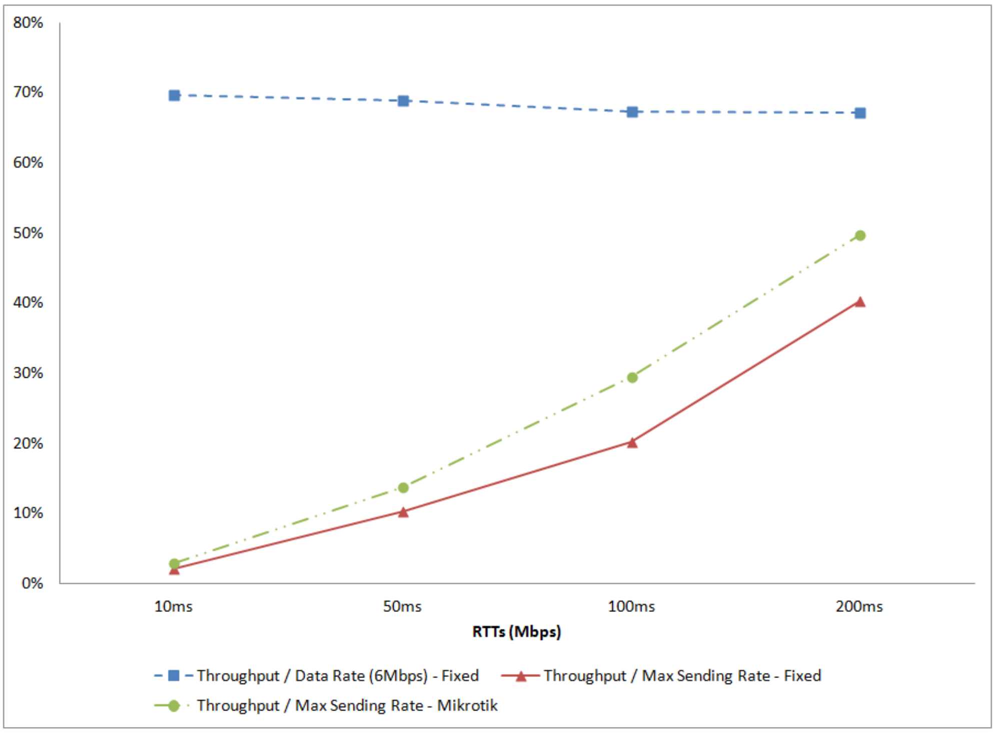


Figure 6.18: Throughput to data rate ratio (6 Mbps) and throughput to maximum sending rate ratio

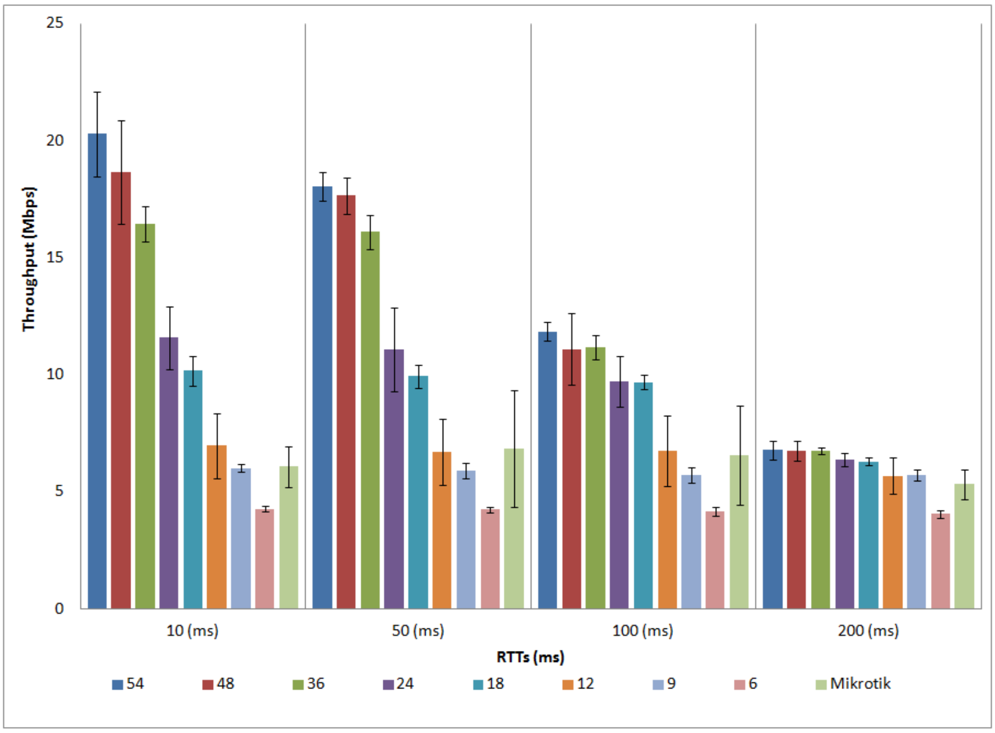


Figure 6.16: Throughput performance - power level: 17 dBm, client location: R2

In Figure 6.17, it shows that the throughput / data rate ratio decreases (roughly from 40% down to 10%) as RTT increases. This indicates the possibility that data rate limits throughput performance decreases with the increase of RTT. On the other hand, the throughput / max. sending rate ratio increases with RTT (roughly from 10% up to 70%). This indicates that the possibility that max. sending rate limits throughput performance increases with RTT. This explains why the throughput of fixed rate decreases with the increase of RTT when high data rates are used.

In Figure 6.18, it shows that the throughput / data rate ratio stays around 70% while the throughput / max. Sending rate ratio increases up to 40%. Although the throughput / max. Sending rate ratio increases with RTT, the ratio (up to 40%) is quite below the throughput / data rate ratio

Window Size	RTT	Maximum TCP Sending Rates
256 KB	10ms	200 Mbps
	DC	Michael Duberry
	DC	Dominic Matteo
	RB	Didier Domi

Table 6.4: Maximum TCP Sending Rates in Relation to RTTs

(around 70%). This indicates that the throughput performance is largely limited by data rate when low data rates are used. The throughput / max sending rate ratio of rate adaptation exhibits the similar pattern as fixed rate when low data rates (e.g., 6Mpbs) are used, indicating its throughput performance is largely limited by data rates. As a result, its throughput does not change much across all RTTs.

This section also describes the comparison of the performance gain by fixed rate over rate adaptation, in relation to power level and AP/client distance (not shown in this paper). The difference is not significant, although Low power level has a slightly greater performance gain. Recall that in UDP the difference is quite significant (Figures 6.5 and 6.12).

6.6 Experiment Summary

This experiment compares the performance of fixed rate and Mikrotik rate adaptation algorithm with both UDP and TCP traffic over an IEEE 802.11g wireless testbed. To better understand the performance issues, in the comparison, a number of factors were taken into account such as AP transmission power, AP/client distance, UDP sending rate and RTT for TCP traffic flows.

This chapter compares the performance of fixed rate and rate adaptation with both UDP and TCP traffic over an 802.11g wireless testbed. To better understand the performance issues, in the comparison, a number of factors were taken into account such as AP transmission power, AP/client distance, UDP sending rate and RTT for TCP traffic flows.

It shows that fixed rate outperforms the Mikrotik rate adaptation in most cases, especially when high data rates are used. The rate adaptation can perform slightly better than fixed rate when fixed rate is forced to use low data rates. However, this slight performance gain disappears when low UDP sending rate is used. Although overall fixed rate performs better, the performance of both mechanisms are not consistent from run to run. This makes it hard to predict for a particular instance which one is going to outperform the other.

It also appears that the performance gain by fixed rate over the Mikrotik rate adaptation is correlated to AP power level. Low power level would see greater performance gain. However, in terms of AP/client distance we do not have such observation, although the performance differs significantly with different AP/client distance. However, we do not rule out the possibility that the difference on AP/client distance is not significant enough in terms of comparing performance gain. For TCP, the performance gain does not differ much with different factors.

Another observation is that the rate adaptation algorithm used in the AP appears to be too conservative for a given condition the algorithm could have chosen a higher data rate. New designs of rate adaptation could take this observation into consideration. For manufacturers, it could be a good idea that for a particular rate adaptation algorithm multiple configuration options are provided for tuning the aggressiveness of the algorithm.

Chapter 7

Theoretical Study of Minstrel

Minstrel[55] rate adaptation algorithm is one of the most common ACK based approach algorithm. It is widely used by many common wireless drivers such as MadWifi, Ath5k, and Ath9k[16]. For the purpose of this thesis, it is important to understand how *Minstrel* rate adaptation algorithm works. Surprisingly, there is very little literature studying the performance of *Minstrel*. In order to conduct a comprehensive real environment experimental based study, it is necessary to conduct a theoretical study. The theoretical study is based on exploring the sources code[94] of *Minstrel* rate adaptation algorithm. The algorithm was invented by Derek Smithies [55] in 2006. It allows different data rates to be used for every remote node. So that *Minstrel* could work better if different wireless channel environment occurred at different remote node. In this chapter, it explored *Minstrel* by explaining the basic Knowledge of *Minstrel* and working process.

7.1 Minstrel Rate Adaptation

Before talking about Minstrel rate adaptation algorithm, it is important to understand four critical design concepts of Minstrel: the retry chain mechanism, the statistics of transmission, the normal transmission, and sampling transmission.

7.1.1 Retry Chain

Minstrel rate adaptation algorithm uses a mechanism known as a multi-rate re-try chain which helps it react to short-term variations in channel quality. The retry chain is one of critical concept of *Minstrel*. The retry chain consists of four rate-count pairs, named r0/c0, r1/c1, r2/c2, and r3/c3. A packet is first transmitted at rate r0 for c0 attempts. If the transmission is not successful after c0 attempts, transmission is attempted at rate r1 for c1 attempts, and the process goes on until the packet is successfully transmitted or ultimately discarded after (c0 + c1 + c2 + c3) unsuccessful transmission attempts. The values of c0 - c3 are chosen so that retry chain is guaranteed to finish in 26ms. If r0/c0 successfully transmits a packet, the rest of retry chain is ignored. Importantly, the retry time for all data rates is almost equal. A high data rate has a higher number of retries than lower data rates due to the faster transmission time of the high rates. The values for r0 - r3 in the retry chain are chosen differently depending on which category of transmission is being performed: normal transmission or sampling transmission. The rates for each transmission category are shown in Table 7.1. The two categories are discussed later.

Retry(*r*)	**Normal Transmission**	**Sampling Transmission**	
		Random < Best	**Random > Best**
0	Best Rate[1]	Best Rate	Random Rate
1	2nd Best Rate	Random Rate	Best Rate
2	Best Probability[2]	Best Probability	Best Probability
3	Base Rate[3]	Base Rate	Base Rate

Table 7.1: Minstrel Retry Chain

[1] A data rate that performs the highest throughput

7.1.2 Statistics of Transmission

Minstrel maintains a history of the packet transmission statistics at each rate based on an Exponentially Weighted Moving Average (EWMA)[95]. These statistics are used to re-evaluate the retry chain every 100 ms. Following the study of the source code, the retry chain involves three main steps to re-evaluate. The first step is to measure the probability of successful transmission for each data rate. The measured successful ratio is given by:

$$\frac{successful\_transmissions}{num\_attempts} \times 100\% \tag{7.1}$$

The second step is to calculate EMWA probability of successful transmission ratio, given by:

$$\frac{success\_ratio \times (100 - ewma\_level) + (old\_ewma\_success\_ratio \times ewma\_level)}{100} \tag{7.2}$$

The success ratio variable is a scaled value; it is the probability of successful transmission based on packet transmissions in the most recent 100 ms interval. The *old_ewma* success ratio is the EWMA scaling value from the previous 100 ms interval. The default scaling value (*ewma*) level is 25%, so the weight of the new result is 75%. This can be compared to *SampleRate*[54], where the weight of the new result is 95%. The third step is to calculate the throughput for each data rate using the following function:

$$ewma\_throughput - ewma\_success\_ratio \times perfect\_throughput \tag{7.3}$$

The *perfect_throughput* is calculated using the time taken to transmit an average-sized packet at the data rate without retransmission.

7.1.3 Normal Transmission

During normal transmission, the rate values in the retry chain are chosen as follows: r0 is set to the rate that achieves the highest *ewma* throughput

[2] A data rate that performs the highest successful transmission rate

[3] The lowest data rate

(see above), r1 is the rate with the second highest *ewma* throughput, r2 is the rate with the highest *ewma* success ratio, and finally r3 is set to the lowest available data rate. This provides a set of gradually decreasing rates that are chosen based on the historical statistics gathered about the condition of the wireless channel in order to provide good throughput in the current wireless environment.

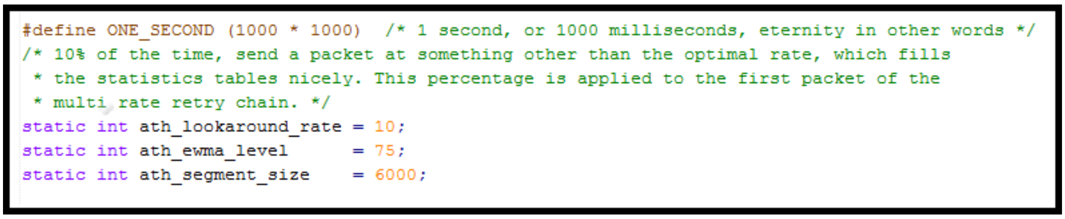
7.1.4 Sampling Transmission

During sampling transmission, frames are chosen to sample the rates not currently in the retry chain, so that the algorithm is able to change the rates in the retry chain if the old rates no longer provide the best throughput. Like *SampleRate* on which *Minstrel* is based, 10% of the data frames are used to sample other data rates. For each sample packet, *Minstrel* randomly picks a rate not currently in the retry chain to sample. The r0 is set to whichever is higher out of the sample rate or the rate with the highest *ewma* throughput and r1 is set to whichever is lower. r2 and r3 remain the rate with the highest *ewma* success ratio and the lowest available rate respectively. This sampling mechanism means that *Minstrel* is more likely to sample higher rates, because higher rates are tried before the rate with the current highest throughput. This results in *Minstrel* being able to effectively increase its sending rates as the quality of the wireless channel increases, which was an area of weakness in previous algorithm such as *SampleRate*.

7.2 Minstrel Source Code Study

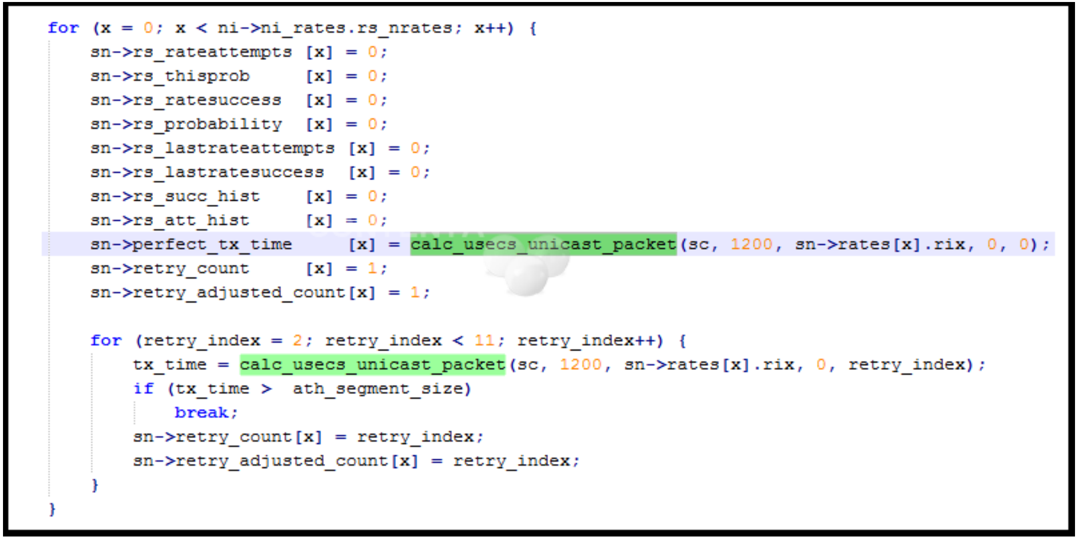
7.2.1 Initialization and Statistic Calculation

System initialization operation has to be executed every single time when *Minstrel* is booted up. The tasks of initialization are to calculate the maximum retry counter for each data rate and reset all the system statistics



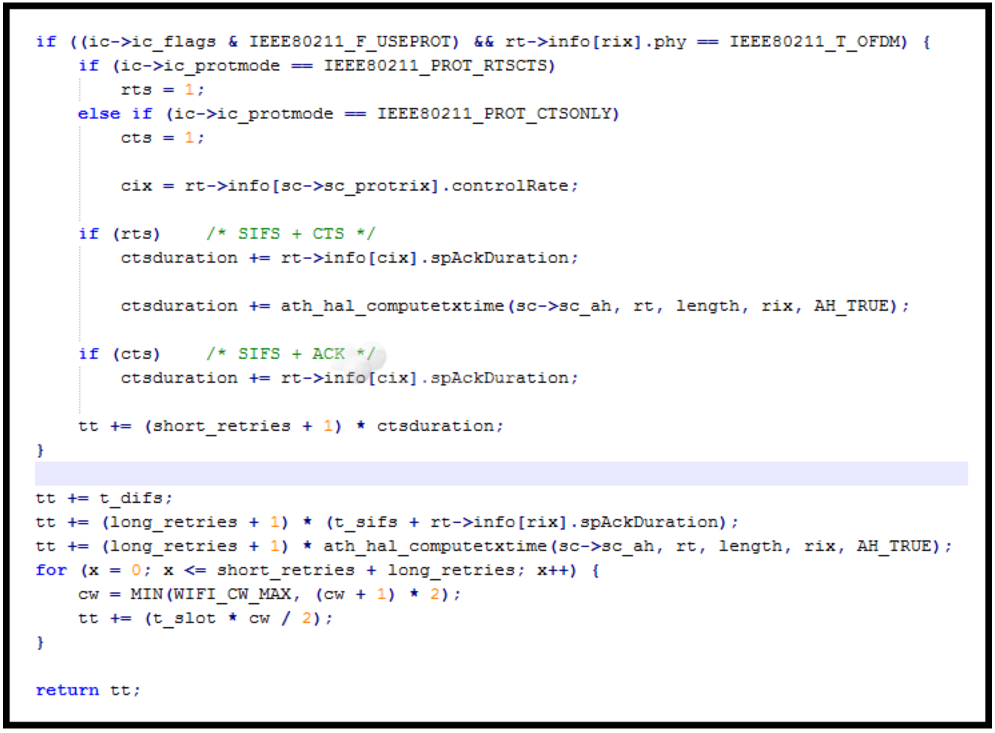
```
#define ONE_SECOND (1000 * 1000)  /* 1 second, or 1000 milliseconds, eternity in other words */
/* 10% of the time, send a packet at something other than the optimal rate, which fills
 * the statistics tables nicely. This percentage is applied to the first packet of the
 * multi rate retry chain. */
static int ath_lookaround_rate = 10;
static int ath_ewma_level      = 75;
static int ath_segment_size    = 6000;
```

Figure 7.1: Minstrel Initialization



```
  for (x = 0; x < ni->ni_rates.rs_nrates; x++) {
      sn->rs_rateattempts [x] = 0;
      sn->rs_thisprob     [x] = 0;
      sn->rs_ratesuccess  [x] = 0;
      sn->rs_probability  [x] = 0;
      sn->rs_lastrateattempts [x] = 0;
      sn->rs_lastratesuccess  [x] = 0;
      sn->rs_succ_hist    [x] = 0;
      sn->rs_att_hist     [x] = 0;
      sn->perfect_tx_time     [x] = calc_usecs_unicast_packet(sc, 1200, sn->rates[x].rix, 0, 0);
      sn->retry_count     [x] = 1;
      sn->retry_adjusted_count[x] = 1;

      for (retry_index = 2; retry_index < 11; retry_index++) {
          tx_time = calc_usecs_unicast_packet(sc, 1200, sn->rates[x].rix, 0, retry_index);
          if (tx_time >  ath_segment_size)
              break;
          sn->retry_count[x] = retry_index;
          sn->retry_adjusted_count[x] = retry_index;
      }
  }
```

Figure 7.2: Minstrel Function: *"ath_rate_ctl_reset"*



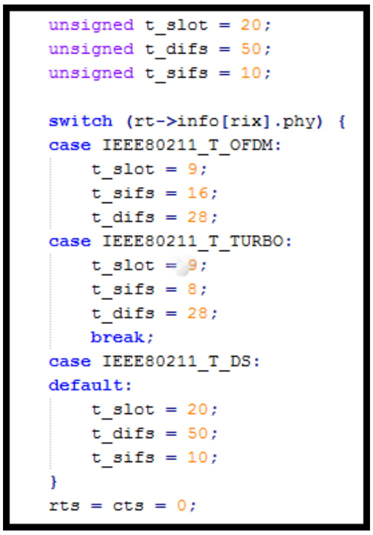
```
if ((ic->ic_flags & IEEE80211_F_USEPROT) && rt->info[rix].phy == IEEE80211_T_OFDM) {
    if (ic->ic_protmode == IEEE80211_PROT_RTSCTS)
        rts = 1;
    else if (ic->ic_protmode == IEEE80211_PROT_CTSONLY)
        cts = 1;

        cix = rt->info[sc->sc_protrix].controlRate;

    if (rts)     /* SIFS + CTS */
        ctsduration += rt->info[cix].spAckDuration;

        ctsduration += ath_hal_computetxtime(sc->sc_ah, rt, length, rix, AH_TRUE);

    if (cts)     /* SIFS + ACK */
        ctsduration += rt->info[cix].spAckDuration;

    tt += (short_retries + 1) * ctsduration;
}

tt += t_difs;
tt += (long_retries + 1) * (t_sifs + rt->info[rix].spAckDuration);
tt += (long_retries + 1) * ath_hal_computetxtime(sc->sc_ah, rt, length, rix, AH_TRUE);
for (x = 0; x <= short_retries + long_retries; x++) {
    cw = MIN(WIFI_CW_MAX, (cw + 1) * 2);
    tt += (t_slot * cw / 2);
}

return tt;
```

Figure 7.3: Minstrel Function: *"calc_usecs_unicast_packet"*

such as probability of successful transmission, number of attempts, etc. There are three important constant variables in this operation as shown in Figure 7.1. The variable *"ath_lookaround_rate"* determines the percentage of time that a sampling transmission is fired. *"ath_segment_size"* is used to calculate the maximum retry counter for a data rate. The default value is 6000 microsecond. *"ath_ewma_level"* is the EWMA level. Figure 7.2 and 7.3 show the functions of calculating the retry counter. The transmission time for each data rate is calculated by performing function *"calc_usecs_unicast_packet"*. The calculation of transmission time is based on the MAC layer configuration and the cost of wireless channel access as shown in Figur 7.4. In addition, the data packet segment size is set to 1200 bytes or 9600 bits. Therefore, the transmission time is an ideal value, it does not represent the transmission time in a real environment. The retransmission time for each data rate (including the time spends on media access) on a retry chain is set to 6000 microseconds as default. Once *Minstrel* rate adaptation algorithm is initialized, the retry counter for each data rate is remained the same. A higher data rate always has larger retry counter than a lower date rate because of a shorter ideal transmission time.

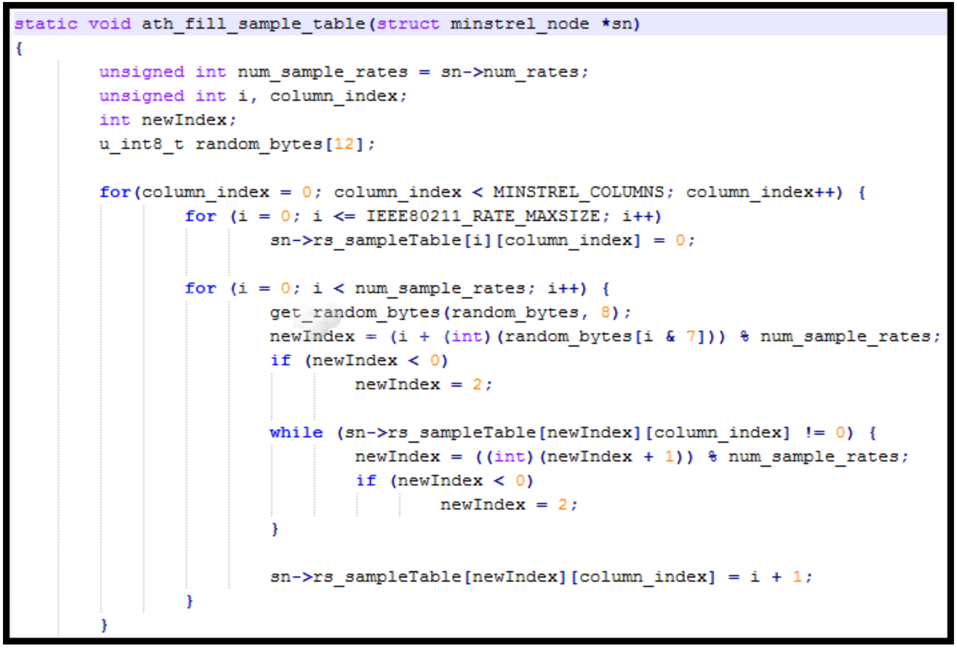
7.2.2 Data Rate Selection

The data sampling acts a very important role in predicating the optimal data rate in a dynamic wireless environment. The *Minstrel*'s sampling algorithm is a major improvement which compares to *SampleRate*. It resolved the issues that *SampleRate* appeared to be stuck in sampling low data rates, and would not move to a higher data rate after a long period of operation. Figure 7.5 shows the function that *Minstrel* generates a table which randomly stores different data rates. So later, it can be used for providing random data rate for sampling transmission. *Minstrel* spends 10% of transmission time to sample a data rate that is not currently used. *"ath_rate_findrate"* (Figure 7.6) is the function that determines whether the



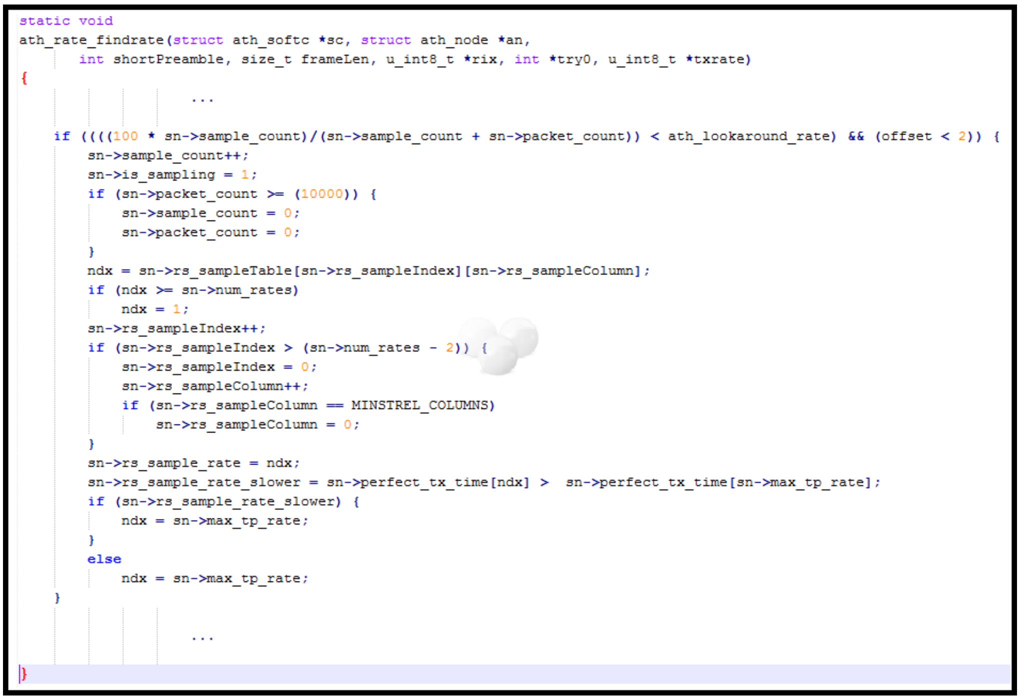
```
unsigned t_slot = 20;
unsigned t_difs = 50;
unsigned t_sifs = 10;

switch (rt->info[rix].phy) {
case IEEE80211_T_OFDM:
    t_slot = 9;
    t_sifs = 16;
    t_difs = 28;
case IEEE80211_T_TURBO:
    t_slot = 9;
    t_sifs = 8;
    t_difs = 28;
    break;
case IEEE80211_T_DS:
default:
    t_slot = 20;
    t_difs = 50;
    t_sifs = 10;
}
rts = cts = 0;
```

Figure 7.4: Channel Access Cost



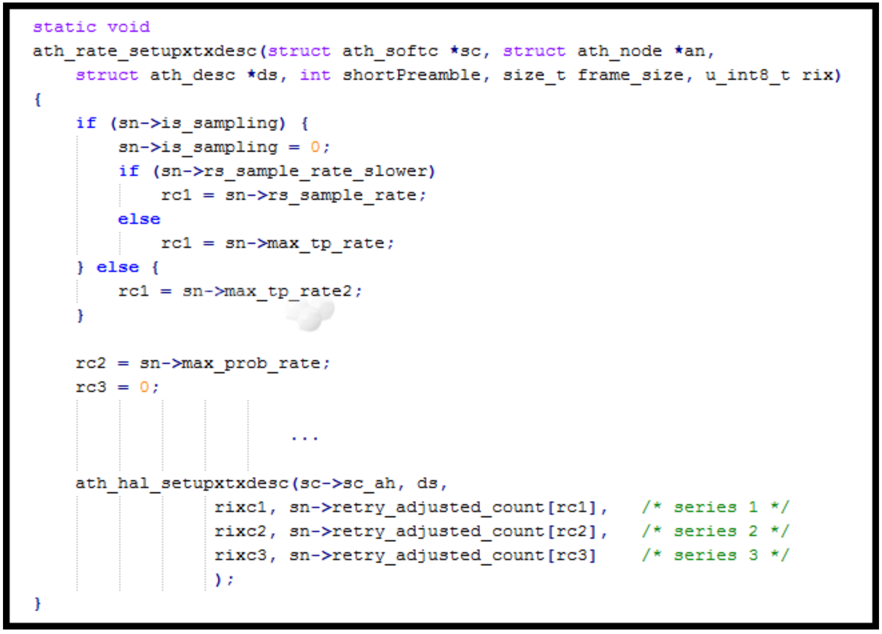
```
static void ath_fill_sample_table(struct minstrel_node *sn)
{
        unsigned int num_sample_rates = sn->num_rates;
        unsigned int i, column_index;
        int newIndex;
        u_int8_t random_bytes[12];

        for(column_index = 0; column_index < MINSTREL_COLUMNS; column_index++) {
                for (i = 0; i <= IEEE80211_RATE_MAXSIZE; i++)
                        sn->rs_sampleTable[i][column_index] = 0;

                for (i = 0; i < num_sample_rates; i++) {
                        get_random_bytes(random_bytes, 8);
                        newIndex = (i + (int)(random_bytes[i & 7])) % num_sample_rates;
                        if (newIndex < 0)
                                newIndex = 2;

                        while (sn->rs_sampleTable[newIndex][column_index] != 0) {
                                newIndex = ((int)(newIndex + 1)) % num_sample_rates;
                                if (newIndex < 0)
                                        newIndex = 2;
                        }

                        sn->rs_sampleTable[newIndex][column_index] = i + 1;
                }
        }
```

Figure 7.5: Minstrel Function: *"ath_fill_sample_table"*



```c
static void
ath_rate_findrate(struct ath_softc *sc, struct ath_node *an,
        int shortPreamble, size_t frameLen, u_int8_t *rix, int *try0, u_int8_t *txrate)
{
                        ...

    if ((((100 * sn->sample_count)/(sn->sample_count + sn->packet_count)) < ath_lookaround_rate) && (offset < 2)) {
        sn->sample_count++;
        sn->is_sampling = 1;
        if (sn->packet_count >= (10000)) {
            sn->sample_count = 0;
            sn->packet_count = 0;
        }
        ndx = sn->rs_sampleTable[sn->rs_sampleIndex][sn->rs_sampleColumn];
        if (ndx >= sn->num_rates)
            ndx = 1;
        sn->rs_sampleIndex++;
        if (sn->rs_sampleIndex > (sn->num_rates - 2)) {
            sn->rs_sampleIndex = 0;
            sn->rs_sampleColumn++;
            if (sn->rs_sampleColumn == MINSTREL_COLUMNS)
                sn->rs_sampleColumn = 0;
        }
        sn->rs_sample_rate = ndx;
        sn->rs_sample_rate_slower = sn->perfect_tx_time[ndx] >  sn->perfect_tx_time[sn->max_tp_rate];
        if (sn->rs_sample_rate_slower) {
            ndx = sn->max_tp_rate;
        }
        else
            ndx = sn->max_tp_rate;
    }

                        ...
}
```

Figure 7.6: Minstrel Function: *"ath_rate_findrate*



```c
static void
ath_rate_setupxtxdesc(struct ath_softc *sc, struct ath_node *an,
    struct ath_desc *ds, int shortPreamble, size_t frame_size, u_int8_t rix)
{
    if (sn->is_sampling) {
        sn->is_sampling = 0;
        if (sn->rs_sample_rate_slower)
            rc1 = sn->rs_sample_rate;
        else
            rc1 = sn->max_tp_rate;
    } else {
        rc1 = sn->max_tp_rate2;
    }

    rc2 = sn->max_prob_rate;
    rc3 = 0;

                    ...

    ath_hal_setupxtxdesc(sc->sc_ah, ds,
                rixc1, sn->retry_adjusted_count[rc1],   /* series 1 */
                rixc2, sn->retry_adjusted_count[rc2],   /* series 2 */
                rixc3, sn->retry_adjusted_count[rc3]    /* series 3 */
                );
}
```

Figure 7.7: Minstrel Function: *"ath_rate_setupxtxdesc"*

current transmission should be in a sampling transmission stage or a normal transmission stage. Every 10000 packets, the sample counter and packet counter reset to zero. If the current transmission is a sampling transmission, then r0 will be either set to the random data rate if the random data rate is higher than the current optimal data rate, or remains the current optimal data rate as r0 if the random data rate is smaller. This is one of the key differences between *Minstrel*'s sampling algorithm and *SampleRate*'s sampling algorithm. As *Minstrel*'s inventor claims that the major improvement stops *Minstrel* to spend time on sampling the lower rates when the channel condition is good. Another important piece of function is "*ath_rate_setupxtxdesc*" (Figure 7.7). The purpose of having this function is to construct the rest of retransmission data rate (r1, r2, and r3). If the randomly selected rate is slower than the current optimal throughput, the randomly selected rate is placed second in the chain. If the link is not good, then there will be data collected at the randomly selected rate. Thus, if the best throughput rate is currently 54 Mbps, the only time slower rates are sampled is when a packet fails in transmission.

7.3 Summary

The determination of a *Minstrel*'s retry chain is depended on the value of the EWMA throughputs and EWMA probability of successful transmission. Moreover, the throughput performance of sending a packet is relied to data rates and the number of retransmitting that packet. The more retransmission causes lower throughput performance.

Previous section has mentioned that *Minstrel* rate adaptation algorithm has four different data rates in a retry chain, each data rate has approximately 6000 microseconds to use for sending a packet before *Minstrel* decided to use next available data rate to send the packet. Retransmission counter and throughput for each data rate is calculated based on an ideal transmission time for each rate. The statistic throughput of *Minstrel* cannot represent the real performance of each data rate in a real environment. These values are indications of data rate selection for *Minstrel*'s retry chain. It is noticed that a higher data rate always has a higher chance to be selected due to its large retransmission counter and the order of retry chain.

Chapter 8

Minstrel Experiment Study

Minstrel is a rate adaptation algorithm that has good performance compared with other algorithms, and is widely implemented in popular wireless drivers, *Ath5k* and *Ath9k*. However, surprisingly, there is very little literature studying the performance of *Minstrel*.

The *Minstrel* theoretical study has indicated that *Minstrel* rate adaptation algorithm prefers to select higher data rates. This characteristic helps *Minstrel* to have faster recovery if wireless channel conditions are from bad to good. However, the aggressiveness of *Minstrel* can also become costly if wireless channel conditions drop from good to bad. Therefore, *Minstrel* rate adaptation algorithm experimental study was focused on UDP transmission in static channel environments and dynamic channel environments.

In this chapter, it presents an experimental study using the MikroTik wireless access point which evaluates the performance of *Minstrel* against fixed rates in a real-world IEEE 802.11g testbed. This experimental study has two major sections, the downlink experiments and uplink experiments. Our published paper [20] is based on our downlink experiments and paper [21] is based on uplink experiments

8.1 Minstrel Experiment Factors

The main experimental study is focused on the performance of *Minstrel* rate adaptation algorithm. The aim of this experiment is to understand the performance issues of *Minstrel* in a real-world environment. Previous studies such as [82] has shown that *Minstrel* rate adaptation algorithm outperforms other rate adaptation algorithms such as *SampleRate*. However this is not to say that *Minstrel* gives the best possible performance in all situations. In *Minstrel* experimental study, part one evaluates the performance of *Minstrel* rate adaptation algorithm in a downlink channel environment, and part two evaluates the algorithm in a uplink channel environment. The study gives insight into areas where *Minstrel* rate adaptation algorithm can perform better. Three locations (L1, L2, and L3) were selected to place the client PC (Figure 5.2). Spot AP was the location for the Access Point. L1 has the shortest transmission distance to the AP while L3 has the longest distance and L2 is in the middle. Two glass walls are between these three locations and the AP. Whilst the 802.11g standard provides data rates up to 54 Mbps, the overhead of the MAC protocol means that the realistic throughput is usually below half the data rate. Therefore an offer load of 35 Mbps is enough to saturate the wireless channel between the laptop and the AP.

The first part of *Minstrel* Experimental Study focuses on evaluating the performance of *Minstrel* rate adaptation algorithm in a downlink channel environment. It is carried out with four scenarios: ″static″ channel conditions (downlink), ″dynamic″ channel conditions (downlink), variable datagram size with ″static″ channels and variable datagram size with ″dynamic″ channels.

The second part of *Minstrel* Experimental Study focuses evaluating the performance of *Minstrel* rate adaptation algorithm in a uplink channel environment. It is also carried out with three scenarios that have different impacts on network performance: ″static″ channel conditions (uplink), dynamic channel conditions (uplink) and mobile transmission.

Static Channel Condition. In this scenario, the transmission power is fixed for the entire duration of the transmission. This means the quality of the wireless channel is relatively stable during each experiment. Different experiments are conducted with different transmission powers ranging from 17 dBm to 1 dBm. High power levels are used to emulate a short transmission distance between the wireless sender and receiver. In this case, the wireless channel quality is good and the channel's resistance to interference is high. Low power levels are used to emulate a long transmission distance or the scenarios where small network coverage is desired. This means that the wireless channel quality is low and the channel is more susceptible to interference. This experiment provides basic performance comparison between *Minstrel* rate adaptation algorithm and fixed rates.

Dynamic Channel Condition. Dynamic wireless channel conditions are emulated by making the transmission power flip between 17 dBm and 1 dBm during experiment run. The transmission power stays at 17 dBm for 2 sec before flipping to 1 dBm, the time of flipping power back to 17 dBm is varied between 2 seconds and 10 seconds . As the result, the degree to which the wireless channel is resistant to interference changes from time to time during the experiment. These dynamic channel tests provide performance comparison between *Minstrel* rate adaptation algorithm and fixed rates in a complex and variable wireless environment.

Variable Datagram Size with "static" Channels. In this experiment, the experiment scenario is similar to the experiment with a static channel condition. The UDP datagram sizes varies from 100 to 2000 bytes. The experiment emulates *Minstrel* reacts to packet losses in a static channel condition. The transmission power is set from 17 dBm to 1 dBm. The datagram size and the transmission power are fixed during the entire transmission.

Variable Datagram Size with "dynamic" Channels. This experiment is similar to the experiment with a dynamic channel condition. In this experiment, the datagram size also varies from 100 to 2000 bytes. The datagram size is fixed during the entire transmission. The transmission power, however, flips between 17 dBm and 1 dBm.

Mobile Transmission. Sometimes the channel channel will change gradually, for example, if a wireless client moves spatially with respect to the AP. In this scenario, we gradually change the wireless channel condition by increasing or decreasing the transmission power during the experiment. We consider both low to high and high to lower changes. We run experiments with different update periods, i.e. the time value in between consecutive power level updates. We vary this period from 2 to 5 seconds. At the beginning of the experiment, the transmission power is either set to 17 dBm or 1 dBm, then during the experiment the transmission power is increased or decreased by 2 dBm increments at each update period until it reaches 1 dBm or 17 dBm.

8.2 Downlink Experiments

This section shows the performance comparison between *Minstrel* rate adaptation algorithm and fixed rate with UDP downlink traffic. The focus of the experiment is on three factors that can have an impact on the performance of the rate adaptation and fixed rate: static channel transmission, dynamic channel transmission and datagram size.

8.2.1 Static Channel Transmission

In this scenario, data is transmitted from server to client for 60 seconds, the transmission power is adjusted at the AP side. The transmission power is kept steady for the duration of each experiment. High power level is used to emulate the scenarios where the channel of the AP side is relatively resistant to interferences. Low power level is used to emulate the scenarios where the AP side of the channel is relatively vulnerable to interferences.

The offered traffic load is set to 35 Mbps.

Figure 8.1, 8.2, and 8.3 shows the throughput performance of fixed rates and *Minstrel* against transmission power. In Figure 8.1, the AP is placed at location L1, which is closest to the client node. It clearly shows that compares to other fixed rates 54Mbps data rate has the best throughput performance through the entire range of transmission power. *Minstrel* has a performance comparable to 54Mbps.

When the AP is moved away from the client node to location L2 (Figure 8.2), the throughput performance drops. 48Mbps and 54Mbps have a better performance than the other fixed rates, while 48Mbps starts to outperform 54Mbps with low transmission power. *Minstrel* has the similar performance as 48/54Mbps with high transmission power. However, when the transmission power is low, *Minstrel* cannot catch up the performance of 48/54Mbps or even 36Mbps.

In Figure 8.3, the AP is further moved away from the client node to location L3. The throughput performance is dropped again. The performance of 54Mbps suffers a lot from this change of location. It now has the worst performance. Data rate 36Mbps gives the best performance. *Minstrel* has a performance comparable to 48Mbps and 36Mbps when the transmission power is high. However, with low transmission power 36Mbps outperforms *Minstrel*.

With the three locations (L1, L2 and L3), the best performer is 54Mbps/ *Minstrel*, 48/54Mbps and 36Mbps, respectively. Figure 3 shows the throughput ratio between *Minstrel* and the fixed rate that achieves the overall best throughput (54, 48 and 36Mbps for L1, L2 and L3, respectively). For locations L2 and L3, the ratio goes down from around 100% to around 70%. Although *Minstrel* is among the best when the transmission power is high, it may not be able to achieve the performance of certain fixed rates when the transmission power is low. This indicates that in the environments vulnerable to interferences *Minstrel* at AP side may not be able to select the optimal data rate for transmission.

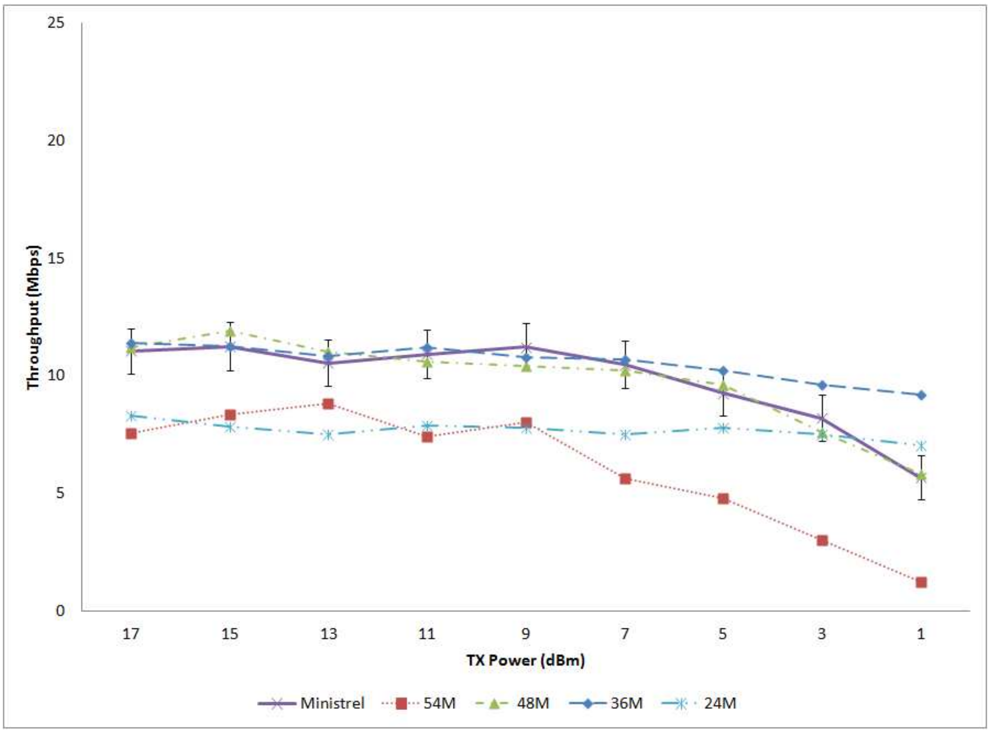


Figure 8.1: Throughput performance with static channels - AP location: L1

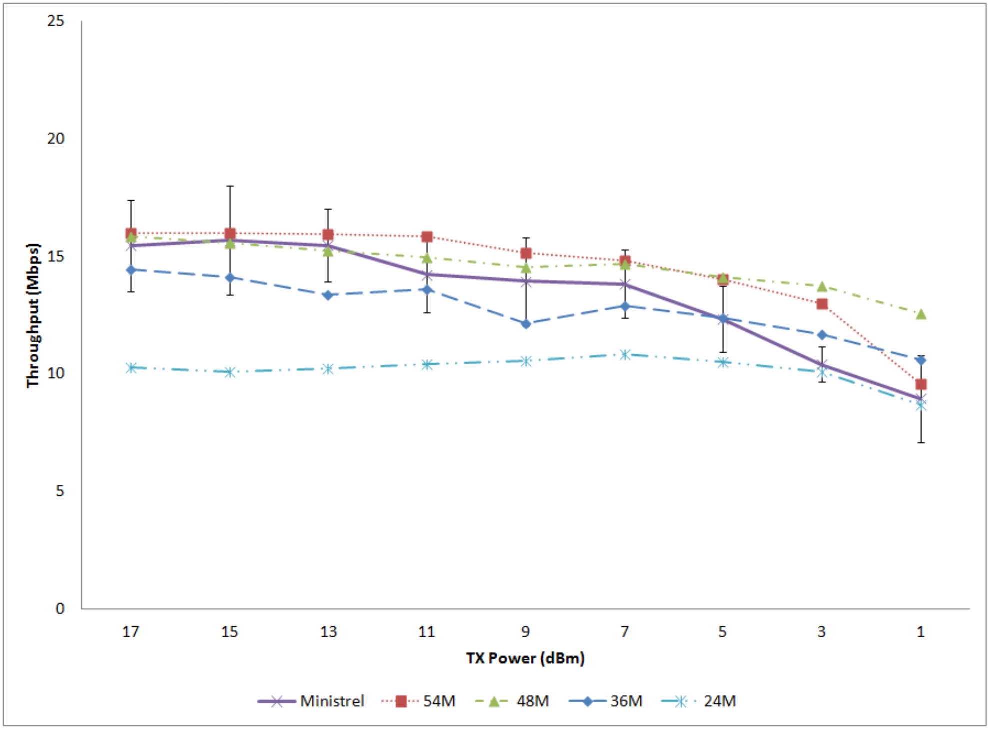


Figure 8.2: Throughput performance with static channels - AP location: L2

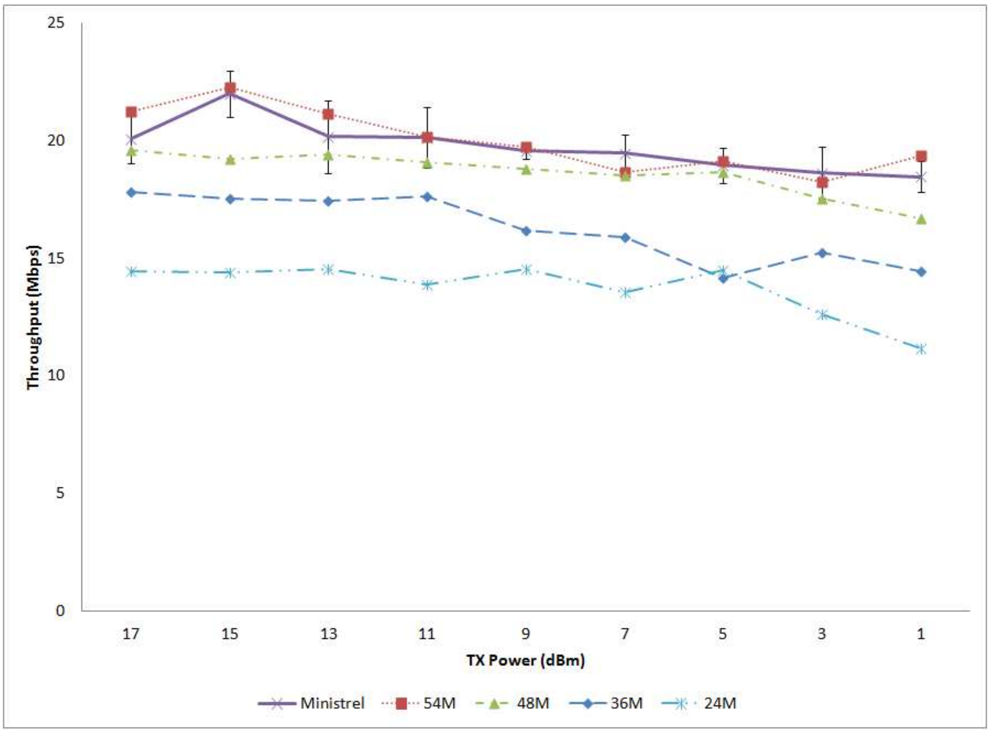


Figure 8.3: Throughput performance with static channels - AP location: L3

8.2.2 Dynamic Channel Transmission

In this section, the experiment is conducted in dynamic channel conditions. A dynamic wireless channel is created by flipping the transmission power between 17 dBm and 1 dBm at the AP side. The transmission power stays at 17 dBm for 2 sec before flipping to 1 dBm. The dynamism of the channel is adjusted by varying the duration that the power level stays low between 1 and 9 seconds, and each test runs for 60 seconds. As a result, the channel becomes resistant/vulnerable to interferences from time to time.

Figure 8.4, 8.5 and 8.6 shows the performance of *Minstrel* and fixed rates at the three locations. At location L1, 54Mbps and *Minstrel* have the best performance. However, at locations L2 and L3 *Minstrel* is not the best performer. At L2, 36/48Mbps consistently outperforms *Minstrel*. At L3, *Minstrel* has a performance similar to 24Mbps, but consistently below

36Mbps. Figure 8.7 shows the throughput ratio between *Minstrel* and the fixed rate that achieves the overall best throughput. For locations L2 and L3, the ratio is around 80%. It is a bit surprising. *Minstrel* has the capability of adapting to channel conditions while fixed rates do not adapt at all. In a dynamic environment, the performance of *Minstrel* is expected to be better. However, it is not always the case. The experimental results indicate that with high channel dynamics at AP side, *Minstrel* does not adapt well.

8.2.3 Datagram Size with Static Channel Transmission

Minstrel reacts to packet losses. In wireless environment, datagram sizes have a direct impact on perceived packet loss behaviours including packet loss ratio. In this section, the focus of the experiment is on the impact of datagram sizes on the performance of *Minstrel* and fixed rates with static channel transmission. The datagram size is set from 100 bytes to 2000 bytes. Similar to Section 8.2.1, the transmission power is set from 17dBm to 1dBm. The datagram size and the transmission power are fixed during the entire transmission.

Figure 8.8, 8.9, 8.10 and 8.11 shows the performance of *Minstrel* and fixed rate against datagram sizes with different locations and transmission power. It shows that the throughput performance initially increases with datagram size and then decreases. This is due to the fact that as the datagram size increases the relative overhead (DIFS/SIFS) gets smaller. However, for a given BER a larger size means higher packet loss rate. This explains why the throughput performance starts to drop when the datagram size continuously increases.

In terms of the comparison between *Minstrel* and fixed rates, in most cases *Minstrel* can achieve a performance similar to the fixed rate that has the overall best throughput. With low transmission power, *Minstrel* is still consistently outperformed by some of the fixed rates. However, there is no clear correlation between datagram size and *Minstrel*'s performance rela-

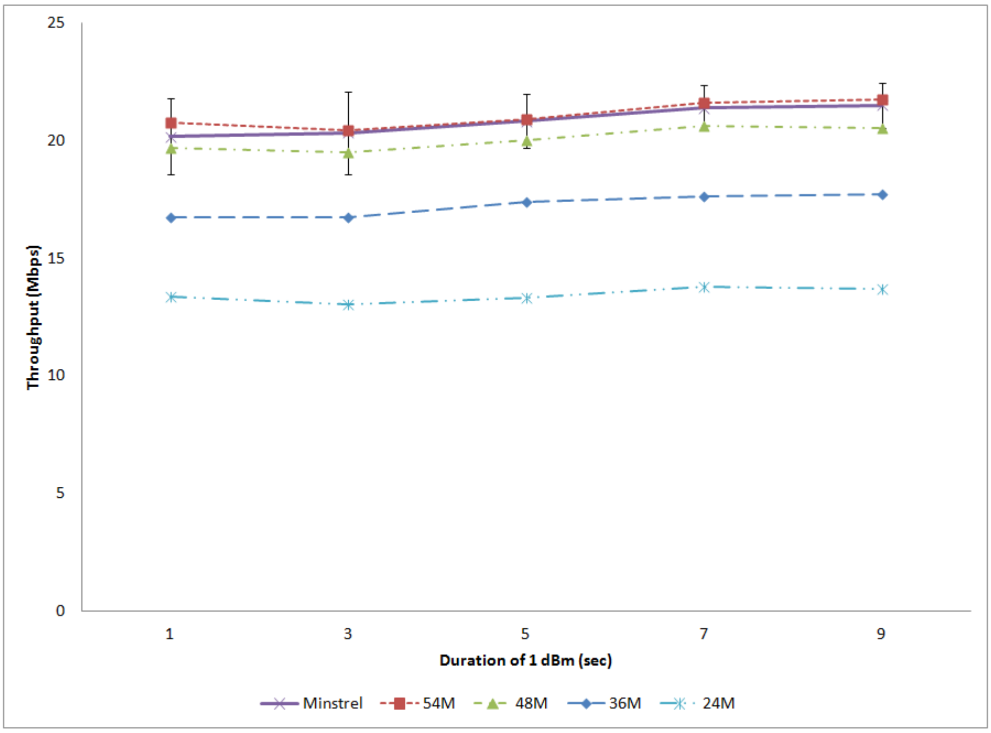


Figure 8.4: Throughput performance with static channels - AP location: L1

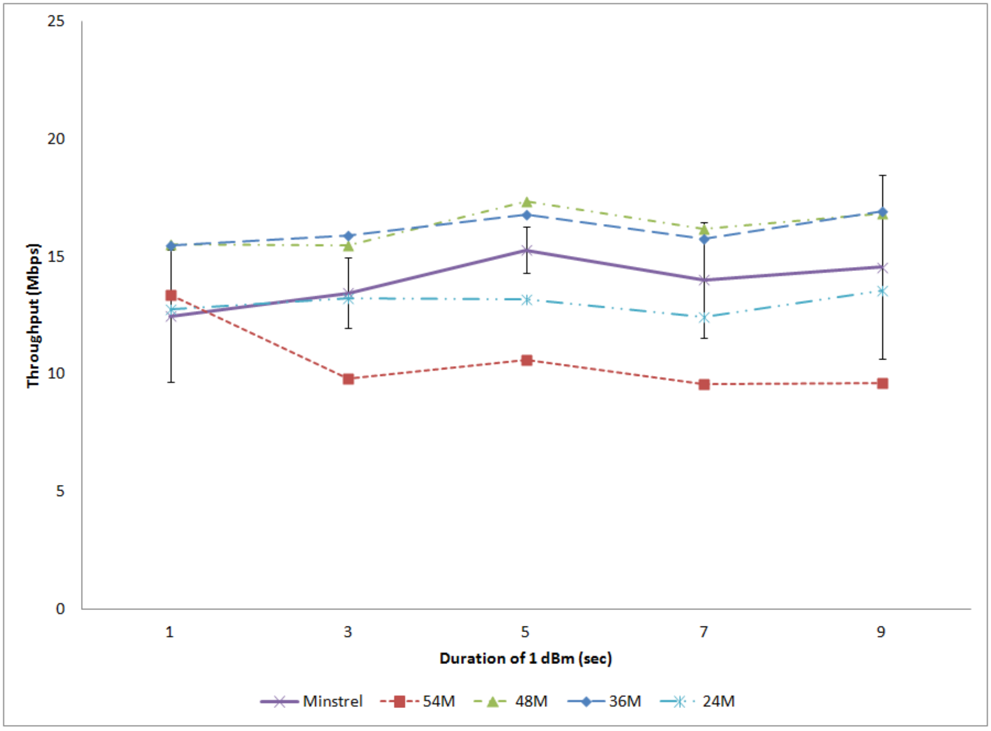


Figure 8.5: Throughput performance with static channels - AP location: L2

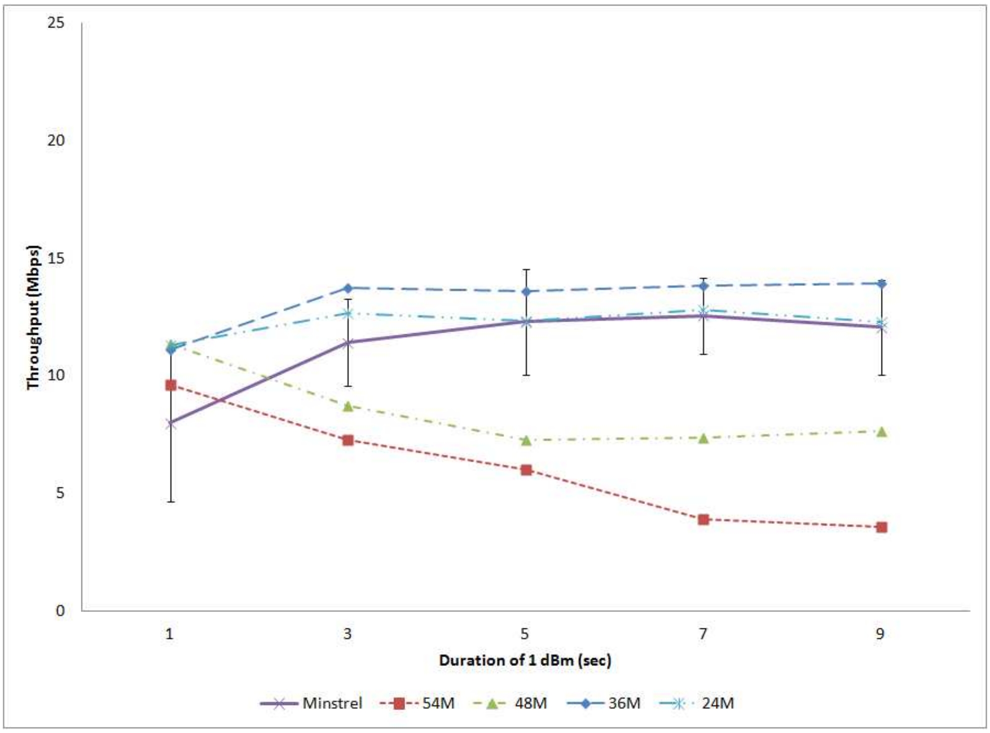


Figure 8.6: Throughput performance with static channels - AP location: L3

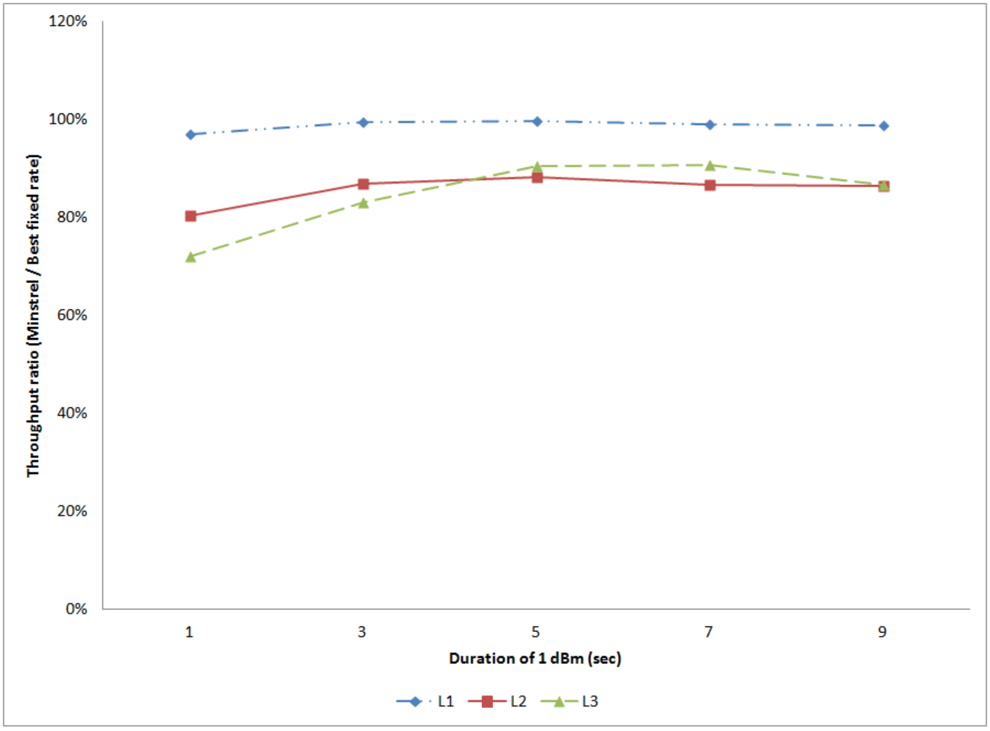


Figure 8.7: Throughput ratio (*Minstrel* / fixed rate)

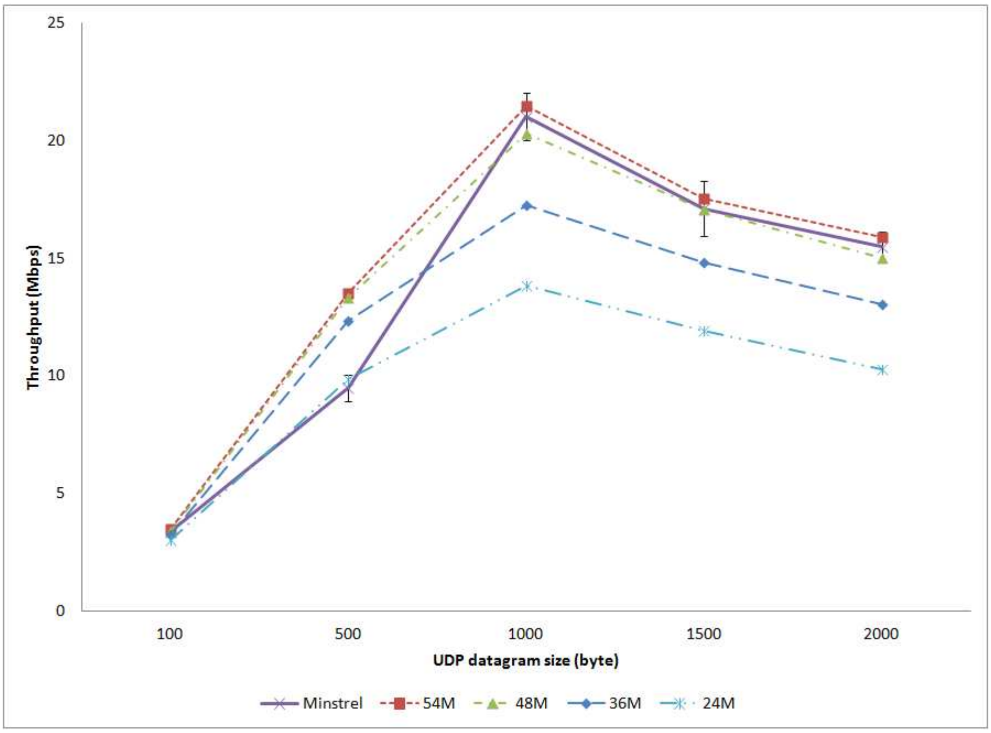


Figure 8.8: Throughput vs. datagram size (static channels) - power level: 1 dBm, AP location: L1

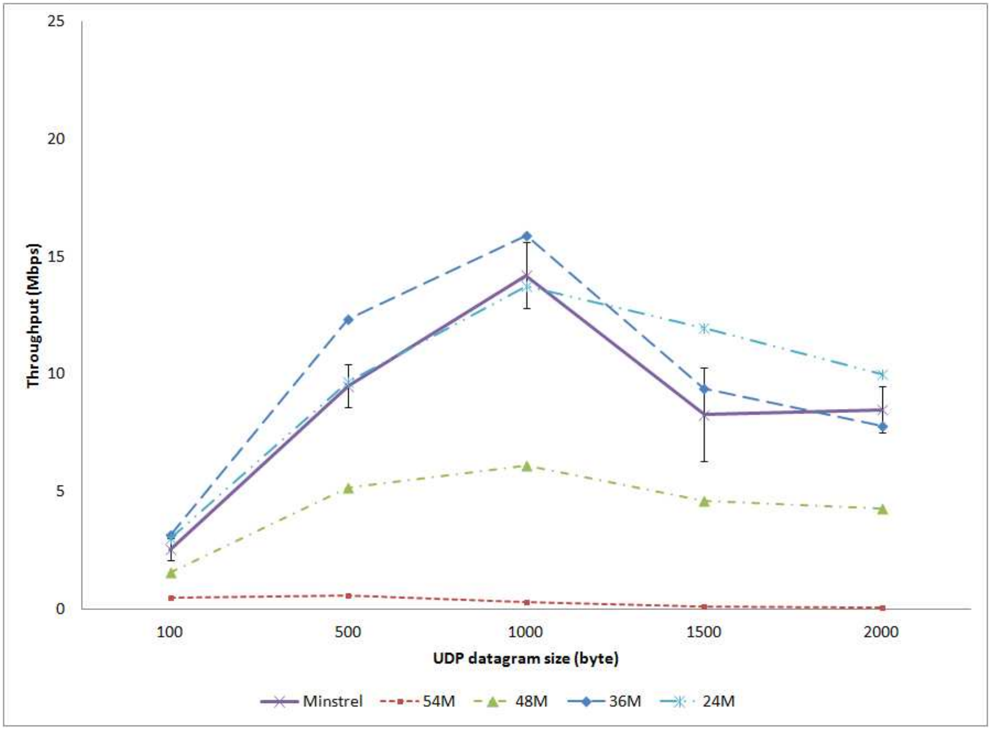


Figure 8.9: Throughput vs. datagram size (static channels) - power level: 1 dBm, AP location: L3

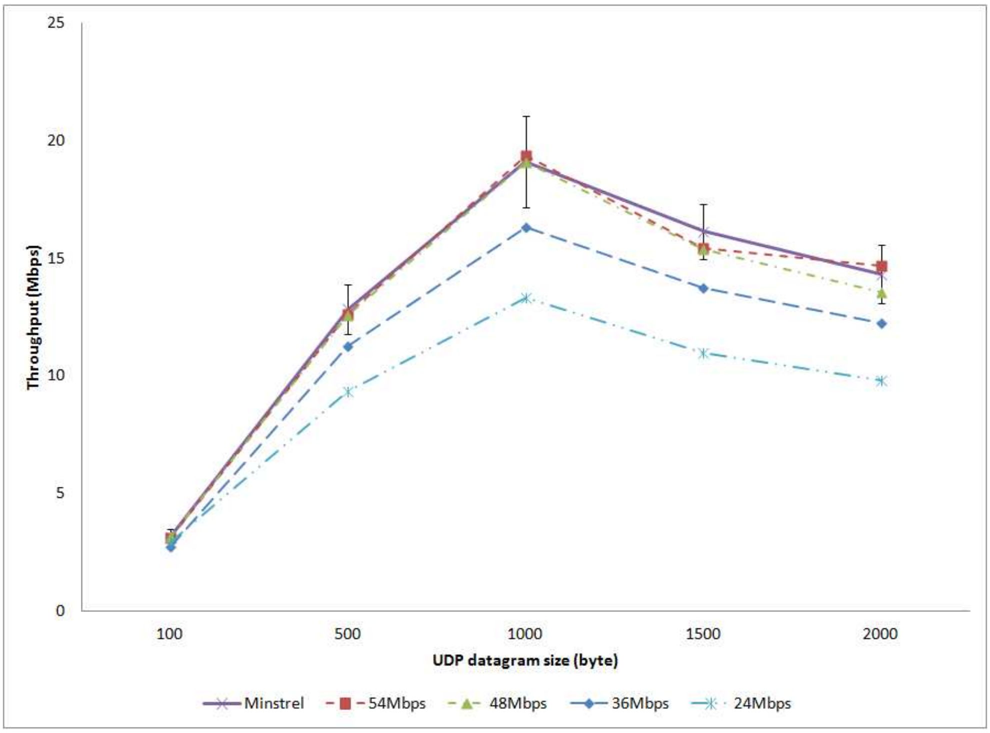


Figure 8.10: Throughput vs. datagram size (static channels) - power level: 5 dBm, AP location: L2

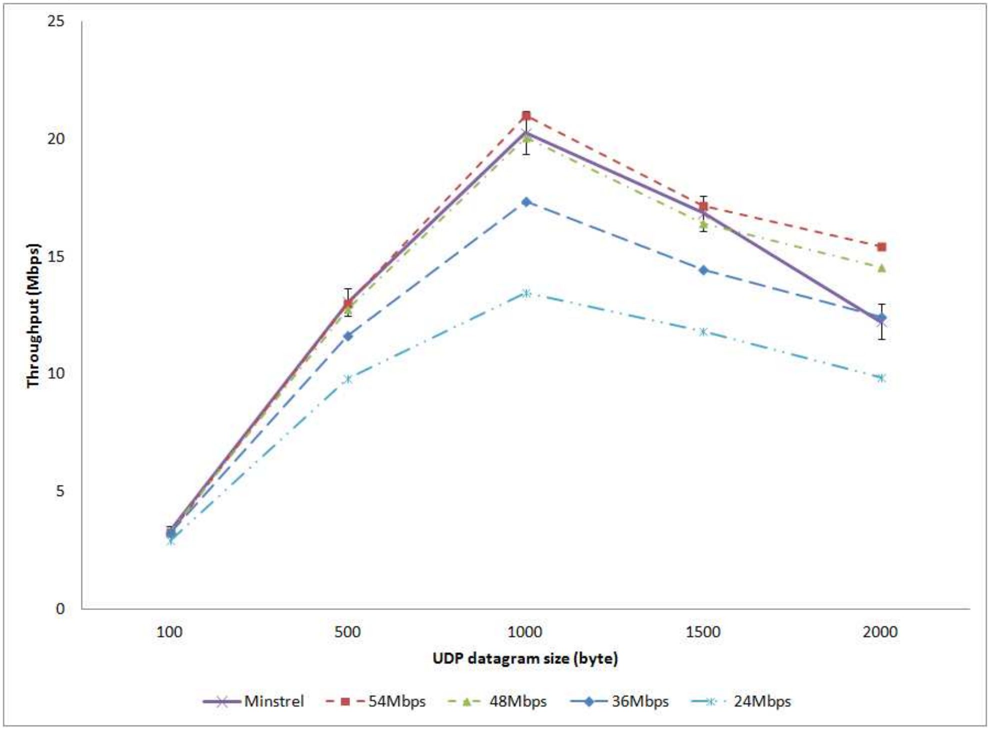


Figure 8.11: Throughput vs. datagram size (static channels) - power level: 9 dBm, AP location: L3

tive to fixed rates.

8.2.4 Datagram Size with Dynamic Channel Transmission

In this experiment, the datagram size is set from 100 bytes to 2000 bytes. The datagram size was fixed during the entire transmission. The transmission power, however, flipped between 17dBm and 1dBm (similar to Section 8.2.2). Figure 8.12, 8.13 and 8.14 shows the performance of *Minstrel* and fixed rates at location L2 with the duration of 1dBm set to 1, 5, and 9 seconds. It shows that in most cases 36/48Mbps outperform *Minstrel*. The performance at location L3 was similar 36Mbps consistently outperformed *Minstrel* with greater gain (results not shown). However, at location L1 *Minstrel* and 54Mbps have the similar performance and were the best performers (results not shown). This again indicates that *Minstrel* performs well with good channel conditions but struggled with the channels vulnerable to interferences.

8.2.5 Downlink Experiment Summary

In this section, the performance of *Minstrel* at the AP side and fixed rates with UDP traffic is compared in an 802.11g wireless testbed. The comparison takes into account a number of factors such as transmission power, AP locations, channel dynamics and datagram sizes. The comparison shows that *Minstrel* at the AP side performs well in many cases (comparable to the fixed rate that has the overall best throughput). This is particularly true with ″good″ or ″stable″ channel conditions. However, with ″poor″ or highly ″dynamic″ channel conditions *Minstrel* is often outperformed by certain fixed rates. This indicates the algorithm does not adapt well to channel dynamics. In addition, the performance of *Minstrel* has no direct link to the datagram sizes. Therefore, the uplink experiments are focused on the impacts on dynamic channel transmission.

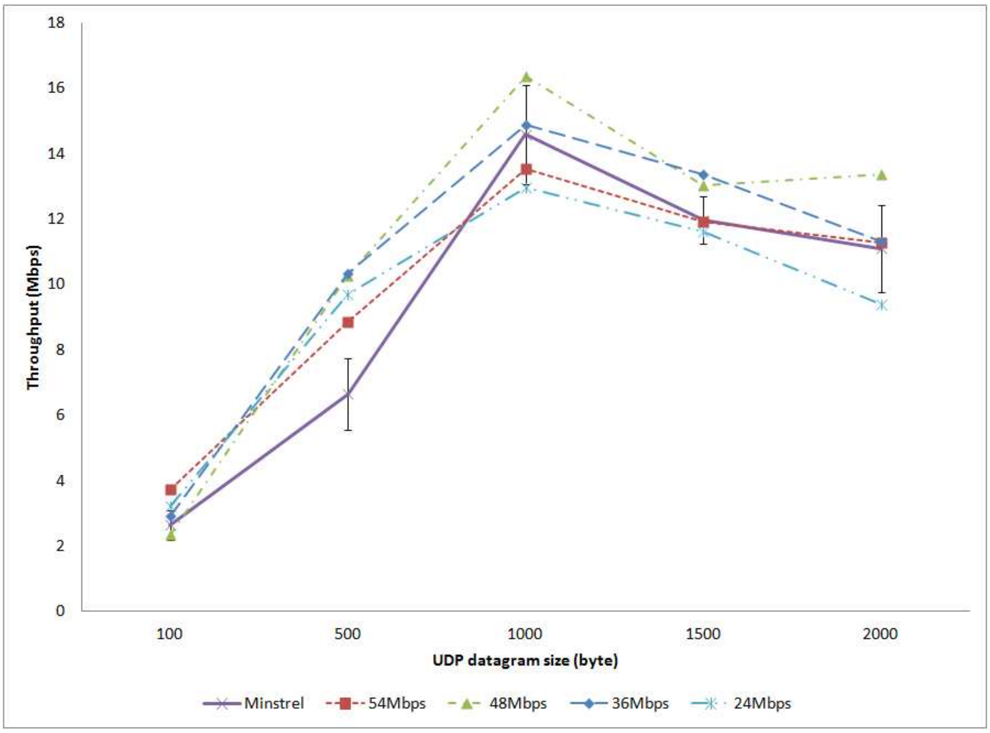


Figure 8.12: Throughput vs. datagram size (dynamic channels) - duration: 1 sec, AP location: L2

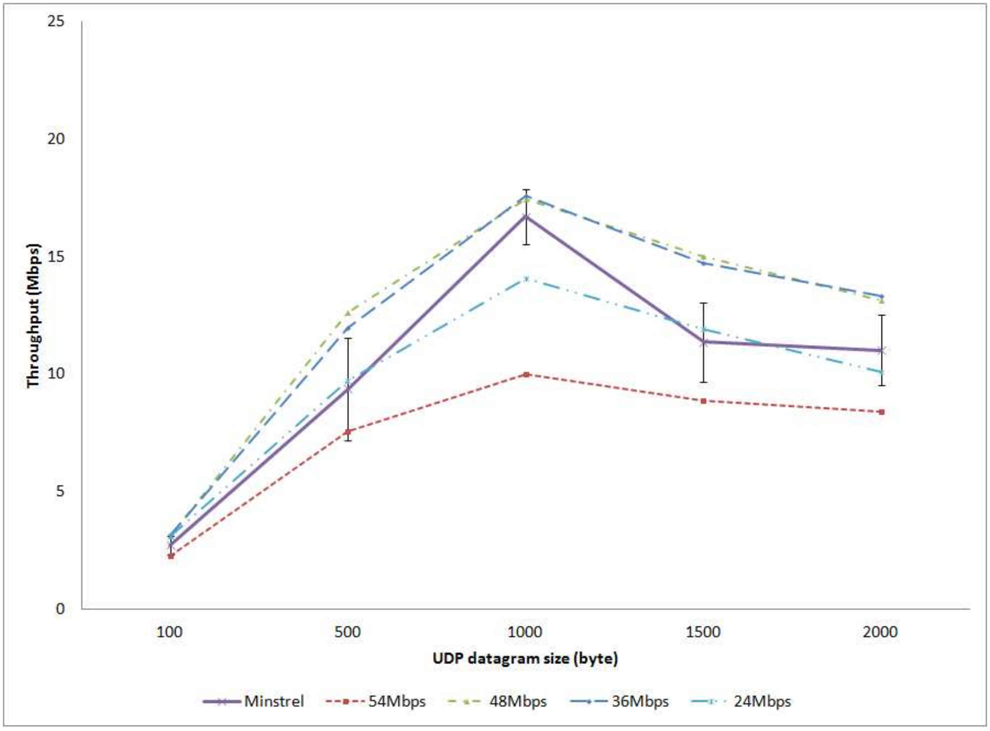


Figure 8.13: Throughput vs. datagram size (dynamic channels) - duration: 5 sec, AP location: L2

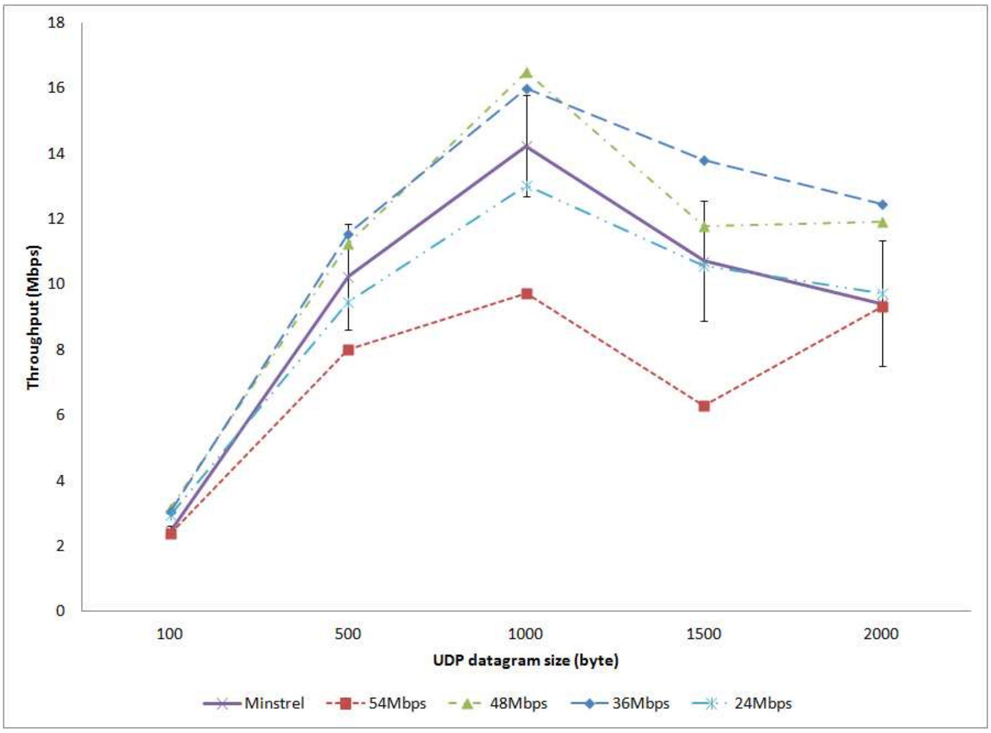


Figure 8.14: Throughput vs. datagram size (dynamic channels) - duration: 9 sec, AP location: L2

8.3 Uplink Experiments

This section shows the performance comparison between *Minstrel* rate adaptation algorithm and fixed rate with UDP uplink traffic. The last experiment indicates the datagram size has no direct impact on the performance of *Minstrel*. Moreover, The dynamic channel transmission from last experiment shows some of fixed rates outperform *Minstrel* rate adaptation. Therefore, more fine grained analyse is proceed in this experiment. The experiment is focused on three factors: static channel transmission, dynamic channel transmission and mobile transmission. Each part of experiment conducted with rate selection statistic maps to deeply evaluate the performance of *Minstrel* rate adaptation algorithm. The static channel transmission experiment provided a guide line of the channel environment. The mobile transmission explores and proves the upside and downside of the aggressiveness of *Minstrel* rate adaptation algorithm.

8.3.1 Static Channel Transmission

In this scenario, data is transmitted from client to server for 60 seconds, the transmission power is adjusted at the client side. The transmission power is kept steady for the duration of each experiment. High power level is used to emulate the scenarios where the channel of the client side is relatively resistant to interferences. Low power level is used to emulate the scenarios where the client side of the channel is relatively vulnerable to interferences. The offered traffic load is set to 35 Mbps.

Figure 8.15 shows how the throughput performance of *Minstrel* and the fixed rates changes for different transmission power levels. The throughput of *Minstrel* decreases as the transmission power decreased. In addition, *Minstrel* at the client side achieves comparable performance to the highest throughput had achieved by the best fixed rate at any power level, and this is corroborated by the two experiments in different locations. This experiment also shows that reducing the transmission power has a significant performance impact on the higher data rates, because more frames are lost due to the low quality of the wireless channel at low transmission power.

Figure 8.16 shows a percentage breakdown of the data rates that *Minstrel* choose to attempt to send packets at each different power level. When the transmission power is set to 17 dBm (i.e. the channel had high noise resistance), *Minstrel* mostly attempts to use the 54 Mbps and 48 Mbps rates, and only a small portion of attempts are made at the 36Mbps data. However, as the transmission power decreases, the channel becomes more vulnerable to interference, and *Minstrel* attempts to send packets using lower data rates. When the transmission power is set to 1 dBm, more than 70%

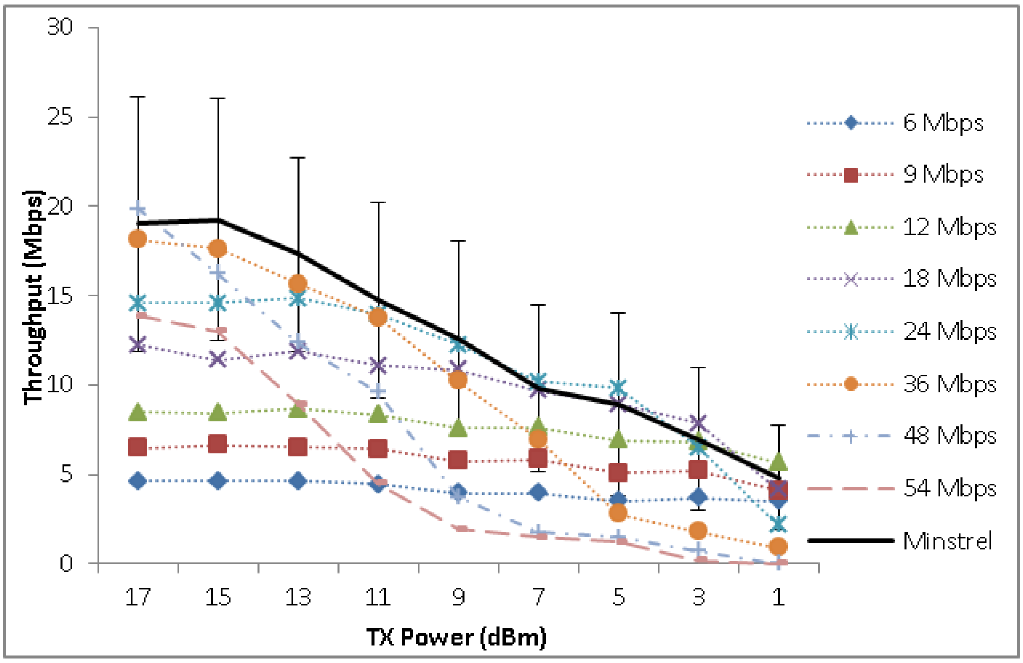


Figure 8.15: Throughput performance with static TX power - AP location: AP1

of attempts are made at the 12 Mbps and 18 Mbps rates (Figure 8.16). By the nature of *Minstrel* sampling algorithm, if the randomly selected data rate is higher than the current retry chain r0/c0 data rate, the retry chain r0/c0 is set to the randomly selected rate. Otherwise, the retry chain r0/c0 remains unchanged. Higher data rates always have a higher chance of being sampled. As Figure 8.16 shows, even when the transmission power is low there are still a small portion of packets that are sent at high data rates. This means that *Minstrel* at client side has enough statistics about the high data rates to rapidly adjust its data rate from low to high when the wireless channel condition improves.

Table 8.1 shows the data rates that achieves the best and 2nd best throughputs when fixed rates are used, and the rates most and 2nd most frequently attempted by *Minstrel*. The data rates in the bold font indicates the situation where *Minstrel*'s most or 2nd most frequently attempted rates do

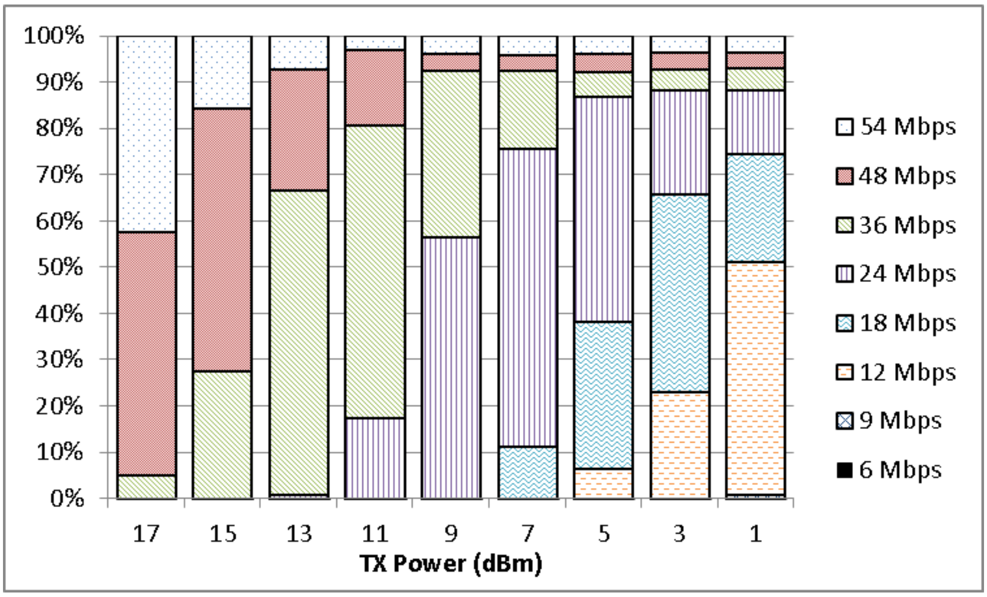


Figure 8.16: *Minstrel* rate selection statistics - AP location: AP1

not match the fixed data rates with the best or 2nd best throughputs. According to the table, the data rates that give the best throughputs are also the same data rate that *Minstrel* most frequently attempted to use in most cases, except when the transmission power is set to 15 dBm. This shows *Minstrel* at client side selects the best rate correctly, however often the 2nd best throughput rates are not used by *Minstrel*. Instead, a higher rate is selected due to its aggressive sampling algorithm.

This experiment reveals that *Minstrel* at client side worked very well in relatively good or static wireless channel conditions. The throughput of *Minstrel* is comparable to the highest throughput had achieved by fixed rates at any transmission power level. This means that in stable channel conditions, *Minstrel*'s tendency to sample and select higher data rates doesn't incur significant performance cost.

TX power (dBm)	Fixed Rate		*Minstrel*	
	Best rate	2nd best rate	Most frequently attempted rate	2nd most frequently attempted rate
17	48	36	48	**54**
15	36	48	**48**	**36**
13	36	24	36	**48**
11	36	24	36	24
9	24	18	24	**36**
7	24	18	24	**36**
5	24	18	24	18
3	18	12	18	12
1	12	18	12	18

Table 8.1: Best rate vs. most frequently attempted rate (AP1)

8.3.2 Dynamic Channel Transmission

In this section, the experiment is conducted in dynamic channel conditions. A dynamic wireless channel is created by flipping the transmission power between 17 dBm and 1 dBm. The transmission power stays at 17 dBm for 2 sec before flipping to 1 dBm. The dynamism of the channel is adjusted by varying the duration that the power level stays low between 2 and 10 seconds, and each test runs for 60 seconds. The results of the previous experiment show that *Minstrel* prefers to select higher data rates in static channel conditions. This behaviour can become costly if the quality of the wireless channel changes drastically.

Figure 8.17 shows the performance of fixed rates and *Minstrel* in relation to the duration of the low transmission power at location AP2. It shows that as the duration of the low power increases, the performance of *Minstrel* and the high fixed rates (54, 48, and 36 Mbps) suffers significantly. This poor performance is expected for the high rates, as they suffers from more packet losses with low transmission power. The performance of *Minstrel* and high rates stabilise after the duration of low power level increases past 5/6 seconds. In contrast, the performance of the low rates is relatively consistent from short to long duration. The figure clearly shows that fixed rates of 12/18Mbps outperform *Minstrel* when the low power duration is more than 4 seconds.

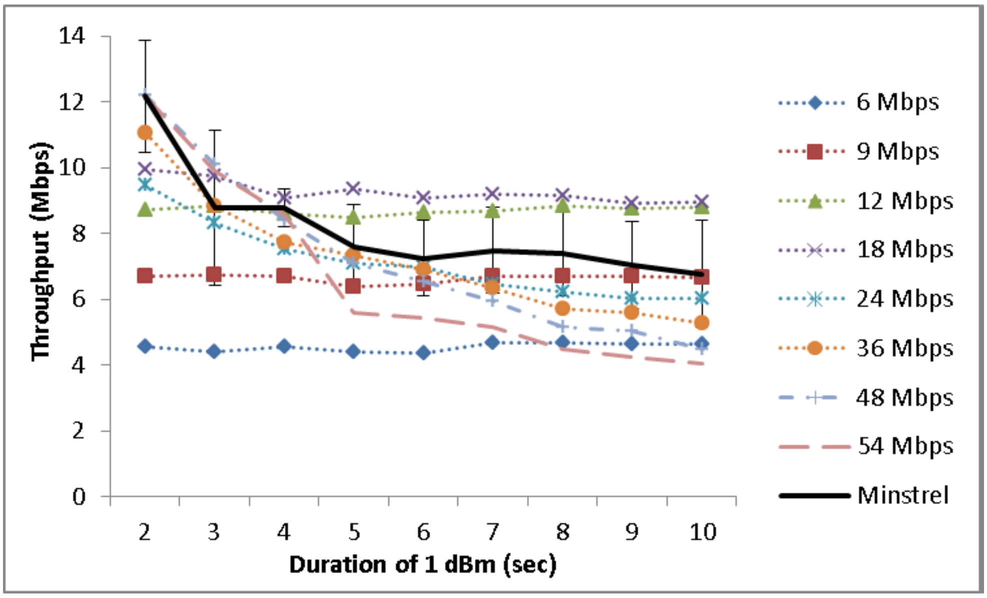


Figure 8.17: Throughput with dynamic TX power - AP location: AP2

The rate breakdown statistics for *Minstrel* are shown in Figure 8.18. In Figure 8.17, The observation is the best throughput is achieved by the 54/48Mbps rates when the duration of low power is low (2/3 seconds). When the duration is 4 to 10 seconds, the rates at 12/18 Mbps appear to have the best throughput. In Figure 8.18, it can be seen that *Minstrel* most frequently selected 54 Mbps over the other rates for transmission. This made *Minstrel* comparable to the best rates (54/48 Mbps) in throughput performance, when the duration of low power is short. As the duration increased, *Minstrel* uses the 54 Mbps rate less frequently and the 18Mbps rate is used more frequently. This is as expected. However *Minstrel* does not use the 18/12 Mbps rates enough, which leads to the poor performance. The rightmost bar in Figure 8.18 shows the case when the cycle consists of high power for 2 seconds and low power for 10 seconds, in other words a 5:1 ratio of low power duration to high power duration. However the figure shows that *Minstrel* attempts low and high rates almost equally, when it clearly should use low rates a much greater percentage of the time.

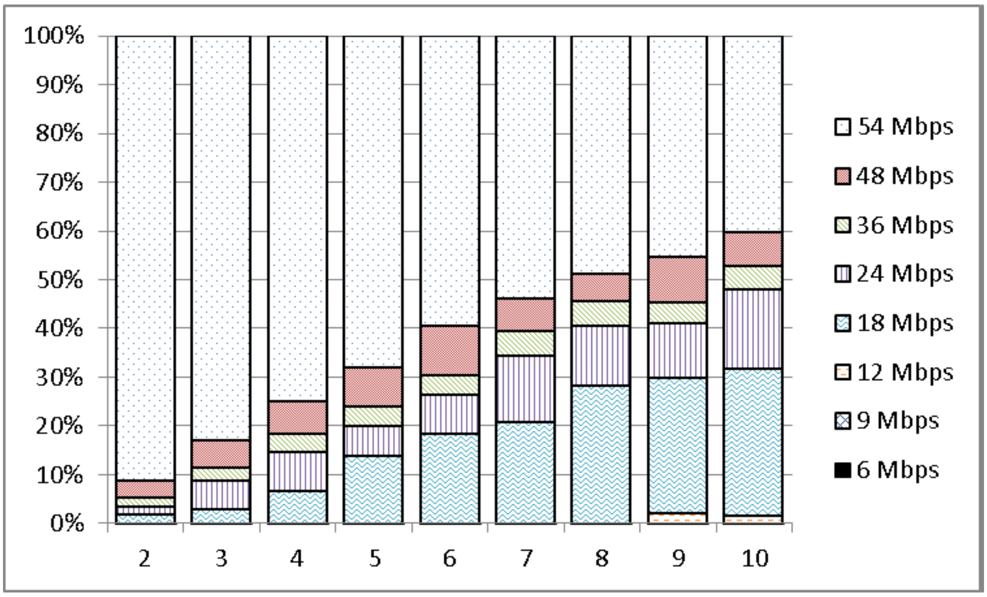


Figure 8.18: *Minstrel* rate selection statistics - AP location: AP2

8.3.3 Mobile Transmission

The previous experiment has shown that a dynamic wireless environment causes serious performance issues for *Minstrel*, because the dynamic channel prevents the algorithm from predicting the channel conditions accurately. Very often, *Minstrel* appears to be too aggressive, selecting high rates when low rates are needed. Whilst this means that the throughput of *Minstrel* can be recovered fast when the channel conditions improve, this technique can be costly in the scenario where channel conditions change drastically.

In this experiment, a further study is applied on the behaviour of *Minstrel* in dynamic wireless environments the channel conditions change gradually from ″bad″ to ″good″ or from ″good″ to ″bad″. The transmission power increases or decreases linearly. The duration of time between power level updates is varied from 2 seconds to 5 seconds, and the transmission power increases or decreases by 2 dBm at every update time until the transmission power is set to either 17 dBm or 1 dBm.

Figure 8.19 the throughput of *Minstrel* and fixed rates as the transmission power decreases from 17 dBm to 1 dBm. The 18 and 24 Mbps fixed rates outperform *Minstrel*, although the performance difference between the 18 and 24 Mbps fixed rates and *Minstrel* become less significant as the duration of updating transmission power increases. When the update duration is short the channel is dynamic, but as the update duration lengthens the channel becomes relatively static. *Minstrel* updates its retry chain every 100ms, which means it updates 20 times in a 2 second duration and 50 times in a 5 second duration. This means that *Minstrel* has a much longer time to adapt to the changes in channel conditions when the update duration is longer.

Figure 8.20 and Figure 8.21 are two examples of *Minstrel*'s rate selection breakdown when the transmission power update duration is 2 seconds and 5 seconds respectively. As expected, *Minstrel* attempts to use low data rates more often as the transmission power decreases. However, fixed rates still outperform *Minstrel*, suggesting that *Minstrel* makes incorrect rate selection as the channel quality deteriorates. It can also be seen that *Minstrel* uses low rates more often with update duration set to 5 seconds than 2 seconds. This explains the smaller performance gap between fixed rates and *Minstrel*, when the update duration is set to 5 seconds (Figure 8.19).

Table 8.2 and 8.3 show the rate most frequently used by *Minstrel* (outside the brackets), and the data rate that achieves the highest ideal through-

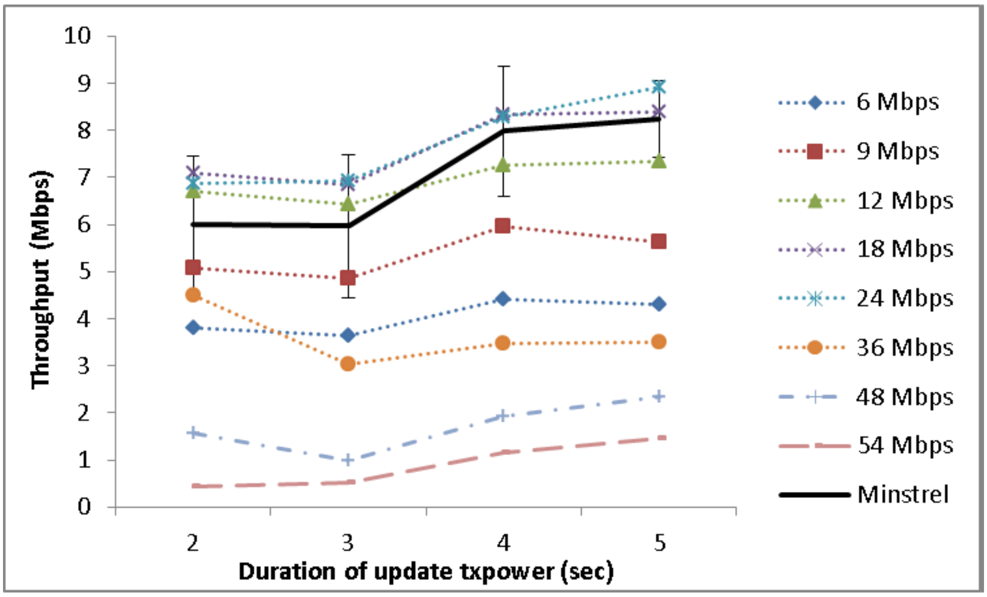


Figure 8.19: Throughput with TX power decreasing from high to low - AP location: AP1

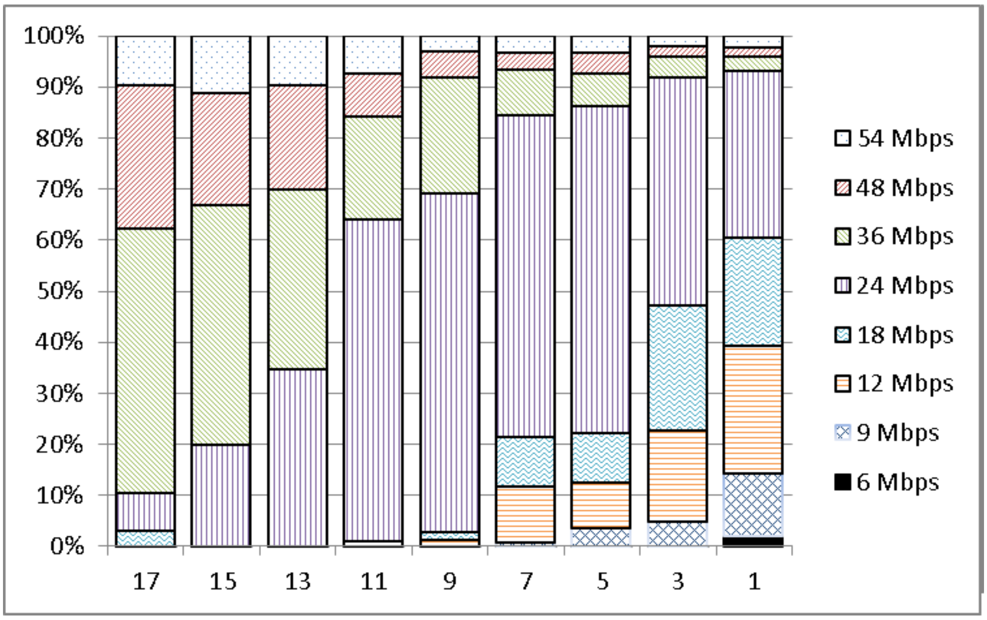


Figure 8.20: *Minstrel* rate selection - Duration of TX power update: 2 sec

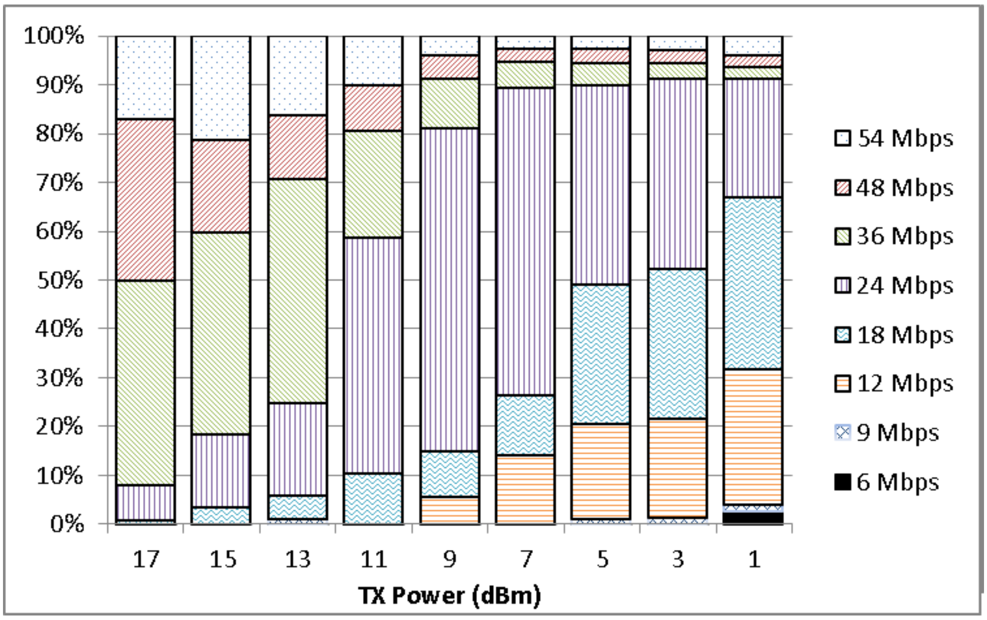


Figure 8.21: *Minstrel* rate selection - Duration of TX power update: 5 sec

put (inside the brackets) for each transmission power update duration at different power levels. The ideal throughput is calculated using the following function:

$$data\_rate \times \frac{number\_of\_successful\_transmission}{total\_number\_of\_transmission} \tag{8.1}$$

Minstrel should strive to use the rate with the highest throughput most often, however during these experiments it often fails to do so, as indicated by the bold entries in Table 8.2. *Minstrel* often makes incorrect rate choices when the power level is low and when the transmission power update duration is short. Again, *Minstrel* tends to select a rate higher than the optimal rate. However, as the update duration increases, *Minstrel*'s selection of rates become more accurate. This explains why *Minstrel*'s performance improves with the update duration.

Figure 8.22 shows the throughput of fixed rates and *Minstrel* when the transmission power increases from 1 dBm to 17 dBm, for different trans-

	2	3	4	5
17	36 (36)	36 (36)	36 (36)	36 (36)
15	36 (36)	36 (24)	36 (36)	36 (36)
13	**36 (24)**	24 (24)	36 (36)	36 (36)
11	24 (24)	24 (24)	24 (24)	24 (24)
9	**24 (12)**	**24 (12)**	24 (24)	24 (24)
7	**24 (12)**	**24 (12)**	**24 (12)**	24 (24)
5	**24 (12)**	**24 (12)**	**24 (12)**	**24 (12)**
3	**24 (12)**	**24 (12)**	**24 (12)**	12 (12)
1	**24 (12)**	**24 (12)**	**18 (12)**	12 (12)

Table 8.2: Rate most frequently used by *Minstrel* (outside the brackets) vs. rate with the highest ideal throughput (inside the brackets) - High TX power to low TX power

mission power update durations varying from 2 to 5 seconds. *Minstrel* has superior performance over the fixed rates in all cases, and the throughput difference between the best fixed rates and *Minstrel* becomes more significant as the transmission power update duration becomes longer. This is the positive outcome of the algorithm which aggressively samples higher rates over lower rates. This means that as soon as the channel conditions improves, *Minstrel* is quickly able to take advantage and use higher rates. Table 8.3 again shows the rate most often used by *Minstrel* and the rate with the highest ideal throughput in each situation. As the results shown in this scenario when the channel conditions are improving, *Minstrel* becomes more accurate in choosing the rate with the highest ideal throughput. Moreover, as the update duration increases, *Minstrel* has more time to adapt and is able to choose the correct rate more accurately.

	2	3	4	5
1	**24 (18)**	**24 (18)**	**24 (18)**	12 (12)
3	**24 (18)**	18 (18)	18 (18)	18 (18)
5	24 (24)	18 (18)	18 (18)	24 (24)
7	24 (24)	**18 (24)**	24 (24)	24 (24)
9	24 (24)	24 (24)	36 (36)	36 (36)
11	36 (36)	36 (36)	36 (36)	36 (36)
13	36 (36)	36 (36)	36 (36)	36 (36)
15	36 (36)	36 (36)	48 (48)	48 (48)
17	48 (48)	48 (48)	48 (48)	48 (48)

Table 8.3: Rate most frequently used by *Minstrel* (outside the brackets) vs. rate with the highest ideal throughput (inside the brackets) - Low TX power to high TX power

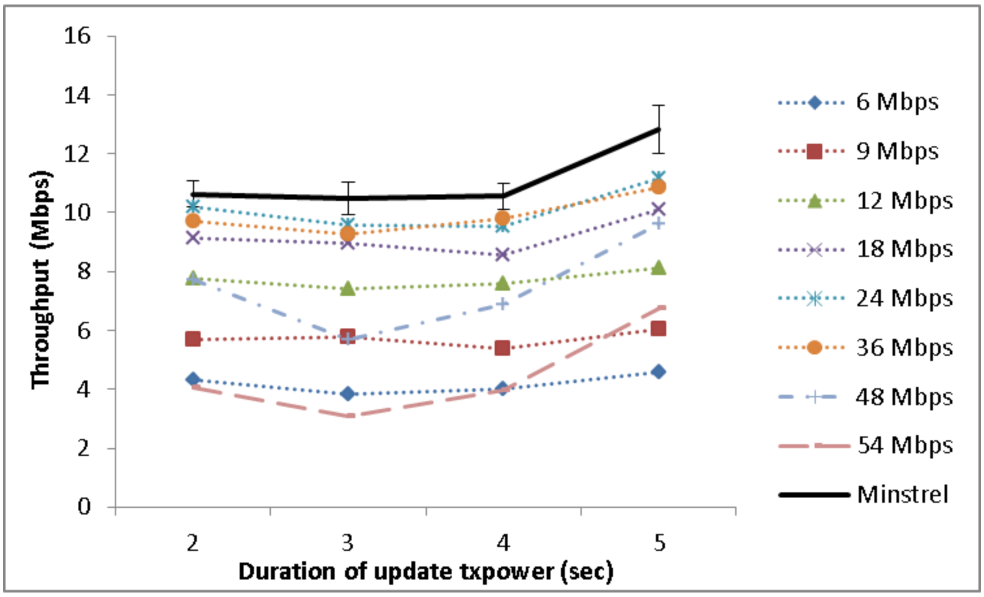


Figure 8.22: Throughput with TX power increasing from low to high - AP location: AP2

8.3.4 Uplink Experiment Summary

Minstrel rate adaptation algorithm is widely used in the client side (laptop, smart phone, etc) and server side (AP) in popular wireless drivers such as *MadWifi*, *Ath5k* and *Ath9k*, and appears to have far better performance than *SampleRate*, which is frequently used for performance evaluation of newly designed rate adaptation algorithms. This section shows the performance comparison between *Minstrel* rate adaptation algorithm and fixed rate with UDP uplink traffic. Results shows that *Minstrel* can achieve a performance comparable to the best performance that fixed rates can possibly offer, when the wireless channel condition is relatively ″static″. With ″static″ channels, *Minstrel* can usually identify the best rates to use. However, in the presence of ″dynamic″ channel conditions, certain fixed rates are able to outperform *Minstrel* significantly. This is ude to *Minstrel*'s tendency to select rates that are too high for the current channel conditions. It also was discovered that *Minstrel* can have poor performance when the wireless signal strength deteriorates from high to low. On the other hand, *Minstrel* outperforms fixed rate if the signal strength increases from low to high. Again, this is due to *Minstrel*'s tendency to select high rates. The theoretical study of *Minstrel* in the chapter 7 showed that a high data rate always had large maximum retransmission counter and had high priority to be used in retry chain, which results that *Minstrel* intends to give higher data rates more chance to be used for transmission. This behavior is ″bad″ when the channel conditions are deteriorating, however it can provide good ″predictions″ when the channel conditions are improving.

Chapter 9

Conclusions

This thesis compares the performance of MikroTik rate adaptation algorithm and *Minstrel* rate adaptation algorithm to fixed rate respectively over an 802.11g wireless testbed. To better understand the performance issues, in the comparison, a number of factors such as AP transmission power, AP/client distance, channel dynamic have been taken into account. The goal of this study is to enhance the understanding of these two common rate adaptation algorithms, and to provide a direction for the future improvement of rate adaptation algorithm. The combination of theoretical studies and field experiment studies were conducted in this research.

9.1 Summary of MikroTik Experiment

MikroTik routers are widely used in New Zealand and around the world. MikroTik's proprietary rate adaptation algorithm is used as the default setting. The algorithm uses number of successful or failed transmission to determine whether the current data rate is optimal. In our experiment, UDP sending rate, AP transmission power level, AP/client distance and RTT for TCP traffic are the experiment variable to compare the performance of MirkoTik rate adaptation algorithm against fixed data rates. It shows that fixed rate outperforms the MikroTik rate adaptation in most cases, especially when high data rates are used. The rate adaptation can perform slightly better than fixed rate when fixed rate is forced to use low data rates. It also appears that the performance gain by fixed rate over the MikroTik rate adaptation is correlated to AP power level. For TCP, the performance gain does not differ much with different factors. Another observation is that the rate adaptation algorithm used in the AP appears to be too conservative. For a given condition the algorithm could have chosen a higher data rate.

9.2 Summary of Minstrel Experiment

The determination of a *Minstrel*'s retry chain is depended on the value of the EWMA throughputs and EWMA probability of successful transmission. Moreover, the throughput performance of sending a packet relies to data rates and the number of retransmission. The more retransmission causes lower throughput performance. *Minstrel* rate adaptation algorithm has four different data rates in a retry chain, each data rate has approximately 6000 microseconds to use for sending a packet before *Minstrel* decided to use the next available data rate to send the packet. A higher data rate always has larger maximum retransmission counter and higher chance to be selected than lower data rates. Therefore, *Minstrel* has faster recovery from having a poor wireless channel to a good wireless channel. The characteristic of the fast recovery also raises a concern when a wireless channel condition moves from good to bad, as we known that a low throughput can be caused by selecting a high data rate in a weak wireless connection environment. Moreover, retransmission counter and throughput for each data rate is calculated based on an ideal transmission time for each rate. The statistic throughput of *Minstrel* cannot represent the real performance of each data rate in a real environment.

Minstrel field experiment is divided into two parts: downlink experiment and uplink experiment. Both experiments compare the performance of *Minstrel* to fixed data rates. Number of factors such as transmission power, AP locations, and channel dynamics were taken into account in both parts of experiment. In the downlink experiment, datagram sizes are also compared as key factor that can impact on the performance of the *Minstrel*. However, the performance of *Minstrel* has no direct link to the datagram size according to the experiment. On the other hand, *Minstrel* has comparable performance to the fixed rate in a ″good″ or ″stable″ channel condition. However, *Minstrel* is often outperformed by certain fixed rate in a ″poor″ or highly ″dynamic″ channel conditions. Therefore, the uplink experiment is focused on exploring the performance of *Minstrel* in a static or dynamic channel condition. The experiment shows that *Minstrel* has a tendency to select high data rate even though the channel condition is not able to support such high rate. It also was discovered that *Minstrel* can have poor performance when the wireless signal strength deteriorates from high to low. However, *Minstrel* outperforms fixed rate if the signal strength increases from low to high. Again, this is due to *Minstrel*'s tendency to select high rates. The theoretical study also showed that a high data rate always had large maximum retransmission counter and had high priority to be used in retry chain, which results that *Minstrel* intends to give higher data rates more chance to be used for transmission. This behaviour is ″bad″ when the channel conditions are deteriorating, however it can provide ″good″ predictions when the channel conditions are improving.

9.3 Overall Contributions

The contributions of this thesis are:

- Establishing an IEEE 802.11 wireless testbed and requirements. The wireless testbed and devices, experiment execution requirements are carefully documented for the future use. The OpenWRT operating system installation manual is created for the operating system of the MikroTik RB411 router.

- Developing a wireless testbed automated controlling program. In order to have accurate test results, all the experiments should be precisely executed according to experiment requirements. An automated controlling program is designed and implemented to reduce experiment errors and to provide more repeatable experiment results.

- Comparing MikroTik rate adaptation algorithm and exploring the issues of the algorithm. The comparsion of MikroTik rate adaptation to fixed rates is carried out over many different factors. The experiment shows that MikroTik rate adaptation algorithm outperforms by fixed rate in many cases. The issue is that the algorithm does not select a higher data rate when a channel condition is good enough to support higher rates.

- Creating a documentation of *Minstrel* rate adaptation algorithm. *Minstrel* rate adaptation algorithm is widely used around the world. However, the documentation of how *Minstrel* works is limited. The feature of *Minstrel* rate adaptation algorithm and working procedure are carefully studied and documented for the future studies.

- Comparing *Minstrel* rate adaptation algorithm and exploring the issues of the algorithm. *Minstrel* rate adaptation algorithm is compared to fixed rate in both uplink and downlink environment. The theoretical study and field experiment study has consistently proved that *Minstrel* rate adaptation prefers selecting a higher data rate. This characteristic of *Minstrel* is good for fast recovery when channel condition is from ″bad″ to ″good″. However, it reduces the performance when the channel condition is from ″good″ to ″bad″.

9.4 Future work

The future works in this area are:

- Evaluate the performance of *Minstrel* in a testbed which contains multiple client nodes.

- Investigate how *Minstrel* is working in a testbed which has MIMO antenna supporting.

- Design and Implement a new rate adaptation algorithm which carries the fast recovery feature from *Minstrel*, but also selecting correct data rate when channel exteriors.

- Evaluate the new design of rate adaptation algorithm in IEEE 802.11g testbed

- Evaluate the new design in a testbed which has MIMO antenna supporting.

Bibliography

[1] J. Heiskala and J. Terry, *OFDM Wireless LANs: A Theoretical and Practical Guide*. SAMS, 2001.

[2] W. Yin, P. Hu, J. Indulska, and K. Bialkowski, "Performance of MAC80211 Rate Control Mechanisms," in *Proceedings of the 14th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems*, pp. 427–436, ACM, 2011.

[3] WiFi. What is it?, [Online]. Available: http://www.wilcorpinc.com/wifi_history.htm.

[4] Cuadro nacional de Atribucin de Frecuencias CNAF, [Online]. Available: http://www.mityc.es/Telecomunicaciones/Secciones/Espectro/cnaf.

[5] Evolution du rgime d'autorisation pour les RLAN, [Online]. Available: http://www.arcep.fr/uploads/tx_gspublication/evol-rlan-250703.pdf.

[6] X. Chen, D. Qiao, J. Yu, and S. Choi, "Probabilistic-Based Rate Adaptation for IEEE 802.11 WLANs," in *Global Telecommunications Conference, 2007. GLOBECOM'07. IEEE*, pp. 4904–4908, IEEE, 2007.

[7] Q. Xia and M. Hamdi, "Smart Sender: a Practical Rate Adaptation Algorithm for Multirate IEEE 802.11 WLANs," *Wireless Communications, IEEE Transactions on*, vol. 7, no. 5, pp. 1764–1775, 2008.

[8] J. Zhang, K. Tan, J. Zhao, H. Wu, and Y. Zhang, “A Practical SNR-Guided Rate Adaptation,” in *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pp. 2083–2091, IEEE, 2008.

[9] 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, pp. 278 – 292, oct-dec 2002.

[10] J. Camp and E. Knightly, “Modulation Rate Adaptation in Urban and Vehicular Environments: Cross-Layer Implementation and Experimental Evaluation,” *Networking, IEEE/ACM Transactions on*, vol. 18, pp. 1949 –1962, dec. 2010.

[11] S. Kim, S. Choi, D. Qiao, and J. Kim, “Enhanced Rate Adaptation Schemes with Collision Awareness,” *NETWORKING 2007. Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet*, pp. 1179–1182, 2007.

[12] 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, pp. 1108–1113, IEEE, 2003.

[13] I. Haratcherev, K. Langendoen, R. Lagendijk, and H. Sips, “Hybrid Rate Control for IEEE 802.11,” in *In Proceedings of ACM MobiWac*, pp. 10–18, ACM, 2004.

[14] I. Haratcherev, J. Taal, K. Langendoen, R. Lagendijk, and H. Sips, “Fast 802.11 Link Adaptation for Real-Time Video Streaming by Cross-Layer Signaling,” in *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, pp. 3523 – 3526 Vol. 4, may 2005.

[15] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris, “Link-Level Measurements from an 802.11b Mesh Network,” *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, pp. 121–132, 2004.

[16] D. Qiao and S. Choi, "Goodput Enhancement of IEEE 802.11 a Wireless LAN via Link Adaptation," in *Communications, 2001. ICC 2001. IEEE International Conference on*, vol. 7, pp. 1995–2000, IEEE, 2001.

[17] 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, 2012.

[18] A. Vlavianos, L. Law, I. Broustis, S. Krishnamurthy, and M. Faloutsos, "Assessing Link Quality in IEEE 802.11 Wireless Networks: Which Is the Right Metric?," in *Personal, Indoor and Mobile Radio Communications, 2008. PIMRC 2008. IEEE 19th International Symposium on*, pp. 1–6, IEEE, 2008.

[19] D. Xia and Q. Fu, "An Experimental Study on Performance Comparison of Rate Adaptation and Fixed Rate in ieee 802.11g," in *Australasian Telecommunication Networks and Applications Conference (ATNAC), 2011*, pp. 1 –6, nov. 2011.

[20] D. Xia, J. Hart, and Q. Fu, "On the performance of rate control algorithm minstrel," in *Personal Indoor and Mobile Radio Communications (PIMRC), 2012 IEEE 23rd International Symposium on*, pp. 406 –412, sept. 2012.

[21] D. Xia, J. Hart, and Q. Fu, "Evaluation of the Minstrel Rate Adaptation Algorithm in IEEE 802.11g WLANs," in *IEEE International Conference on Communications (ICC)*, 2013.

[22] "IEEE Standard for Information Technology- Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements-Part 11: Wireless Lan Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Std 802.11-1997*, pp. 1 –445, 1997.

[23] H. Zimmermann, “OSI Reference Model–The ISO Model of Architecture for Open Systems Interconnection,” *Communications, IEEE Transactions on*, vol. 28, pp. 425 – 432, apr 1980.

[24] J. Field, “Logical Link Control,” in *Proceedings, IEEE INFOCOM’86*, 1986.

[25] I. C. S. L. M. S. Committee *et al.*, “Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications,” 1997.

[26] R. Metcalfe and D. Boggs, “Ethernet: Distributed Packet Switching for Local Computer Networks,” *Communications of the ACM*, vol. 19, no. 7, pp. 395–404, 1976.

[27] “IEEE Standard for Information Technology- Telecommunications and Information Exchange Between Systems-Local and Metropolitan Area Networks-Specific Requirements-Part 3: Carrier Sense Multiple Access With Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications - Section Five,” *IEEE Std 802.3-2008 (Revision of IEEE Std 802.3-2005)*, pp. 1 –615, 26 2008.

[28] IEEE 802.11 Wireless Local Area Networks - The Working Group for WLAN Standards, [Online]. Available: http://www.ieee802.org/11/.

[29] P. Acharya, A. Sharma, E. Belding, K. Almeroth, and K. Papagiannaki, “Rate Adaptation in Congested Wireless Networks through Real-Time Measurements,” *Mobile Computing, IEEE Transactions on*, vol. 9, pp. 1535 –1550, nov. 2010.

[30] S. Gollakota and D. Katabi, *Zigzag Decoding: Combating Hidden Terminals in Wireless Networks*, vol. 38. ACM, 2008.

[31] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge university press, 2005.

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

[33] M. Carroll and T. Wysocki, "Fading Characteristics for Indoor Wireless Channels at 5 GHz Unlicensed Bands," in *Mobile Future and Symposium on Trends in Communications, 2003. SympoTIC'03. Joint First Workshop on*, pp. 102–105, IEEE, 2003.

[34] M. Luby, L. Vicisano, J. Gemmell, L. Rizzo, M. Handley, and J. Crowcroft, "The Use of Forward Error Correction (fec) in Reliable Multicast," tech. rep., RFC 3453, December, 2002.

[35] S. Lin, D. Costello, and M. Miller, "Automatic-repeat-request error-control schemes," *Communications Magazine, IEEE*, vol. 22, no. 12, pp. 5–17, 1984.

[36] G. Vannucci, "Direct Sequence Spread Spectrum (DSSS) Communications System with Frequency Modulation Utilized to Achieve Spectral Spreading," 1992. US Patent 5,150,377.

[37] L. Cimini Jr, "Analysis and Simulation of a Digital Mobile Channel Using Orthogonal Frequency Division Multiplexing," *Communications, IEEE Transactions on*, vol. 33, no. 7, pp. 665–675, 1985.

[38] "IEEE 802.11a, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Higher-Speed Physical Layer in the 5 GHz Band," *IEEE Std 802.11a-1999*, 1999.

[39] "IEEE 802.11g, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Further Higher Data Rate Extension in the 2.4 GHz Band," *IEEE Std 802.11g-2006*, 2006.

[40] "IEEE 802.11n, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Enhancements for Higher Throughput," *IEEE Std 802.11n-2009*, pp. 1 –565, 2009.

[41] "IEEE 802.11ac, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Enhancements for Very High Throughput for Operation in Bands below 6GHz," *IEEE Std 802.11ac-draft*.

[42] H. Sanderford Jr, "Binary Phase Shift Keying Modulation System and/or Frequency Multiplier," 1998. US Patent RE35,829.

[43] S. Tatu and R. Bosisio, "Quadrature Phase Shift Keying (qpsk)," *Encyclopedia of RF and Microwave Engineering*, 2005.

[44] W. Webb and L. Hanzo, *Modern Quadrature Amplitude Modulation: Principles and Applications for Fixed and Wireless Channels: One*. IEEE Press-John Wiley, 1994.

[45] S. Yi, S. Chun, Y. Lee, S. Park, and S. Jung, "Medium Access Control (MAC)," *Radio Protocols for LTE and LTE-Advanced*, pp. 119–149, 2012.

[46] B. O'hara and A. Petrick, *IEEE 802.11 Handbook: a Designer's Companion*. Standards Information Network, 2005.

[47] E. Hapeman, W. Zeuch, J. Crandall, S. Carioti, S. Barclay, and C. Underkoffler, "Telecom Glossary 2000–American National Standard T1," 2001.

[48] B. Sklar, *Digital Communications*, vol. 2. Prentice Hall, 2001.

[49] T. Rappaport, *Wireless Communications: Principles and Practice*. IEEE press, 1996.

[50] A. Kamerman and L. Montehban, "WaveLAN II: A High-performance Wireless LAN for the Unlicensed Band," *Bell Labs Technical Journal*, Summer 1997.

[51] Y. Rong, A. Teymorian, L. Ma, X. Cheng, and H. Choi, "A Novel Rate Adaptation Scheme for 802.11 Networks," *Wireless Communications, IEEE Transactions on*, vol. 8, no. 2, pp. 862–870, 2009.

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

[53] ONOE Rate Adaptation Algorithm, [Online]. Available: http://madwifi-project.org/browser/madwifi/trunk/ath_rate/onoe.

[54] J. Bicket, "Bit-rate Selection in Wireless Networks," Master's thesis, 2005.

[55] Minstrel Rate Adaptation Algorithm Documentation, [Online]. Available: http://madwifi-project.org/browser/madwifi/trunk/ath_rate/minstrel/minstrel.txt.

[56] PID Rate Adaptation Algorithm, [Online]. Available: http://linuxwireless.org/en/developers/Documentation/mac80211.

[57] P. Chevillat, J. Jelitto, A. Barreto, and H. Truong, "A Dynamic Link Adaptation Algorithm for IEEE 802.11a Wireless LANs," in *Communications, 2003. ICC'03. IEEE International Conference on*, vol. 2, pp. 1141–1145, IEEE, 2003.

[58] D. Qiao and S. Choi, "Fast-Responsive Link Adaptation for IEEE 802.11 WLANs," in *Communications, 2005. ICC 2005. 2005 IEEE International Conference on*, vol. 5, pp. 3583 – 3588 Vol. 5, may 2005.

[59] G. Holland, N. Vaidya, and V. Bahl, "A Rate-Adaptive MAC Protocol for Multihop Wireless Networks," *ACM MOBICOM*, 2001.

[60] G. Judd, X. Wang, and P. Steenkiste, "Low-Overhead Channel-Aware Rate Adaptation," in *ACM MobiCom*, pp. 354 – 357, 2007.

[61] M. Heusse, F. Rousseau, R. Guillier, and A. Duda, "Idle Sense: an Optimal Access Method for High Throughput and Fairness in Rate Diverse Wireless LANs," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 4, pp. 121–132, 2005.

[62] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: A Media Access Protocol for Wireless LAN's," in *ACM SIGCOMM Computer Communication Review*, vol. 24, pp. 212–225, ACM, 1994.

[63] F. Tobagi and L. Kleinrock, "Packet Switching in Radio Channels: Part II–The Hidden Terminal Problem in Carrier Sense Multiple-Access and the Busy-Tone Solution," *Communications, IEEE Transactions on*, vol. 23, no. 12, pp. 1417–1433, 1975.

[64] P. Karn, "MACA-A New Channel Access Method for Packet Radio," in *ARRL/CRRL Amateur radio 9th computer networking conference*, vol. 140, pp. 134–140, 1990.

[65] C. Fullmer and J. Garcia-Luna-Aceves, *Solutions to Hidden Terminal Problems in Wireless Networks*, vol. 27. ACM, 1997.

[66] C.-T. Tai, "Complementary Reciprocity Theorems in Electromagnetic Theory," *Antennas and Propagation, IEEE Transactions on*, vol. 40, pp. 675 –681, jun 1992.

[67] Wireless Channel Bonding, [Online]. Available: http://www.intel.com/support/wireless/sb/CS-025343.htm.

[68] R. Gummadi and H. Balakrishnan, "Wireless Networks Should Spread Spectrum Based on Demands," in *In HotNets 08*, Citeseer, 2008.

[69] T. Moscibroda, R. Chandra, Y. Wu, S. Sengupta, P. Bahl, and Y. Yuan, "Load-Aware Spectrum Distribution in Wireless LANs," in *Network Protocols, 2008. ICNP 2008. IEEE International Conference on*, pp. 137–146, IEEE, 2008.

[70] M. Vutukuru, H. Balakrishnan, and K. Jamieso, "Cross-Layer Wireless Bit Rate Adaptation," in *ACM SIGCOMM*, 2009.

[71] J. Camp and E. Knightly, "Modulation Rate Adaptation in Urban and Vehicular Environments: Cross-Layer Implementation and Experimental Evaluation," in *Proceedings of the 14th ACM international conference on Mobile computing and networking*, pp. 315–326, ACM, 2008.

[72] B. Sadeghi, V. Kanodia, A. Sabharwal, and E. Knighlty, "Opportunistic Media Access for Multirate Ad Hoc Networks," *ACM MOBICOM*, 2002.

[73] "IEEE 802.11b, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Higher-Speed Physical Layer Extension in the 2.4 GHz Band," *IEEE Std 802.11b-2005*, pp. 1 –721, 2005.

[74] The Network Simulator - ns-2, [Online]. Avaialbe: http://www.isi.edu/nsnam/ns/.

[75] J. Wu and I. Stojmenovic, "Ad Hoc Networks," *COMPUTER-IEEE COMPUTER SOCIETY*, vol. 37, no. 2, pp. 29–31, 2004.

[76] R. Punnoose, P. Nikitin, and D. Stancil, "Efficient Simulation of Ricean Fading within a Packet Simulator," in *Vehicular Technology Conference, 2000. IEEE VTS-Fall VTC 2000. 52nd*, vol. 2, pp. 764–767, IEEE, 2000.

[77] J. Kim, S. Kim, S. Choi, and D. Qiao, "Cara: Collision-aware rate adaptation for ieee 802.11 wlans," in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pp. 1 –11, april 2006.

[78] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, "Performance Anomaly of 802.11b," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, vol. 2, pp. 836–843, IEEE, 2003.

[79] M. Manshaei and T. Turletti, "Simulation-Based Performance Analysis of 802.11a Wireless LAN," in *Proceeding of International Symposium on Telecommunications. IRAN-Isfahan*, 2003.

[80] S. H. Wong, H. Yang, S. Lu, and V. Bharghave, "Robust Rate Adaptation for 802.11 Wireless Networks," in *ACM Mobicom*, pp. 146–157, 2006.

[81] E. Ancillotti, R. Bruno, and M. Conti, "Experimentation and Performance Evaluation of Rate Adaptation Algorithms in Wireless Mesh Networks," pp. 7–14, 2010.

[82] W. Yin, K. Bialkowski, J. Indulska, and P. Hu, "Evaluations of MadWifi MAC Layer Rate Control Mechanisms," in *Quality of Service (IWQoS), 2010 18th International Workshop on*, pp. 1 –9, june 2010.

[83] K. Bialkowski and M. Portmann, "Design of Testbed for Wireless Mesh Networks," in *Antennas and Propagation Society International Symposium (APSURSI), 2010 IEEE*, pp. 1–4, IEEE, 2010.

[84] W. Yin, P. Hu, J. Indulska, M. Portmann, and J. Guerin, "Robust MAC-Layer Rate Control Mechanism for 802.11 Wireless Networks," in *IEEE Conference on Local Computer Networks (LCN)*, p. 8, 2012.

[85] Atheros Chipsets, [Online]. Avaiable: http://madwifi-project.org/wiki/Chipsets.

[86] W. C. Jakes, *Microwave Mobile Communications*. 1974.

[87] OpenWrt Wireless Freedom, [Online]. Available: https://openwrt.org/.

[88] OpenWrt Backfile Readme, [Online]. Available: https://backfire.openwrt.org/10.03/README.

[89] iperf, [Online]. Available: https://iperf.sourceforge.net.

[90] The MadWifi Project - OpenHAL, [Online]. Available: https://madwifi-project.org/wiki/About/OpenHAL.

[91] Bash - The GNU Bourne-Again Shell, [Online]. Available:http://tiswww.case.edu/php/chet/bash/bashtop.html.

[92] Expect, [Online]. Available: http://www.nist.gov/el/msid/expect.cfm.

[93] Manual: Wireless FAQ, [Online]. Avaliable: http://wiki.MikroTik.com/wiki/Manual:Wireless_FAQ.

[94] Minstrel Rate Adaptation Algorithm Source Code, [Online]. Available: http://madwifi-project.org/browser/madwifi/trunk/ath_rate/minstrel/minstrel.c.

[95] D. Montgomery, *Introduction of Statistical Quality Control*. Hoboken, New Jersey: John Wiley & Sons, 2005.