An Improved 802.11b Interference Model for Network Simulation

Trevor Bosaw

A thesis submitted in partial fulfillment of
the requirements for the degree of

Master of Science

in

Electrical Engineering

University of Washington

2010

Program Authorized to Offer Degree: UW Department of Electrical Engineering

University of Washington
Graduate School

This is to certify that I have examined this copy of a master's thesis by

Trevor Bosaw

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

_____
Sumit Roy

_____
Tom Henderson

_____
etc

Date: _____

In presenting this thesis in partial fulfillment of the requirements for a masters degree at the University of Washington, I agree that the Library shall make its copies freely available for inspection. I further agree that extensive copying of this thesis is allowable only for scholarly purposes, consistent with fair use as prescribed in the U.S. Copyright Law. Any other reproduction for any purposes or by any means shall not be allowed without my written permission.

Signature_____

Date_____

University of Washington

Abstract

An Improved 802.11b Interference Model for Network Simulation

Trevor Bosaw

Chair of the Supervisory Committee:
Professor Sumit Roy
Electrical Engineering

Obtaining repeatable data from a controlled wireless experimental network set-up is both time-consuming and difficult; accordingly, network simulation is a popular mechanism for performance evaluation of such networks. Given the nature of wireless propagation, comprehensive models for the physical layer is an ongoing challenge. Many network simulators use analytical abstractions for the wireless channel that, while convenient, are too simplistic. For example, many simulators typically treat interference as additive white Gaussian noise to simplify calculations, which leads to significant inaccuracies in many scenarios.

The purpose of this thesis is to develop enhanced models for *in-network* interference in an 802.11b network for use within ns-3 implementation. As a first step, available analytical models for various 802.11b interference scenarios are gathered; these are extended by using Simulink for other scenarios. For experimental data, the Carnegie Mellon University wireless network emulator [1] was chosen, due to its unique ability to provide controlled and repeatable wireless channel conditions. Data from experiments using this emulator are collected and tabulated to support the analytical abstractions. Finally, the implementation of the 802.11b PHY/MAC stack within ns-3 is studied, and changes to its interference model is proposed as a result. The contributions of this thesis are to extend the analysis and data from these aspects to improve capture effect, preamble interference, and payload interference modeling for network simulators, specifically with ns-3.

TABLE OF CONTENTS

LIST OF FIGURES

ACRONYMS

AWGN: Additive White Gaussian Noise

BER: Bit Error Rate

CCA: Clear Channel Assessment

CCK: Complementary Code Keying

CRC: Cyclic Redundancy Check

DBPSK: Differential Binary Phase Shift Keying

DQPSK: Differential Quadrature Phase Shift Keying

DSSS: Direct-Sequence Spread Spectrum

MBPS: Megabits Per Second

PER: Packet Error Rate

SER: Symbol Error Rate

SFD: Start Frame Delimiter

SIR: Symbol to Interference Ratio

SNR: Symbol to Noise Ratio

WLAN: Wireless Local Area Network

Chapter 1

INTRODUCTION

1.1 Motivation

The success of 802.11 Wireless Local Area Networks (LANs) has led to its integration in increasing numbers of computing (desktops, laptops/notebooks) and personal mobile (smartphones and PDAs) devices. Increasingly, Internet access via WiFi networks is available at a variety of places including homes and offices, as well as in public places such as airports, hotels, malls and sporting arenas. The ubiquity of 802.11 WLAN networks implies growing number of users for a network design that was originally intended for home use (i.e. few users/devices per access point); such increasing user density leads to scenarios that are increasingly interference-limited. Such interference arises both from other overlapping 802.11 networks - homogeneous or in-network network interference - as well as heterogeneous (external, out-of-network) interference sources such as Bluetooth devices that may be co-located. Accurate performance analysis for such scenarios is an ongoing challenge that is not well-understood - wherein, the role of credible network simulators plays a key role.

Ideally, research in this topic should be complemented by credible experimentation with WLAN networks. As is well-known, reproducible experimentation with wireless networks is difficult, arising from the dynamic nature of the wireless channel. The interactions of the transmitted signal with the physical environment is complex and multi-layered (channel propagation loss, multipath fading, scattering etc.) and statistics specific to the scenario must be obtained via time-consuming data collection. Often, the greatest impediment to experimentation with WLAN is its own success; this implies the lack of any 'quiet space' (that would not interfere with an existing data network) in the 2.4 GHz band for controlled experimentation. This persistence of interference is perhaps the most difficult aspect of the wireless channel to predict or offset, and this ubiquity of the WLAN standard has allowed the problem of interference modeling to progress substantially.

Due to the complexity of the wireless channel interactions, software network simulators are commonly used for performance analysis of most wireless networking scenarios. As a result, the fidelity of the typical outputs of such simulators in terms of network layer metrics (aggregate throughput, delay, packet loss) are highly sensitive to the implementations for the lower (notably physical and MAC) layers within the simulator. Additionally, many network simulators measure packet behavior with a discrete event system, where the simulator does not calculate individual bits transmitted and received, but instead allocates events whenever something happens in the system. The network simulator ns-3 [3] uses a simulator library that schedules, orders, and executes events, such as the receipt or completed transmission of a packet. These events completely describe the behavior in the system, and serve to simplify the calculation required for the simulation (as opposed to calculating transmitted bits and performing arithmetic on the bit level). Due to this discrete event system approximation that most network simulators utilize, it is important that these simulators remain modular and simplify the process of changing a specific layer implementation. For example, the ns-3 architecture consists of various distinct libraries with specific functions, with the ability for users to write and link their own libraries. Most of the classes within these libraries are implemented as purely virtual classes, to allow for the easy addition of an alternate class implementation.

As can be readily understood, accurate representations or abstractions of interference within the PHY/MAC protocol stack implementations in the simulator is a fundamental step towards achieving desired network level fidelity. Typically, clear channel conditions (detecting a desired packet in the presence of additive background noise only) are adequately represented; however, interference scenarios are often implemented naively. For example, the network simulator ns-3, (which is generally regarded as having a good 802.11 PHY/MAC implementation) treats interference as Gaussian noise, which can often lead to inaccurate performance predictions [9].

1.2 Contribution

The objective of this thesis is to enhance the 802.11b interference model within ns-3. This is accomplished by both studying the situations in which the ns-3 noise interference model

is not adequate, and by supplying experimental data to serve as reference for appropriate receiver behavior in these cases.

The main contributions of this thesis are to submit changes for the following three areas in the ns-3 network simulator: the PHY layer receiver packet capture behavior, the error rate model for interference during a received packet's preamble, and the error rate model for interference during a packet's payload. These contributions are determined from a Simulink receiver model [4], interference data from the CMU Wireless Emulator [1], and interference experiment results from other sources.

The Simulink model, as well as communications analysis, is used to provide statistical results on a 1Mbps 802.11b signal with interference. This model is built upon an existing Simulink 802.11b transmitter/receiver chain that calculates the bit error rate by transmitting and receiving a bitstream with channel noise. The original model has no capacity to calculate packet error rate, and therefore cannot be directly used to correctly model packet reception behavior. To rectify this, the IEEE standard was consulted, and a four stage model was built for packet acquisition. For a packet to be marked as correctly acquired in the physical layer, all four stages must pass. These stages are: energy detection, SFD match, header CRC check, and payload CRC check. Each stage models the specific implementation a hardware receiver uses to determine whether a packet is accepted or dropped.

The CMU Wireless Emulator interference data gathered in this thesis includes the most basic interference case, the presence of a single interference source. Since there is little available research describing the effects of interference on packet reception for a sub-packet resolution, the experiment was documented thoroughly, and many sources of error were accounted for and minimized. Also, the clear channel response of the emulator was measured and matched against verified theoretical results, in order to provide further validation of the accuracy of the experiment results. The CMU Emulator interference model is integrated into ns-3 in the form of a parallel implementation to the current ns-3 wireless error rate and interference modeling, and provides a lookup table in the case of single source interference.

1.3 Overview

The thesis is organized as follows: In Chapter 2, relevant background research on interference in the wireless physical layer is described, and compared against current network simulators. This provides a clear purpose as to why changes need to be made in the interference models of these simulators. Chapter 3 details the methods followed for the CMU Emulator experiments, the Simulink model creation, and the ns-3 interference tests. In Chapter 4, the physical layer capture effect is explained, as well as the simple changes necessary for ns-3 to exhibit the behavior of this effect. In Chapter 5, the effects of interference during a packet's preamble is investigated, and additions to the ns-3 error rate model to account for these effects are presented. Chapter 6 discusses the effects of interference during a packet's payload, and provides experimental, as well as analytical results, to propose an improved ns-3 error rate model for payload interference. Finally, Chapter 7 concludes the thesis, with a discussion on potential future work for the subject.

Chapter 2

BACKGROUND

2.1 Introduction

Out of all the network layers, the physical layer is the most difficult to fully understand and model. The reason for this is that this layer is the only one not fully defined in software. For example, in a networked system, the physical layer cannot specify its channel between two connected nodes, but must instead predict it. This causes difficulty if the channel is not ideal. In wired systems, the prediction of the channel response is relatively simple. There is rarely interference, and noise rates are minimal. With the introduction of wireless technology, however, new problems have surfaced. The wireless channel is volatile, with unpredictable frequency shifts, fading characteristics, and other effects not present in a wired connection. In addition, interference has emerged as an increasingly influential problem. While many channel characteristics, such as gaussian noise, can be predicted due to the probabilistic nature of random variables that are based on these characteristics, interference cannot be modeled in the same way. Also, due to the unique behavior from every type of interference source, it is difficult to fully characterize and model a channel with interference.

In this chapter, clear channel analysis is initially presented, to show the mathematics behind the noise and interference implementation of many network simulators, including ns3. Next, current research attempts to understand interference are described, as well as the accuracy of these studies. Also, the current state of wireless network simulators is discussed, since it is these simulators that directly benefit from studies done on interference modeling. These sections show the necessity for a more thorough interference research attempt, as well as a reconsideration of current interference behavior implementations in wireless network simulators.

2.2 802.11b Clear Channel Model for 1 Mbps

An initial source of information on wireless interference is the IEEE commissioned study on 802.15 and 802.11b interference [6]. This study derives theoretical models of 802.11b BER and PER in the case of noise interference. According to testbed results from [9], however, the results from these models do not exactly match up with actual data. In the ns-3 clear channel validation document [12], the original theoretical 802.11b BER models are taken and expanded upon due to derivations from [13] and [14]. Following is a summary of the models derived in the document.

The simplest of BER derivations, the 1 Mbps BER, is extremely similar to the chip error rate for DBPSK. Taking into account that there is one bit per symbol, and 11 chips per bit, the BER for 1 Mbps is:

$$BER = 1/2e^{-E_b/N_0} \tag{2.1}$$

where E_b is the received energy per bit, and N_0 is the noise power spectral density.

Since each bit is independent, a simple binomial method can be used to find the PER from the BER:

$$PER = 1 - (1 - BER)^n \tag{2.2}$$

where n is the number of bits in the packet.

2.2.1 802.11b Clear Channel Model for 2 Mbps

In [13], it is shown that the chip error rate for 2 Mbps uses the Marcum Q-Function, which has a semi-infinite integration. Because of this, an approximation is used for the DQPSK chip error rate calculation. This result is directly used in the derivation of the BER for 2 Mbps, which is:

$$BER = \frac{\sqrt{2}+1}{\sqrt{8\pi\sqrt{2}}} \frac{1}{\sqrt{\frac{E_b}{N_0}}} e^{-(2-\sqrt{2})\frac{E_b}{N_0}} \tag{2.3}$$

again, where E_b is the received energy per bit, and N_0 is the noise power spectral density.

The PER calculation for 2 Mbps is the same as the one for 1 Mbps:

$$PER = 1 - (1 - BER)^n \tag{2.4}$$

where n is the number of bits in the packet.

2.2.2 802.11b Clear Channel Model for 5.5 Mbps

Since 5.5 Mbps and 11 Mbps 802.11 signals use CCK, there is no direct analytical method to derive BER equations for these bitrates. However, in [14] it shows that the analysis for biorthogonal-key modulation can exactly represent the PER in 5.5 Mbps CCK modulation. From [14], the symbol error rate for a 16-biorthogonal signal set (or 16-CCK, which is what is used in the case of 5.5 Mbps) is:

$$SER_{16} = 1 - \int_{-\sqrt{\frac{E_s}{N_0}}}^{\infty} [2\alpha(x + \sqrt{\frac{E_s}{N_0}}) - 1]^7 \frac{\exp(-x^2/2)}{\sqrt{2\pi}} dx \tag{2.5}$$

where α is the standard Gaussian distribution function, E_s is the energy per symbol, and N_0 is the noise power spectral density.

The PER calculation for 5.5 Mbps is similar to 1 Mbps and 2 Mbps:

$$PER = 1 - (1 - SER_{16})^s \tag{2.6}$$

where s is the number of symbols in the packet.

2.2.3 802.11b Clear Channel Model for 11 Mbps

The derivation of 11 Mbps error rates are again taken from [14], and the symbol error rate is approximated by the symbol error rate for a 256-biorthogonal signal set:

$$SER_{256} = 1 - (1 - SER_{16})^2 \tag{2.7}$$

where SER_{16} is the symbol error rate for 16-CCK above.

The PER for 11 Mbps is simply:

$$PER = 1 - (1 - SER_{256})^s \tag{2.8}$$

where s is the number of symbols in the packet.

All of the PER calculations are shown to be completely valid in clear channel conditions, when additive white gaussian noise is the only interference on the 802.11 signal. In order for the channel modeling of any wireless simulator to be considered accurate, these simulators must be validated against clear channel theoretical results such as these. In this thesis, any tool used to describe 802.11b interference is first compared against these models to determine the basic accuracy of the tool.

2.3 802.11b Interference Error Rate Experimental Behavior in Hardware Tests

Despite their accuracy in clear channel conditions, packet error rate models for AWGN channels do not necessarily extend to interference dominated scenarios. A reason for this is that an 802.11b packet includes a preamble, header, and payload, each that exhibit different behaviors at the receiver. Figure 2.1 shows a common 802.11b packet with a long preamble and header, as well as the packet payload. Receiver operations such as synchronization and channel estimation must occur during the packet preamble, and CRC checks must be passed at the header and payload for the packet to be accepted at the receiver.

The interference explorations in [15] discuss that the packet success rate probability is dominated by the ability of the receiver to successfully synchronize with the intended packet. In 802.11b, there are three ways a packet can be dropped at the physical layer: if the receiver does not properly synchronize to the receiver, if the header CRC fails, or if the payload CRC fails. [15] shows that for both 802.11b short and long preamble, the probability of a synchronization failure is extremely high, unless the signal to interference ratio is greater than 10 dB. Figure 2.2 shows the synchronization results from this paper.

These results compare to the paper's findings for the probability of CRC failure (as shown in Figure 2.3), which for 2 Mbps is less than .2 at a signal to interference ratio of -8 dB, and is negligible when signal to interference ratio is greater than 4 dB (these results

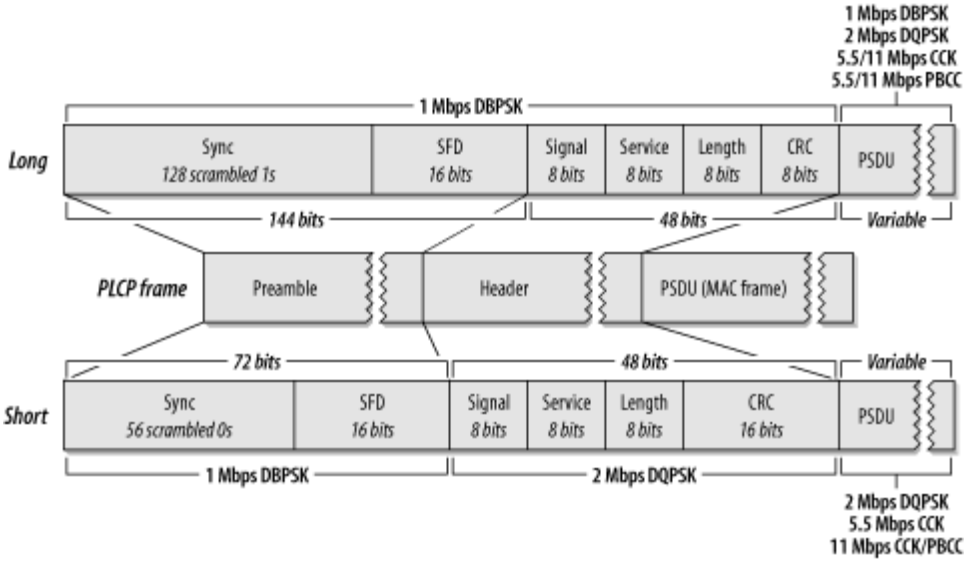


Figure 2.1: Frame format for both an 802.11b long preamble and short preamble packet. The PSDU section is referred to as the payload throughout this thesis.

are true for all payload packet sizes). Even at the highest 802.11b data rate of 11 Mbps, the probability of CRC failure is less than .03 at a signal to interference ratio of 5 dB.

The interference results from [9] provide a significantly different account of interference behavior than in [15]. Figure 2.4 shows the results of an interference test of two nodes transmitting under hidden node conditions (the two transmitters are out of range from each other, but both are in range of a single receiver). The x-axis represents the amount of time elapsed after the intended packet has been transmitted, and before interference is added to the channel. The y-axis represents the received signal strength of the intended packet (the interference packet is always transmitted with dBm equal to -82, so the horizontal line at -82 dBm is equivalent to 0 dB SIR). The colors indicate the amount of correctly received packets at each delay and received signal strength value, and range from 0 packets (for a packet success rate of 0), to 200 packets (for a packet success rate of 1). These tests have been completed for all four 802.11b bit-rates.

The results from Figure 2.4 agree with [15] that the contribution to packet error rate from the preamble synchronization dominates that of the CRC failure for low bit-rates. In

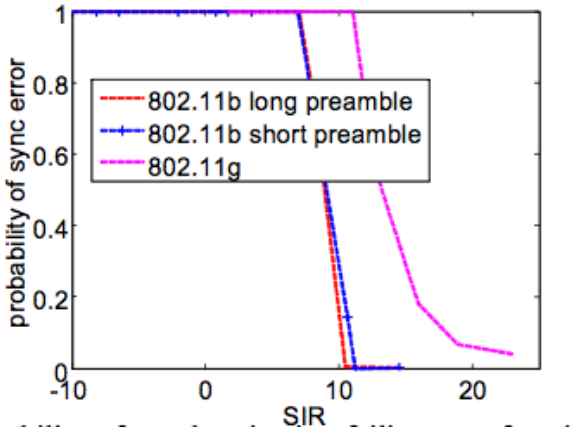


Figure 2.2: Probability of synchronization as a function of SIR (Courtesy of [15]).

the graphs for 1 Mbps and 2 Mbps, it shows that it is critical the intended packet is allowed a specific amount of time for packet synchronization (in the case of these graphs, this time is approximately 37 microseconds). After this initial time period, the reception behavior is improved by about 4 dB. The two studies diverge, however, on the specific SIR necessary for a successful synchronization and packet reception. Whereas [15] states that a packet needs at least 10 dB SIR to successfully synchronize to a packet, [9] shows that this is only true when the interference begins at the start of the packet, and that if interference is delayed by even 3 microseconds, the packet can successfully synchronize in levels as low as -2 dB SIR.

For higher bit-rates of 5.5 Mbps and 11 Mbps, the results in Figure 2.4 are quite different than those in [15]. While the results agree that the very first initial synchronization (about 3 microseconds in these graphs) is extremely important in packet reception, after this initial period it appears that CRC payload/header failure dominates reception behavior. In these graphs, packets can still be successfully received at SIR values of 1 dB or higher (3 dB or higher for 11 Mbps), again disagreeing with the result in [15] stating that packet synchronization requires at least 10 dB SIR in order to be successful.

The differences in these papers suggest that a much more thorough interference research is necessary before either source can claim to have compelling interference behavior data.

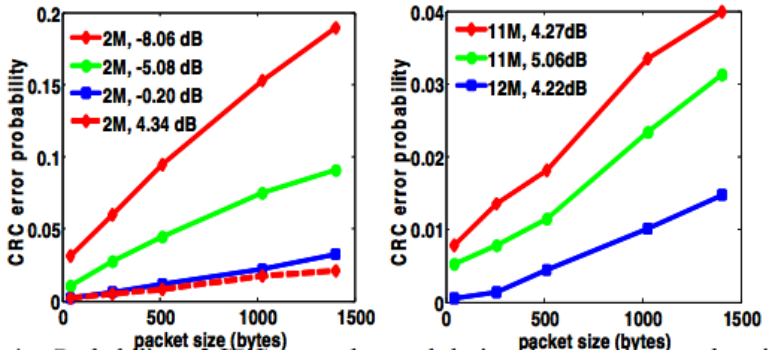


Figure 2.3: Probability of CRC error as a function of packet size and for various SIR values (Courtesy of [15]). The left hand graph represents curves for 2Mbps, and the right hand graph represents curves for 11Mbps and 12Mbps (the 12Mbps bit-rate is not discussed in this thesis).

It is difficult to determine whether the deviations in these papers result from experimental error, or from simply using different hardware. Since preamble detection and synchronization is largely implementation based (there is little in the IEEE standards stating how this should be done), it is likely that these functions vary greatly between wireless cards, and this serves as explanation for the differences in results. Also, experiments in [8] show how certain aspects of an 802.11b packet can be exploited with specific types of interference. This shows that even small differences in test interference can result in varying results. All of these small sources of error can contribute to a large error offset, and may result in the differences seen in these two papers on interference behavior.

2.4 802.11b Interference Error Rate Analytical Behavior

In additional to experimental research on the subject of wireless interference, there is also established mathematical analysis that attempts to describe interference behaviors. In this section, an analysis on the bit error rate and packet error rate response of a system with an 802.11b 1Mbps signal interfered by another 802.11b 1Mbps signal is performed. To simplify the analysis, it is assumed that the receiver is perfectly synchronized to the transmitted signal, and that no pulse shaping filter is used. In this system, there are three components: a 1Mbps intended signal, a 1Mbps interference signal, and AWGN noise. The

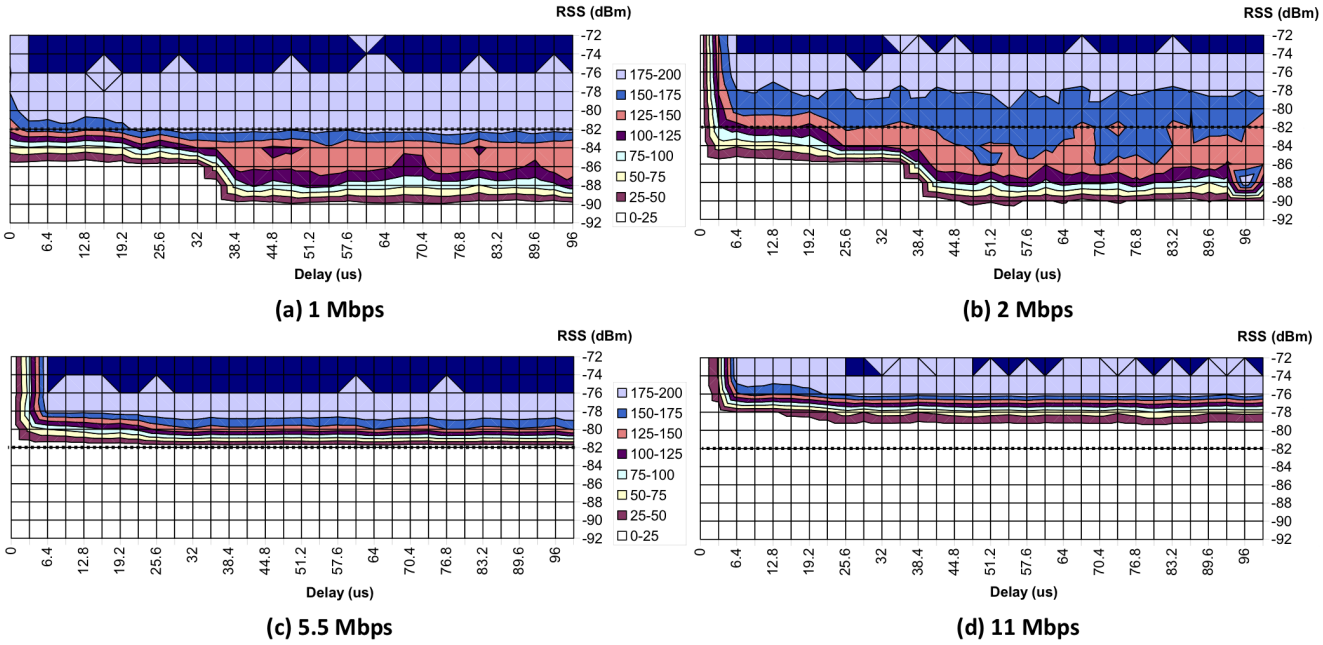


Figure 2.4: Packet capture versus interference delay (Courtesty of [9]). For the 1Mbps and 2Mbps bit-rates, interference negatively impacts the reception probability to a greater degree when it starts during the preamble.

interference signal is delayed by a time phase offset (τ), and the amplitude of the intended and interference signal are $\sqrt{E_s}$ and $\sqrt{E_i}$, respectively. This results in the following baseband equivalent signal at the receiver:

$$r(t) = \sqrt{E_s}\, A\, p(t) + \sqrt{E_i}\, I\, p(t-\tau)\, cos(\phi) + n(t) \tag{2.9}$$

where A is the intended bit (with either a +1, or -1 value), I is the interference bit (again, either +1 or -1), ϕ is the phase offset between the intended and interference signals, $n(t)$ is the AWGN noise component in the channel, and $p(t)$ is the 11 chip Barker sequence that acts as the transmit pulse. Additionally, t is measured in microseconds, and $p(t)$ is defined to be nonzero only when $0 \leq t < 1$.

The receiver is a filter that is matched against the Barker sequence transmit pulse $p(t)$. This filter correlates the input signal $r(t)$ against $p(t)$, over one symbol duration. The following equation describes the match filter:

$$y(T) = \int_0^T r(t)p(t)dt \tag{2.10}$$

where T is the symbol length (one full 11 chip Barker sequence).

The match filter output $y(T)$ then consists of three components:

$$y(T) = s(T) + i(T) + n(T) \tag{2.11}$$

where the s(T) component represents the intended signal, the i(T) component represents the interference signal, and the n(T) component represents the channel noise.

With the assumption that the receiver is perfectly synchronized to the intended transmitter, the $s(T)$ component will contribute a large correlation result. This result is dependent on A in $r(t)$, and is a large positive result (if A is positive), or a large negative result (if A is negative). The $i(T)$ component is largely based on the τ value in $r(t)$, as the Barker sequence is very robust to delayed correlation, and even a small shift in the interference component of $r(t)$ will greatly reduce the contribution to $y(T)$ from $i(T)$. Finally, $n(T)$ is the noise component, and is defined by the E_b/N_0 (energy per bit over noise power spectral density ratio) in the system. The goal at the receiver is to correctly resolve the intended transmitter's data, so a decision rule to predict the intended bit A is simply:

$$\left\{ \begin{array}{l} y(T) > 0, A = 1 \\ y(T) < 0, A = -1 \end{array} \right\} \tag{2.12}$$

Depending on the SNR and SIR (E_s versus N_0 and E_s versus E_i), the $i(T)$ or $n(T)$ components may dominate the $s(T)$ term, and may cause an incorrect decision for A. This probability can be averaged over a large number of test bits, and can be used to determine a bit error rate, or a packet error rate. To modify the analysis to calculate packet error rate, two methods can be used. Both methods require an updated Equation 2.9, since the equation currently does not take into account the fact that each intended bit is interfered by two interference bits (when $\tau \neq 0$). Thus, the following equation is used in place of Equation 2.9:

$$r(t) = \sqrt{(E_s)}\, A\, p(t) + \sqrt{(E_i)}\, (I_x\, p(t-\tau) + I_{x-1}\, p(1+t-\tau))\, cos(\phi) + n(t) \tag{2.13}$$

where I_x represents the current interference bit, and I_{x-1} represents the previous interference bit. Also, recall that $p(t) = 0$ when $t < 0$ or $t \geq 1$.

The same correlation and decision is still performed as in Equations 2.10 and 2.12. To calculate a bit error rate using this method, a large sequence of intended bits must be generated, and for each bit, the following equation used to determine the probability that the bit will be resolved correctly:

$$\begin{aligned} P(c) = {} & 1/4 * P(c|I_x = +1, I_{x-1} = +1) + 1/4 * P(c|I_x = +1, I_{x-1} = -1) \\ & + 1/4 * P(c|I_x = -1, I_{x-1} = +1) + 1/4 * P(c|I_x = -1, I_{x-1} = -1) \end{aligned} \tag{2.14}$$

where $P(c)$ is the probability of correctly resolving the specific bit, and each right hand term in the equation represents the four interference cases for that specific bit.

With the assumption that each intended bit is independent, a bit error rate for the bit sequence can then be found by averaging $P(c)$ over a very large number of bits. This bit error rate can then be used to calculate a packet error rate from Equation 2.8.

The second method does not assume that each intended bit is independent, and instead calculates $P(c)$ for each specific intended bit. A bit sequence is used to represent the bits in a packet, and the following equation is used to calculate a packet error rate:

$$PER = 1 - P_1(c) * P_2(c) * P_3(c)...P_{x-1}(c) * P_x(c) \tag{2.15}$$

where PER is the packet error rate, and the subscript for each $P(c)$ represents the probability of resolving the corresponding intended bit. It is assumed that there are x intended bits in the bit sequence.

Figure 2.5 shows an example of the differences between the two methods, as a function of τ. This comparison shows that the two methods are extremely similar, and differences between the two are only noticeable during the specific τ offset when packet error rate

transitions from low to high, or high to low. For the purposes of this thesis, the method using Equation 2.15 is used to calculate an analytical packet error rate for 1Mbps.

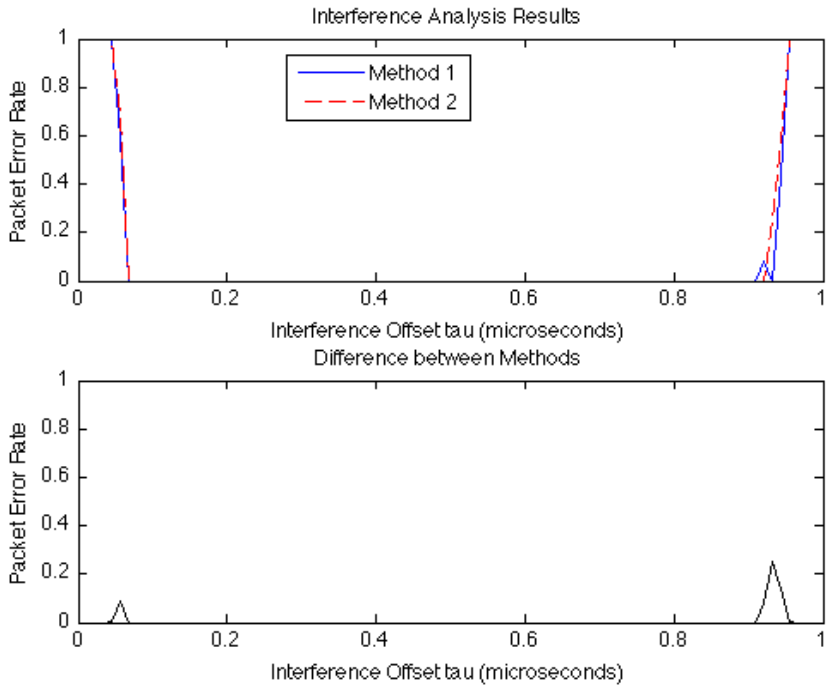


Figure 2.5: Comparison graph between packet error rate calculation methods using Equation 2.8 (referred to as Method 1 in the graph), or using Equation 2.15 (referred to as Method 2 in the graph). For this case, SNR = 0 dB, and SIR = 0 dB.

2.5 802.11b Capture Effect

The behavior of receivers known as the physical layer capture effect is another interference related research topic mentioned in many papers. As described in [10], the capture effect is the ability of the receiver to successfully receive a stronger packet in a collision, even if the receiver was already synchronized to the weaker packet. This study mentions three possibilities in the collision of two packets: both packets being dropped, the stronger packet being received correctly only if it has a sufficiently higher signal strength and is the first packet to arrive, and the stronger packet being received correctly if it has a sufficiently higher signal strength, regardless of whether it arrives first or last. The paper proceeds to

prove that the interference behavior on receivers matches the third option. While modeling packet reception in this manner greaty simplifies interference behavior, the finding that a receiver can switch reception of a packet to a stronger one, even after it is synchronized to the first packet, is very important. These results serve to potentially increase the throughput of network simulator interference modeling by 50 percent.

2.6 Current State of 802.11b Network Simulators

Perhaps due to the lack of published research on 802.11 interference effects, wireless network simulators largely simplify the calculation necessary for proper interference modeling. At least one network simulator, OPNET, even simplifies the problem to such a degree that any collision between two packets results in both packets being dropped by the receiver [11]. The popular network simulator ns-2 takes the most common approach by creating a threshold based interference model [2]. If the signal to interference ratio falls below a threshold (meaning interference energy is high) during the receipt of a packet, then the packet is dropped, regardless of the length or type of interference. The specific values of these thresholds may be due to derivation from research averaging the packet success rate for a certain signal to interference ratio, but this is for a very specific scenario, which can result in loss of accuracy if the interference occurs in merely a slightly different way than ns-2 predicts. For example, 802.11 DSSS modulation is robust against narrowband interference, but if narrowband interference with a very high signal strength is mixed with the intended signal, then ns-2 would calculate the result as a dropped packet, whereas the packet is not likely to be dropped in reality. Or, if another 802.11 signal interfered with the intended signal, then the original packet may be dropped even at low signal to interference ratios, while ns-2 would accept the packet.

The ns-3 network simulator attempts to fix this problem by taking a bit error rate approach to interference modeling [3]. In ns-3, the intended signal is split into chunks according to changes in packet bit-rate or interference energy. A bit error rate is calculated for each chunk, which is then used to calculate overall error rate for each chunk. Finally, these error rates are used to calculate the total packet error rate. The problem with this implementation is that these bit error rates are calculated using theoretical equations that

are validated against clear channel conditions, not interference. As shown in papers such as [9], interference does not cause the same behavior as noise, and when treated as such, does not create an accurate model.

Most network simulators follow either the ns-2 threshold based or ns-3 bit error rate based interference modeling implementation. Since many of these simulators can be validated over a clear channel, these interference models appear to operate suitably when matched to receiver response data. However, since there is little definitive interference data to corroborate these simulators against, so they cannot currently be validated in the case of interference.

Additionally, many simulators do not take into account the physical layer capture effect, as discussed in [10]. Instead of retaining the possibility of dropping a packet that the receiver is currently synchronized to, and picking up another incoming packet, most network simulators simply drop both packets (if the incoming packet's energy is high enough), or retain only the original packet. Unlike other interference behavior however, adding this feature is simple and easily integrated into these simulators, so the capture effect addition to ns-3 described in this thesis is easily applied to other network simulators.

2.7 Summary

To date, the research on WiFi interference does not comprehensively cover all scenarios and implementation issues, and there remain questions about what and how to model these issues in network simulators. Many resources do not corroborate in certain areas, while they agree in others. Because of this, a thorough study into exactly what creates certain characteristics of interference behavior is necessary. Each paper on interference behavior reveals a different aspect of interference modeling, and only by combining information from these multiple resources can interference behavior be fully understood and modeled.

Due to these discrepancies, network simulators have not currently been successful in creating accurate interference models. ns-3 has one of the most sophisticated noise models, with theoretical calculations validated in clear channel. However, this model does not match up to measured values for interference, and these differences must be reconciled.

Chapter 3

CMU EMULATOR, NS-3, SIMULINK MODEL DEVELOPMENT

3.1 Introduction

In order to investigate the effects of interference on an 802.11b receiver, three tools were used: the ns-3 Network Simulator, the CMU Wireless Emulator, and a Simulink interference model. This chapter introduces these three tools, and describes the method in using each of them to set up interference experiments, as well as any validation necessary to prove the accuracy of each implementation. Whenever interference tests are introduced for any test, the reference to intended packet describes the received packet from a chosen reference transmitter source. A reference to interference packet describes the received packet from an undesirable transmitter source. Signal to interference ratio defines the difference in signal strengths between the received packets from these two sources.

3.2 ns-3 Wireless Simulator

Despite the rapidly increasing presence of interference in the 802.11b transmission frequencies, interference modeling in network simulators remains limited. Many simulators model interference from a simple threshold, while others, such as ns-3, rely on a clear channel validated model for interference. In this section, the prodecure to measure three main areas of interference on packets in ns-3 is explained. These behaviors are: the physical layer capture effect, single packet preamble interference, and single packet payload interference. These three aspects consist of the majority of interference cases, and it is important to determine the characteristics of ns-3 under these three types of interference, before improvements can be made on the simulator.

For all ns-3 tests, the most current repository was compiled and used (as of April, 2010), which is available from [3]. The script example 'wifi-simple-interference.cc', available in the main repository, was used for all interference tests. This script sets up a three node

system, with one receiver, one intended transmitter, and one interference transmitter, where the intended and interference transmitter send packets to the receiver, with the intention of causing collisions and measuring interference effects. From this script, the interference delay, packet size, intended transmitter signal strength, and interference transmitter signal strength can be controlled through input commands. For the purposes of the interferences tests in this thesis, a slight modification was made to the script to allow for the transmission of multiple intended and interference packets. All mentions of an interference script refer to this example script.

3.2.1 Physical Layer Capture Effect

The ns-3 physical layer is composed of a state machine, split into four states: TX, RX, IDLE, and CCA_BUSY. Each node initializes in the IDLE stage, and if the power in the channel is measured to be above a clear channel assessment (CCA) busy threshold, the state can switch to CCA_BUSY. In both of these stages, any packet above a packet receipt threshold can be received at any time. When the node begins to transmit a packet, the state is automatically changed to the TX state, regardless of the current state of the node (it is the medium access control (MAC) layer's responsibility to prevent a transmission while the node is in the RX state, so the physical layer does nothing to avoid this). While the node is in the TX state, no packets can be received. Finally, the node can switch to the RX state from the IDLE or CCA_BUSY state whenever a packet is received (again, assuming the packet's power is above the packet receipt threshold). When in the RX state, the node cannot receive any further packets.

To test this behavior, the ns-3 example interference script was used, and settings were calibrated so that the intended packet signal strength was greater than the interference packet signal strength, the interference delay was negative (so the interference packet was received before the intended packet), and the number of packets transmitted was set to 1000. With this script, it was found that the ns-3 physical layer did behave as expected, and that there were no circumstances where an intended packet can be received when the physical layer is already synchronized to another packet. As detailed further in Chapter 4,

this behavior is not accurate, and changes are necessary in this area.

3.2.2 Preamble Interference Model

The interference modeling implementation in ns-3 currently has no functionality for the calculation of the effects of interference during the preamble of a received packet. Any interference that occurs during a packet preamble is not treated as interference at all. It is obvious that this behavior is not correct, and Chapter 5 details the recommended changes for this aspect of interference in ns-3.

3.2.3 Payload Interference Model

In the current ns-3 implementation, payload interference is determined by calculating a specific bit error rate for a specific region, and then applying this bit error rate as a binomial distribution on the amount of bits in the region (as described in Chapter 2). To do this, it first constructs an event array of the power changes in the channel over the time period that the intended packet is received. This array is able to keep track of changes of interference for multiple packets, as well as noise or interference existing before the receipt of the intended packet. In order to actually calculate the packet error rate, the interference helper first splits the time segment that the packet is received into chunks. Each chunk represents a section of the received packet (either the header or payload) with a specific interference power level. Each chunk can only have a single interference power level, and so a received packet with many changes in interference will have many chunks, while a packet with a constant level of interference over its time of receipt will only be split into two chunks (header and payload). Figure 3.1 shows an example of the chunk distribution for a typical scenario where a packet is received with interference.

After splitting the received packet into chunks, chunk success rates are found for each section. To do this, the bit rate of the packet, interference power, and time of chunk are used. The bit rate of the packet is used to decide what base error rate function is used (bit error rate is different between 1 Mbps and 5.5 Mbps, for example, because the latter uses CCK coding), the interference power is used to calculate the signal to noise ratio of the

chunk, and the time segment length is used (in conjunction with the bit rate) to determine how many bits are in the chunk. Finally, to get chunk success rate, the bit error rate is used in a summing binomial function which is then raised to the power of the number of bits in the chunk. Taking all of these chunk success rates and multiplying them together then results in the packet error rate, which is sent up to the physical layer to determine whether the packet should be dropped or not.

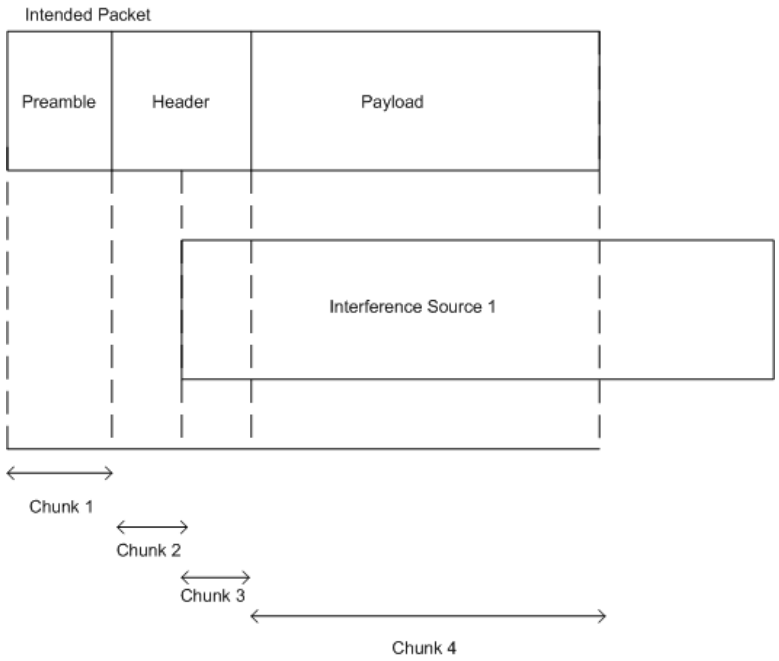


Figure 3.1: Visual description of where ns-3 splits a packet into chunks for a typical interference scenario

The majority of the ns-3 interference modeling occurs in the calculation of payload interference. This model takes into account multiple packet interference, as well as different methods for every common 802.11b bit-rate, but it does not corroborate with actual receiver behavior. To test the ns-3 payload interference behavior, the ns-3 example interference script was used. The intended packet signal strength was set to -82 dBm, so that the effects of noise in the system would be reduced. The interference packet signal strength was ranged from -90 dBm to -76 dBm, and the time offset of the interfering signal was ranged from 0 (the interference packet interferes with the intended packet completely) to the length of

the intended packet (any time offset greater than this, and there would be no interference occurring on the intended packet). The packet size was set to 1500 bytes, and 2000 packets were transmitted for each test.

The results of this test show that the ns-3 interference modeling simply treats interference as noise, which, as shown in Chapter 6, is only an accurate representation of payload interference under specific conditions. For all other conditions, a different payload interference method must be derived.

3.3 CMU Wireless Emulator

The CMU wireless emulator [1] is an FPGA based wireless network emulator that allows for the emulation of wireless signals and channels between a series of nodes. Each node is a wireless device that is physically connected to a central FPGA, which then emulates an actual wireless channel between nodes. These nodes each have an Atheros card with Madwifi drivers, and can be set up with completely separate channels between nodes, allowing for, among many other functions, testing of the most common interference scenarios (such as hidden terminal).

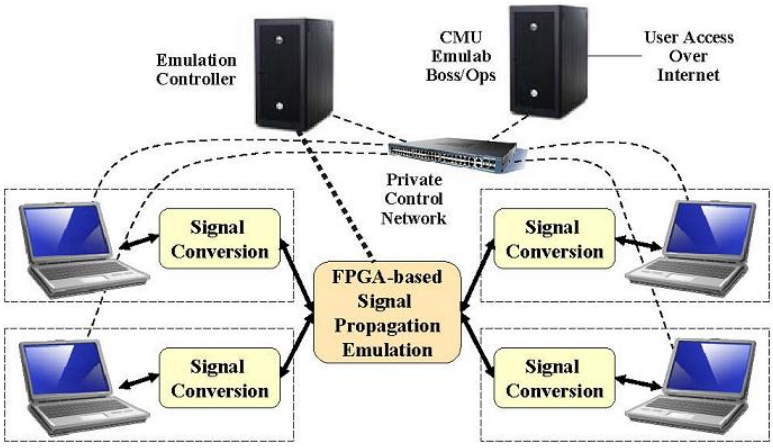


Figure 3.2: Diagram of the CMU wireless emulator

Currently, there is very little work done investigating the effects of interference on packet reception with a sub-packet resolution. The main reason for this is the difficulty of setting

up a consistent and reliable channel between nodes. The CMU emulator does not experience this difficulty, however, as the channel can be explicitly set up and altered as necessary. In the emulator, each node pair can be set up with a specific and independent channel from every other node pair. With this, full channel control is made simple.

Using this tool, a clear channel validation is performed in order to prove the accuracy of the emulator, followed by single source interference experiments. These experiments are used to investigate the nature of 802.11 interference, as well as serving as an interference model for network simulators in the form of a simple lookup table. The results from these experiments are detailed in Chapter 6.

3.3.1 Clear Channel Experiment and Validation

While there have already been clear channel 802.11b results published for the CMU emulator, many changes have occurred in the system since these results. Because of this, the current system setup must be validated for clear channel conditions before any interference tests can be carried out. A simple two node setup was used to measure packet success rate as a function of received signal strength (RSS). To perform the experiment, one node was set up as a transmitter, and another as a receiver. The path-loss between these nodes was set to a certain value in the CMU emulator settings, and 2000 packets were sent from the transmitter to the receiver. The number of successfully received packets over the total number of sent packets were then saved as the packet success rate at that path-loss (path-loss is converted to RSS, in order to match established packet success rate results). This was repeated for various path-loss values, and for all 802.11b bit-rates (1 Mbps, 2 Mbps, 5.5 Mbps, and 11 Mbps).

The findings from the Clear Channel Experiment closely match with the verified CMU emulator Clear Channel data reported in [12], effectively validating the system for clear channel. With these results, interference experiments can be completed with the knowledge that at least the foundation of the CMU emulator noise response matches actual hardware results.

3.3.2 Single Source Interference Experiment

It is intuitive to assume that a simple three node experiment would be sufficient to properly test the effects of same channel interference on packet reception. To do this, two nodes would transmit at the same bit rate and channel, and a third node would receive packets from the other nodes. The transmitting nodes would be synchronized so that one node would transmit at a given time before the transmission of the second node, allowing an interference delay offset to be measured. Packet error rate would then be determined as a function of this interference delay offset. In reality, however, relatively severe timing constraints exist, due to the operating system present on the nodes. Because of these constraints, any packet ordered to send on a node is not necessarily transmitted immediately, but within a certain time window, on the order of milliseconds. This is a difficulty in packet interference, because an entire packet can be received in as little as a fraction of a millisecond. This initial experiment must be corrected to take these timing constraints into account.

To remedy this difficulty, a five node experiment was constructed. In this experiment, the nodes are referred to as: the transmitter, the interferer, the receiver, and the two sinks. The receiver and the first sink can receive packets from the transmitter, while the receiver and the second sink can receive packets from the interferer. No other node propagation combination was allowed (these channels implement an infinite propagation loss on the CMU emulator). The propagation loss between the transmitter and the first sink, as well as the interferer and the second sink, was kept at a minimum, so that these sinks will pick up every single packet sent from the corresponding node. The propagation loss from the transmitter to the receiver, and from the interferer to the receiver, was kept as two variables, in order to test the effect of signal to interference ratio and packet loss. Figure 3.3 shows the node topography.

To solve the previously mentioned timing problem, propagation delays between the nodes were set to zero. This allows the sinks to receive packets from the corresponding nodes at the exact same time the receiver node receives the packets (the channel is much more robust in terms of timing, due to the reliance on an FPGA for its model). The first sink can then be used to see the exact time that each packet from the transmitter was received, the

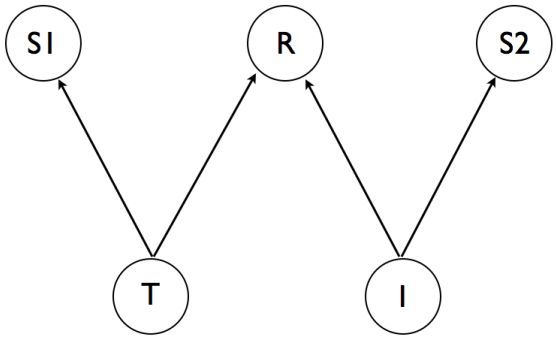


Figure 3.3: CMU Emulator Interference Experiment Node Topography

second sink can see the exact time that each packet from the interferer was received, and the receiver node can see whether the packet was resolved or dropped. Next, the transmitter and the interferer node were set to send packets at slightly varying time intervals. It is not possible to directly set the location in the transmitted packet where interference starts with this setup, but it can be seen exactly how the transmitted and interference packet match each other, and with enough runs of the experiment over different transmit intervals for both the transmitter and interferer, data can be gained for all offsets of the interfering packet versus the transmitting packet. Figure 3.4 shows how the transmitter with a certain transmit interval, and the interferer with a slightly different transmit interval, can produce data with different offsets of the interfering and transmitting packet.

These interference tests were completed with 802.11b, and all four bit rates were tested (1 Mbps, 2 Mbps, 5.5 Mbps, 11 Mbps). Packet size was set to 1500 bytes, and packets were sent via broadcast, to prevent retry attempts. Also, all the nodes were connected to a central, 'local access network' node, which kept the node timings synchronized through the use of periodically sent broadcast packets. In order to determine the synchronization accuracy of this setup, the timestamps from the receiver node packets were compared to

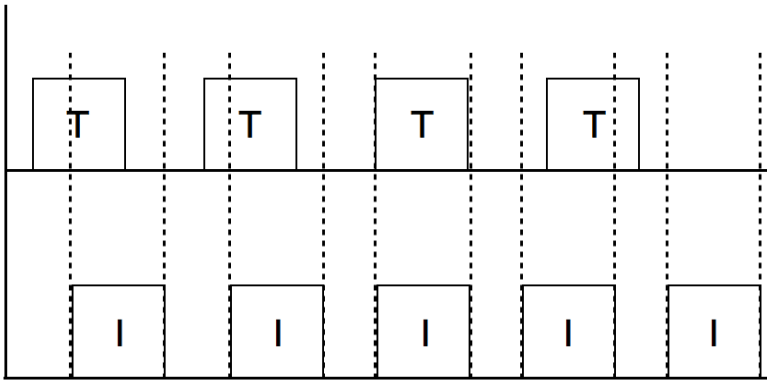


Figure 3.4: Packet overlaps with transmitter and interference source sending packets at slightly different time intervals

the timestamps from the corresponding packets from either sink node. It was found that in nearly all cases, the receiver node packet timestamps remained within 20 microseconds from the corresponding packet timestamps from either sink one or sink two. This suggests a timing accuracy suitable for payload interference tests, but not necessarily accurate enough for preamble interference tests. Because of this, the scope of the experiment was limited to payload interference.

3.3.3 Emulator accuracy and validation

Since there is little existing research investigating the effects of interference on packet error rate for a sub-packet resolution, any measured results from interference tests cannot be easily correlated with existing results. Instead, results must be validated with controlled measurements of different aspects of the experiment, and sources of error must be minimized. The different aspects of the experiment that may contribute to inaccuracy in the data include: timestamp offset, received signal strength, clear channel results, and specific node characteristics. In order to ensure that the packet received at each sink and the receiver are the same, a unique transmission number was added to each broadcast packet from the transmitter and the interferer. Also, the method to receive packet information was to use the 'tcpdump' program set on a monitor port at both sinks and on the receiver.

Packet timestamp offset

To determine exactly the packet activity of the system, the timestamps corresponding to each received packet were referred to. These timestamps were synchronized among all the nodes through the use of small beacon packets periodically distributed through the local access network. Since the experiment involves sending large packets at a rapid rate, there are no assurances as to whether the monitor nodes are able to receive these beacons and stay synchronized. It is important that this synchronization is maintained, because a large inconsistency between the timestamps of a packet received at a sink node and the receiver would prevent an accurate reconstruction of the channel activity. To test for node synchronization, the timestamp for each packet resolved by the receiver during a normal experiment was compared to the timestamp for the corresponding packet from the sink node (either the first or second sink, depending on whether the receiver packet was from the intended source, or interference source, respectively). The difference between these two timestamps was saved for each packet pair, and statistics were established for this data over all of the experiments run. It was found that that the variance in these timestamp differences was no greater than 20 microseconds for all experiments, and that outliers (timestamp differences greater than 50 microseconds) were extremely rare. Figure 3.5 shows an example stem plot of the timestamp offsets for all packets during a single experiment.

The variance range suggests that the timestamp offset between the receiver and sinks is consistent, and is accurate on the order of approximately 20 microseconds. Thus, in order to provide compelling channel activity reconstruction, the granularity of the measurements must be greater than 20 to 30 microseconds. This means that the results cannot accurately be used to determine the effects of interference in the packet preamble and header (which are received in approximately 96 microseconds), but can be used to determine the effects of interference in the payload, which is received on the order of milliseconds. These measurements show that the receiver remains adequately synchronized to the sink nodes throughout the experiment, and that the packet timestamps are consistent among nodes, suggesting that packet receipt times are being adequately determined.

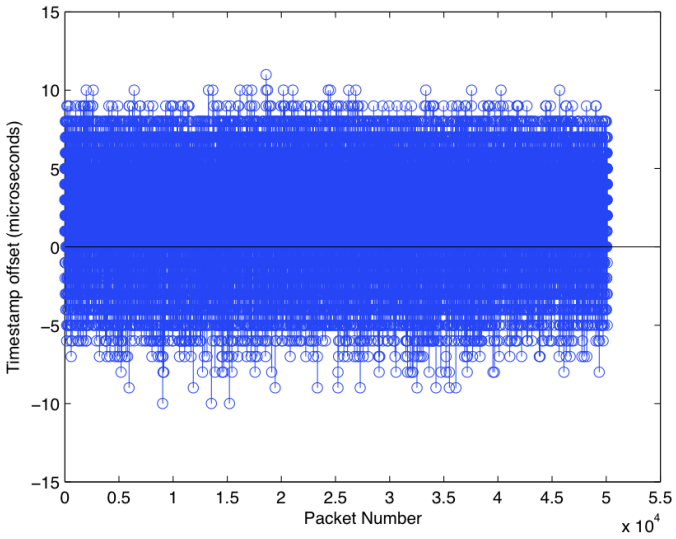


Figure 3.5: Timestamp offsets for each packet. An offset is defined as the difference between the timestamp and the receiver and the timestamp at the corresponding sink, for a specific packet

Received Signal Strength (RSS)

The RSS registers on the CMU emulator nodes are not calibrated, so these values are only consistent to a resolution of about one to two dB. A more accurate method to calculate the RSS for each packet in the CMU emulator is to take the transmission power of the packet and subtract the set pathloss of the channel. Since results are being measured on the order of a single dB, the RSS dB values must be extremely accurate. The same clear channel test detailed above was performed for each sink node, ensuring that this method of finding RSS for each packet was accurate. These clear channel experiments showed that the method of taking the transmission power and subtracting the pathloss to get the proper RSS value is in fact an accurate method, and is not an important source of error in this experiment.

Non-interfering Channel Results

In order to ensure that both sink nodes are working properly throughout the experiment, a count of all the packets received was made at both sinks (since there is no interference at

these sinks, these are considered clear channel results). This count was then matched to the total transmitted packet counts at the intended and interference source. A simple packet error rate was then determined by the division of the received packet counts at the sinks by the transmitted packet counts at the intended/interference source. This packet error rate was found to be less than .001 in all cases, suggesting that each sink node did not fail and worked as expected for all experiments.

Clear channel results could not be run on the receiver node concurrently with the interference tests, so a clear channel test was performed on the receiver node before and after each interference test (where 2000 packets were sent from the transmitter to the receiver, and the packet success rate was determined from this result). This process resulted in a packet error rate of less than .001 for all experiments, which ensured that the clear channel response for the receiver remained appropriate throughout the experiment process. Also, timestamp offset results remained within acceptable values for all measured experiments, which suggest the receiver performed as expected for the experiments.

3.3.4 Result Presentation

The CMU Emulator interference experiment results presented throughout the thesis commonly are arranged in the form shown in Figure 3.6. This figure shows an example graph describing the regions of interest for the experiment results. This data was created by measuring collisions in all of the situations where the first sink node measured a packet's time stamp that was within a certain range of a packet's time stamp measured by the second sink node. These situations were added up to form a total collisions array. Next, the receiver node was referred to, and for each collision, it was determined if the receiver node was able to resolve a packet from the desired transmitter, or if the packet was dropped. The number of packets resolved in favor of the desired transmitter, divided by the total number of collisions in the region of interest, formed the packet success ratio as seen in these graphs. Each stem represents the percentage of collisions resolved at each delay value, where delay is defined as the time interference packet is received minus the time transmitter packet is received (For example, a stem at delay = 2 milliseconds with the value .78 means that 78

percent of collisions were resolved in favor of the desired packet, when the desired packet arrived at time x and the interference packet arrived at time x+2ms). In order to reduce the visual noisiness of the data, the values were binned into delay time groups equal to approximately 2 percent of the packet length for each bit-rate.

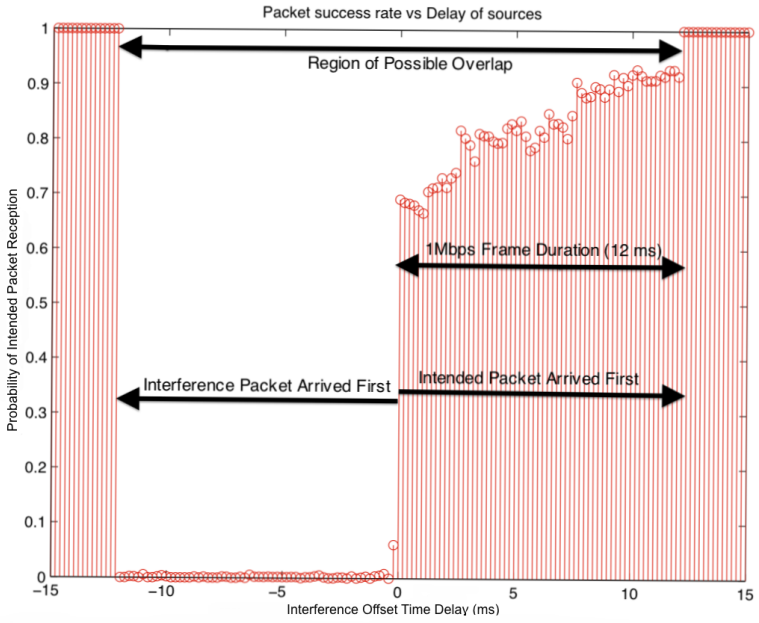


Figure 3.6: Example layout for CMU emulator experiment results. The figure represents packet success rate as a function of time delay, where time delay is defined as the absolute time that the interference packet arrived, minus the absolute time that the intended packet arrived.

3.4 Simulink WLAN Interference Model

While the CMU emulator data provides an important resource for simulating interference in network models, it does not provide for a more fundamental understanding of the basic causes of interference effects. A Simulink interference model is then used to more fully understand 802.11b interference. Since there are many aspects in a hardware receiver that are difficult to accurately model in software, this model is not used as a standard to judge the exact behaviors of interference. Instead, the Simulink interference model is used to provide insight into the different aspects of interference, and how important each aspect is.

This model is built to match the IEEE 802.11b standard as closely as possible, in order to reduce any source of error within experiment results. With the use of this software model, interference characteristics can be investigated at the bit level, the chip level, or even at the sub-chip level, which is difficult to perform with hardware testing.

3.4.1 802.11b Receiver Technical Background

A normal 802.11b receiver is implemented at the physical layer as a state machine with four main sections. These sections are the Sync, the SFD, the header, and the payload sections. For a packet to be accepted by the physical layer, it must successfully pass all four sections. Figure 3.7 shows the entire receiver state machine, including these four sections.

Sync Section

In the Sync section, the receiver must correctly detect and synchronize to an incoming packet. If the receiver fails to do this, either the presence of the packet will not be known, and decoding will never begin, or the receiver will not be able to correctly decode incoming bits since it is not properly synchronized to the packet. In order to properly detect an incoming packet, a receiver must detect the energy in the channel, and when it reaches a certain threshold, decoding is started. An upper bound for this threshold is stated in the IEEE standard as being between -70 and -80 dBm, depending on the transmission power. Since the noise floor in most channels typically remains at -100 dBm, 802.11b receivers must correctly detect a packet if it is 20-30 dBm above the noise floor. Most receivers lower this threshold substantially, and are able to detect packets very close to the noise floor.

Once detection has occurred, synchronization is attempted. In 802.11b, each symbol is spread using an 11-chip Barker code. Due to the nature of the code, there is an enormous correlation peak with itself, but when correlating the sequence against even a slightly delayed version of itself, correlation is very small. Because of this, a simple rake receiver is used to correlate incoming data and to determine the beginning of a symbol. This rake receiver is 11 taps long, and samples the data every 1/11 microseconds. In the Sync stage, the greatest peak from this receiver is likely the beginning of a bit, and the receiver synchronizes itself

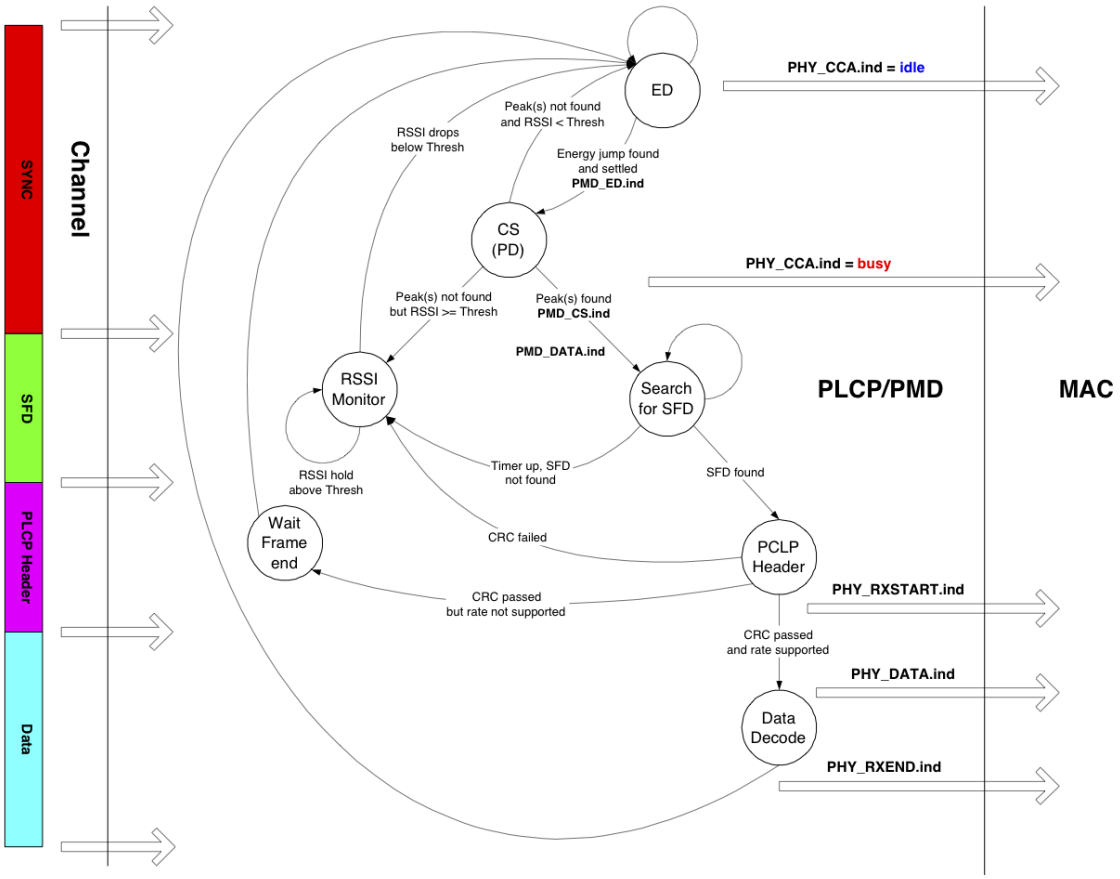


Figure 3.7: 802.11b PLCP Receiver State Machine, courtesy of [7]

accordingly.

An 802.11b preamble allows 128 bits, transmitted at 1 Mbps (or 56 bits, in the case of a short preamble) for the Sync stage. Additionally, the IEEE standard suggests that the packet detection occur in less than 15 microseconds [5], in order for there to be enough time for synchronization and other receiver initialization duties.

SFD Section

Once the packet is detected and synchronized, the receiver begins to search for the SFD sequence in the packet preamble. This sequence is merely a 16 bit specific sequence of

bits that informs the receiver that this is an 802.11b packet (the SFD is reversed for short preamble). If the receiver finds this exact sequence, then it begins decoding the header. Otherwise, packet decoding stops after a certain timeout.

Header Section

The packet header is used to gain information about the packet length and transmission rate. The header is transmitted at 1 Mbps (or 2 Mbps for short preamble), and ensures that the transmission bit-rate is supported, as well as ensuring that the CRC is passed. If either of these are not accomplished, decoding of the packet is stopped, and the packet is dropped.

Payload Section

Since the physical layer does not deal with the contents of an 802.11b payload, the only test at this section is to make sure that the 32-bit CRC at the end of the packet passes. If it does, then the payload data is passed up to a higher layer for further processing. If the CRC does not pass, the packet is dropped.

3.4.2 The Four Stage Simulink Interference Model

The Matlab Communications Toolbox includes a Simulink 802.11b WLAN PHY demo. This demo determines bit error rates for different channel conditions, and it assumes perfect synchronization. The underlying framework is accurate, and matches clear channel validations. In order for this to be used in interference modeling tests, additions were made to implement a four stage packet detection model. These stages were developed around the 802.11b receiver state machine. In the updated Simulink model, there are four stages that must be passed for a packet to be accepted. These stages are the energy detection stage, the SFD detection stage, the Header CRC stage, and the Payload CRC stage. They are designed to be IEEE 802.11b compliant. In addition to these stages, some other small additions to the original Simulink 802.11b WLAN PHY model were explored, but it was found that these additions either made little difference towards overall packet success rate, or were too

unpredictable for measurement at this stage, and they were dropped for the purposes of these tests.

Energy Detection Stage

For this stage, samples for the first 15 microseconds (this is the maximum amount of time for packet detection according to IEEE) of each packet are buffered, and the variance of these samples is taken, in order to receive an energy detection value. This value is compared to the given threshold (set at -76 dBm to comply with IEEE), and if it remains above the threshold, then this stage passes.

SFD Detection Stage

This stage is extremely simple, as simply the received bits in the SFD section are compared to the specific pattern. If there are no errors in these bits, then this stage passes.

Header CRC Stage

A CRC calculation and checking block was added to the original Simulink model in order to implement this stage. A CRC is calculated for the header bits and appended to the header (as described in the IEEE 802.11b standard) at transmission. At reception, the header CRC is determined, and if it passes the CRC check, then this stage passes.

Payload CRC Stage

Similarly to the header CRC stage, the Simulink model was updated in order to add CRC functionality, by adding a CRC block at the end of the packet payload. The CRC value of the received signal is calculated, and if the CRC check passes, then this stage passes.

Other Additions

In addition to the four stages added to the original Simulink 802.11b WLAN model, some implementation specific additions were explored in an attempt to create a model that more closely matches an actual receiver. These additions were automatic gain control, and offset

calibration. To implement the automatic gain control, the peak energy was detected in the SYNC field of the receiver. This peak was then used as a normalizing factor, and all incoming samples were scaled by this offset. To implement the offset calibration, four rake correlators were added in the SYNC stage, in a similar implementation to [16]. These correlators each had 11 taps and measured the samples 1/11 microseconds apart, but at slightly different offsets from each other. The correlator with the highest correlation value was determined as the proper bit synchronization. For the entire packet SYNC field, this detection ran, and the highest overall correlation peak was determined to be the proper position for bit synchronization of the packet. If this wasn't the correct synchronization for the incoming packet, then it was assumed that a bit error could eventually result, and the packet was dropped, due to an imminent header CRC or a payload CRC error.

It was found that the automatic gain control made absolutely no effect on packet error rate. This is true because the software receiver locates the nearest neighbor symbol for each received symbol, and scaling this symbol strength will retain the distance ratios to each symbol in the constellation map. The second addition, offset calibration, did effect packet success ratio to a considerable degree. However, it is difficult to be certain that an actual 802.11b receiver uses this method for offset calibration, and a decision was made to remove the aspects of the receiver that weren't necessarily standardized in the IEEE 802.11 document, so this addition was removed as well..

3.4.3 Clear Channel Validation

An important step in ensuring the Simulink Interference Model accurately reflects an actual 802.11 receiver is to test the simulator in additive white gaussian noise, through a clear channel validation. The original Simulink 802.11b WLAN model was set up to investigate the bit error rate behavior as a result of a noise channel, but there were no packet error rate calculations. The four stage Simulink model detailed above fixes this shortcoming, allowing packet error rate to be determined as a function of signal to noise ratio. To carry out the clear channel validation, 1000 packets were transmitted/received at each SNR ranging from -8 to 12 dB, in 1 dB increments. This was performed for each 802.11b bit-rate. Figure 3.8

shows the results of the Simulink clear channel tests.

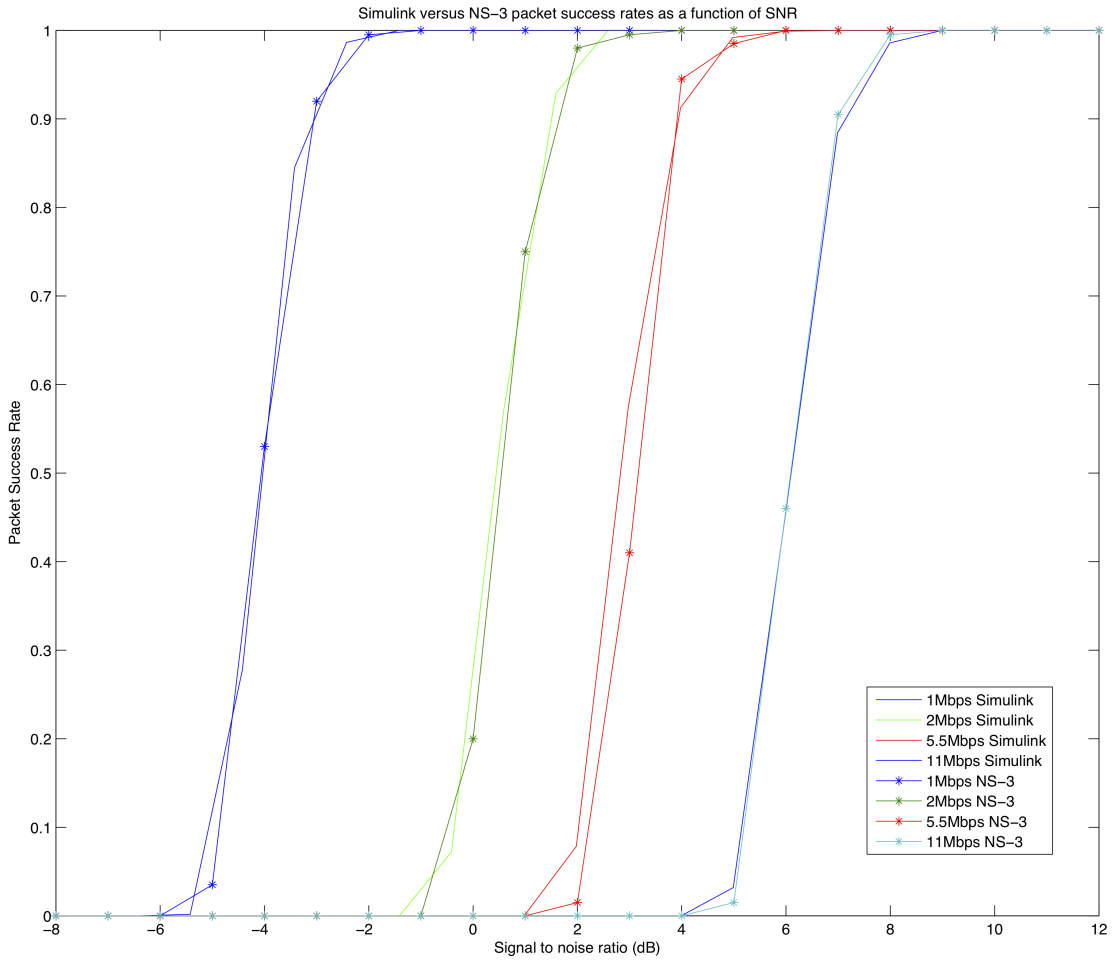


Figure 3.8: Clear Channel Behavior for Simulink versus ns-3. Both of these implementations match the packet success rate clear channel calculations from Equations 2.1-2.8

It is apparent that the four stage Simulink interference model indeed matches validated clear channel results. This suggests that the basic demodulation and chip de-spreading at the receiver is correct, as both of these aspects are large factors for receiver behavior in an additive white gaussian noise channel.

3.4.4 Single Source Interference Results

Once the clear channel case has been validated, interference tests can be performed. For these tests, an additional transmitter was created, and the transmitted interference packet was added to the original transmitter's packet. The interference packet was delayed by a specific amount prior to being added to the original packet, in order to determine the effects of packet interference on offset delay. The noise floor with respect to the intended transmitter was kept as a variable, and ranged among the critical region for each bit-rate (802.11 bit-rates typically are effected by noise at an SNR of -6 to 8 dB). Finally, a gain variable was introduced in the interference transmitter, so that the effects of packet reception from signal to interference ratio could be determined. For every SIR and offset delay combination, 1000 packets were transmitted/received.

3.4.5 1Mbps Interference Validation

To additionally prove that the Simulink Interference Model produces appropriate results, the results from the 1Mbps interference analysis in Section 2.2.6 were compared to the output of the Simulink tests. In the 1Mbps interference analysis, Equations 2.13-2.15 were used to calculate a packet error rate as a function of signal to noise ratio, signal to interference ratio, and interference offset τ. A bit sequence of 8192 intended bits was calculated, and the packet error rate resulting from these equations was directly compared to the Simulink results (for corresponding SNR, SIR, and τ offset values). In the Simulink Interference Model, 1000 packets were transmitted/received for every corresponding τ, SNR and SIR combination.

Figure 3.9 shows a comparison between packet error rates of the Simulink Interference Model, and the 1Mbps interference analysis. Signal to noise ratio is set to 0 dB, signal to interference ratio is set to -6 dB, and the interference offset τ is varied from 0 to 1 microsecond. The packet size in this experiment was set to 1024 bytes, or $(1024 * 8)$ 8192 bits. As shown in the comparison figure, both the Simulink Interference Model and the 1Mbps interference analysis form a beta distribution as a function of the τ offset. This distribution is very similar between both implementations, however the Simulink model

displays some noisiness in the packet error rates when τ is approximately between .1 and .9 microseconds. The reason for this noisiness is due to the pulse shaping filter implemented in the Simulink model. When the filter is removed, and tests are run, the Simulink model and the interference analysis show nearly identical results (as seen in Figure 3.9).

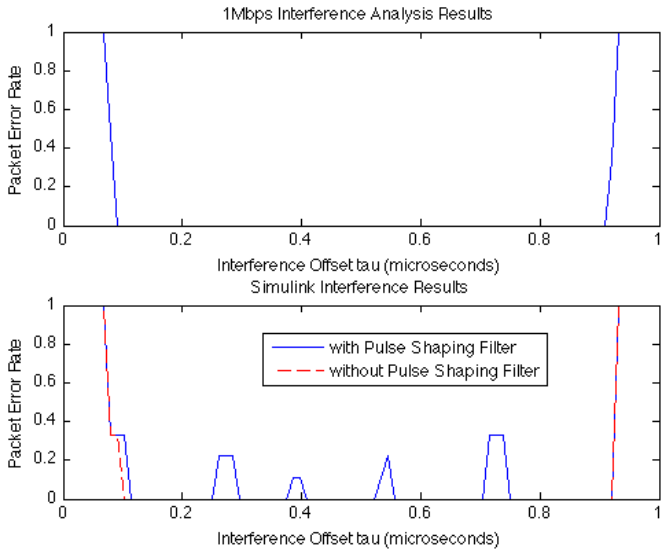


Figure 3.9: Comparison graph between results from the Simulink Interference Model (with and without the pulse shaping filter) and the 1Mbps Interference Analysis (determined from the packet error rate calculation in Equation 2.15). Packet error rates are shown as a function of τ interference offset. SIR = -6 dB, and SNR = 0 dB.

For the next set of comparisons, the packet error rates of both implementations were compared by averaging the τ offset as it ranged from 0 to 1 microsecond, and varying both signal to noise ratio and signal to interference ratio. Figure 3.10 shows a comparison between the Simulink Interference Model and the 1Mbps interference analysis as SIR is ranged from -8 to 8 dB (and SNR is kept constant at 0 dB), and Figure 3.11 shows a comparison between the two models as SNR is ranged from -5 to 5 (and SIR is kept constant at 0 dB). The two models exhibit very similar results in most cases, with only small increases in packet error rate in the Simulink model due to the pulse shaping filter, occurring at low SIR values. These results suggest that the Simulink Interference Model exhibits reasonable behavior at 1Mbps, and can be characterized by a simple packet error rate analysis.

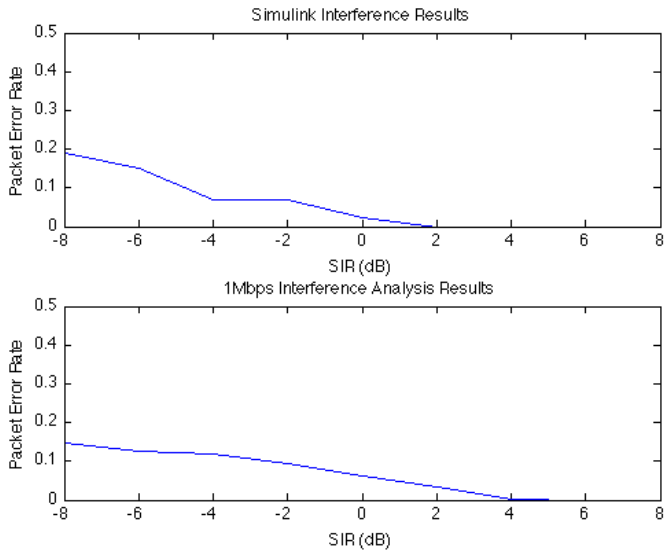


Figure 3.10: Comparison graph between results from the Simulink Interference Model and the 1Mbps Interference Analysis (determined from the packet error rate calculation in Equation 2.15). Packet error rates are shown as a function of SIR. For this case, SNR = 0 dB.

Further bit-rates were not tested (2Mbps, 5.5Mbps, and 11Mbps), but it is assumed that a similar analysis and testing could be performed with comparable results. The Simulink Interference Model allows for an easy change in bit-rate transmission, thus experiments validating other bit-rates can be performed with little change to the experiment methodology.

3.4.6 Potential Sources of Error

Despite extensive efforts to create a simulator that closely models actual hardware, there are still many potential sources of error that could effect results. These errors could result from differences in algorithm implementation, additional features in the hardware, inaccuracies introduced from hardware, and the transition from analog to digital signals. Of these reasons, signal synchronization is the area that likely separates the Simulink interference model from an actual hardware receiver to the highest degree. In a hardware 802.11b receiver, the main correlator can drift as much as 1/11 microseconds during the reception of a single bit [16]. Hardware receivers implement multiple rake correlators to help correct

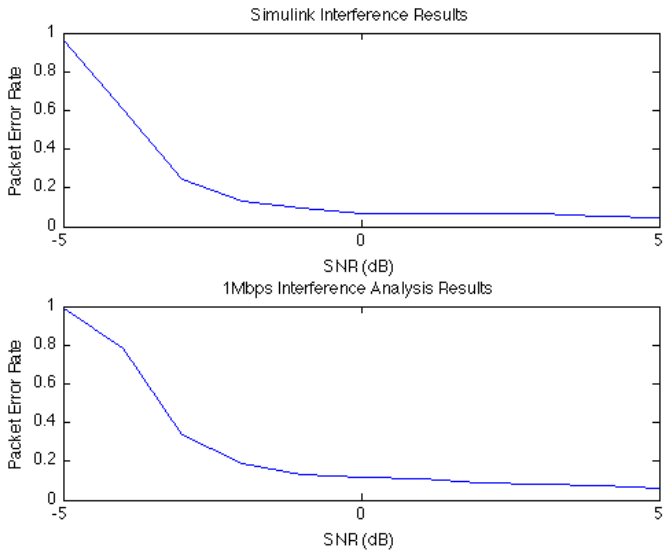


Figure 3.11: Comparison graph between results from the Simulink Interference Model and the 1Mbps Interference Analysis (determined from the packet error rate calculation in Equation 2.15). Packet error rates are shown as a function of SNR. For this case, SIR = 0 dB.

this drift. In noise, this is not typically a problem, as it is highly unlikely this noise will provide a false correlation peak that will force the receiver to incorrectly synchronize. In interference, however, it is possible for the receiver to synchronize to an interference bit, introducing the possibility for the receiver to decode an incorrect bit, and the original packet to be dropped. Without a thorough investigation on the true nature of this receiver drift, it is difficult to model this in software, and so the four stage Simulink interference model does not take into account this behavior. There is no data on the total effects of the receiver drift, but it is likely that this is the cause for the discrepancy between the Simulink model and with hardware, and may explain the inconsistencies between the Simulink and CMU emulator results.

Other sources of error include the automatic gain control, and in frequency offset estimation. The Simulink implementation of automatic gain control normalized every sample by the largest measured power value. This is a passive method of automatic gain control that only measures the channel power for a short time and normalizes by a calculated constant.

This is similar to the actual implementation for automatic gain control, however, a difference occurs in the possibility for overflow and underflow. In the Simulink model, all samples are propagated as real numbers, while a hardware receiver analog-to-digital converter merely converts input signals to a certain number of bits. Automatic gain control attenuates the higher order bits in a hardware receiver, and interference arriving later during packet reception can then cause either a conversion overflow or underflow, as described in [8]. This could cause differences in behavior that the Simulink model did not take into account. Also, all 802.11b receivers utilize a frequency offset estimation that the Simulink model does not explore at all. This is likely cause of only marginal differences between hardware receivers and the simulator.

For future experimentation, it is necessary for the Simulink model to address these areas of potential error.

Chapter 4

PHYSICAL LAYER CAPTURE EFFECT

4.1 Introduction

The physical layer capture describes the behavior of a receiver when capturing a packet. While the physical layer capture behavior in many network simulators, such as ns-3, is to synchronize to an initial packet, and to not accept any additional packets until processing on the initial packet is finished, prior research [10] has shown that this behavior is not correct for all receiver implementations. According to [10], a receiver can exhibit behavior referred to as the 'physical layer capture effect', which is the case in which a receiver can synchronize to a new packet even if it is already synchronized to a current packet. In [10], it was found that this situation occurred whenever the followup packet was of a higher signal strength than the original packet. While it is difficult to truly determine whether this effect matches threshold based behavior, or if more complicated models are necessary, this study shows that this effect exists, and that it accounts for a large difference in packet reception behavior.

4.2 Experimental Results

From ns-3 interference tests, it was found that the simulator does not model this physical layer capture effect, and that in all cases, when a receiver was synchronized to an initial packet, it would not synchronize to an incoming packet. Since this is a clear difference between ns-3 results and data from [10], these differences must be reconciled.

To further verify the capture effect, results from the CMU Emulator Interference tests were consulted. There was little evidence of the physical layer capture effect for the 1 Mbps and 5.5 Mbps cases in the CMU Emulator results, and the rate of success for this capture effect remained below 5 percent under all signal to interference ratio combinations and interference time delay offsets that were tested. For the 2 Mbps and 11 Mbps cases, evidence

of the capture effect became evident in high signal to interference ratios. Figure 4.1 shows the example receiver behavior when interference occurs for 2 Mbps at a signal to interference ratio of 8 dB. The negative time delays represent collisions where the interference was synchronized to the receiver before the intended packet was received. The positive packet success rates when time delay is negative provides clear evidence of the physical layer capture effect. In this case, it is evident that the capture effect contributed for packet success rates of greater than 30 percent.

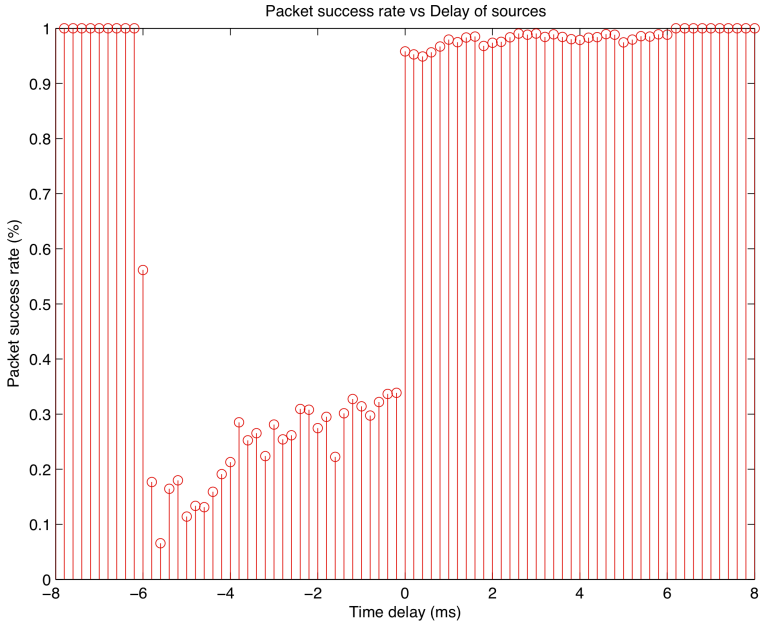


Figure 4.1: CMU Emulator interference test for 2 Mbps and SIR = 8 dB. Figure shows packet success rate versus time delay of interference source.

Figure 4.2 shows another example from the CMU Emulator interference tests, with an 11 Mbps transmitted signal with a signal to interference ratio that ranges from 4 dB to 8 dB. Again, the negative time delays represent collisions where the interference was synchronized to the receiver before the intended packet was received, and the packet success rates during these negative delays represent evidence of the physical layer capture effect. In this case, when the packets are transmitted at a signal to interference ratio of 4 dB, there is no evidence of the capture effect. When the signal to interference rate is increased, the packet success

rate for the capture effect scenario increases, with a packet success rate of approximately 10 percent when signal to interference rate equals 8 dB.

These results show evidence of the capture effect, but under different cases as those witnessed by [10]. Where [10] observed the capture effect occurring in most situations where the signal to interference ratio was above 0 dB, the CMU Emulator interference results only perceived the capture effect when signal to interference ratio was very high (6 dB and above), and only for specific bit-rates. Until these differences are reconciled, it is clear that an interference model implementation of the capture effect must be able to easily account for either result.

4.3 ns-3 Physical Layer Changes

While it is confirmed that the ns-3 main repository currently has no model for the physical layer capture effect, some ns-3 contributed code has this functionality implemented. This project includes wifi enhancements to the current ns-3 model, with a focus on the wifi physical layer (located at: http://code.nsnam.org/timob/ns-3-wifiex/). The basic physical layer included in ns-3 is preserved here, with simple code additions to the RX state in order to include the capture effect. In this implementation, if a node is in the RX state (receiving an initial packet), it can drop the initial packet in favor of a second packet, given that the second packet's signal strength is greater than a certain threshold. There is also the ability to set two different thresholds for two capture effect cases (the case when a second packet arrives at a receiver when it is decoding the first packet's preamble, or the case when a packet arrives when the receiver is decoding the first packet's payload). Flags can be set or removed for these cases, so the capture effect can be removed for one or both of these cases.

The simplicity of this threshold based approximation for the capture effect is suitable as an initial implementation. The current capture effect implementation in the ns-3 wifi enhancements side-project only include thresholds for 1Mbps and 2Mbps, and both of these thresholds use data from the IEEE 802.15 on 802.11 interference study [6]. In this case, a threshold for a specific bit-rate is determined as the signal to noise ratio where bit error rate is less than 10^{-9}, which according to [6] is 5 dB for 1Mbps and 8 dB for 2Mbps. These thresholds are equivalent for both capture effect cases. As contrast, the current ns-2

repository [2] sets defaults for these thresholds at 5 dB for the case when the second packet arrives when the receiver is decoding the first packet's preamble and 10 dB for the case when the second packet arrives while the receiver is decoding the first packet's payload. These thresholds are the same for all bit-rates. While CMU Emulator Interference data was not collected at high enough signal to interference ratios to determine if the ns-2 capture effect thresholds are correct, it is recommended that these defaults be extended to ns-3, as they represent upper limit thresholds for all bit-rates.

4.4 Conclusion

The physical layer capture effect is a receiver capture behavior that can account for up to 50 percent of the throughput in a wireless network [10]. The presence of the effect is evident in both the CMU Emulator interference tests performed in this thesis, as well as the hardware experiments performed by [10]. In addition, capture effect functionality is simple to add to the ns-3 network simulator, and does not effect any behavior outside of the specific addition. It is clear then, that accounting for the capture effect is beneficial for ns-3 and would allow for a more accurate interference model.

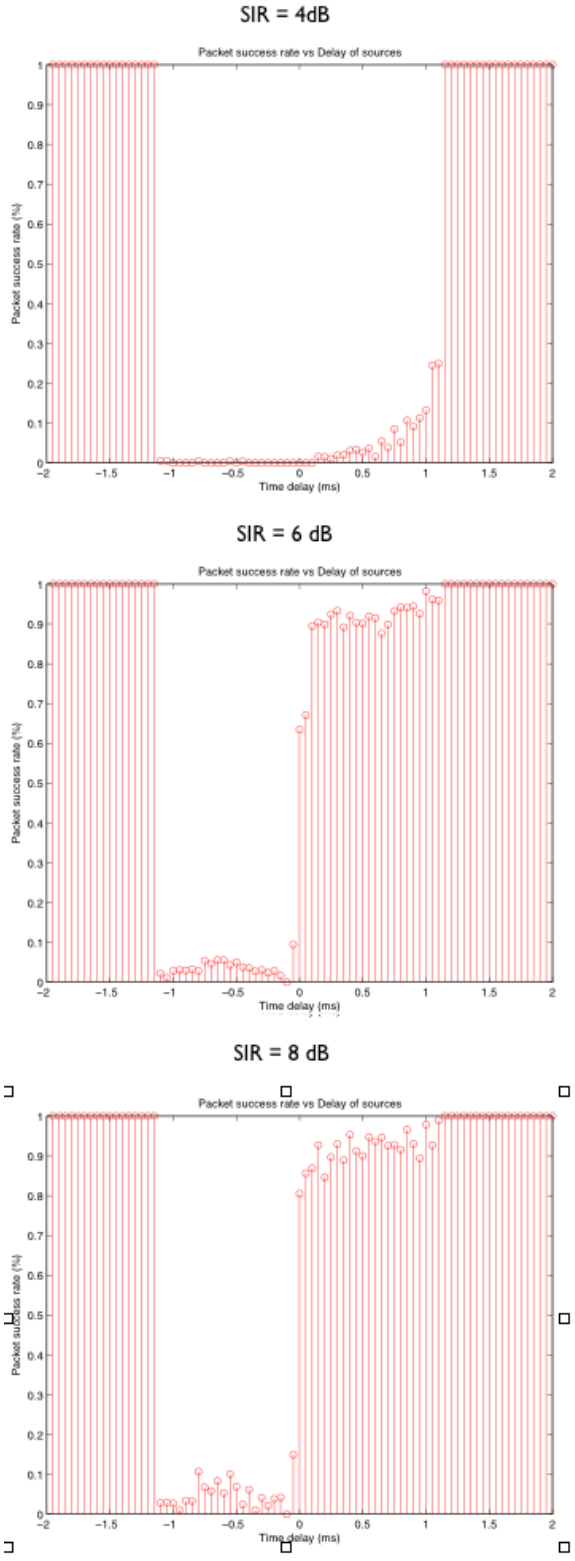


Figure 4.2: CMU Emulator interference test for 11 Mbps and SIR ranging from 4 dB to 8 dB. Each graph shows packet success rate versus time delay of interference source.

Chapter 5

PREAMBLE INTERFERENCE MODELING

5.1 Introduction

Preamble interference refers to interference that occurs when a receiver is decoding a packet's preamble. This is perhaps the most difficult area of interference to model, because there are many duties that the receiver must complete during the preamble in order to properly synchronize to the incoming packet, and the methods to accomplish these duties can greatly range between receiver implementations. These duties include energy detection, automatic gain control, and packet synchronization, three functions that are not necessarily robust to interference. Also, as described in [8], specific types of interference during a preamble can greatly effect packet error rate behavior, as the gain detection and synchronization functions can be easily exploited. The ns-3 Wireless Simulator currently has no functionality to calculate any of these interference behaviors during a packet's preamble, which is clearly inaccurate.

5.2 Experimental Results

The preamble interference results presented in [9] (these results are introduced in Section 2.2.5, and reproduced here for convenience. Please refer to Figure 5.1), provide a very useful reference for packet preamble interference. Figure 5.1 shows the reception behavior of a packet under interference, when the interference source is delayed by up to 96 microseconds (the length of the intended packet preamble and header).

The experiments were performed with packets operating with a short preamble, which allocates 72 bits, transmitted at 1 Mbps, for the preamble. This implies that it takes 72 microseconds for a short preamble to be received. However, the results show that it is imperative that the receiver synchronize to the packet without interference during the first 32 microseconds of its receipt (at the 1Mbps and 2 Mbps bit-rates), with a 4 dB improvement

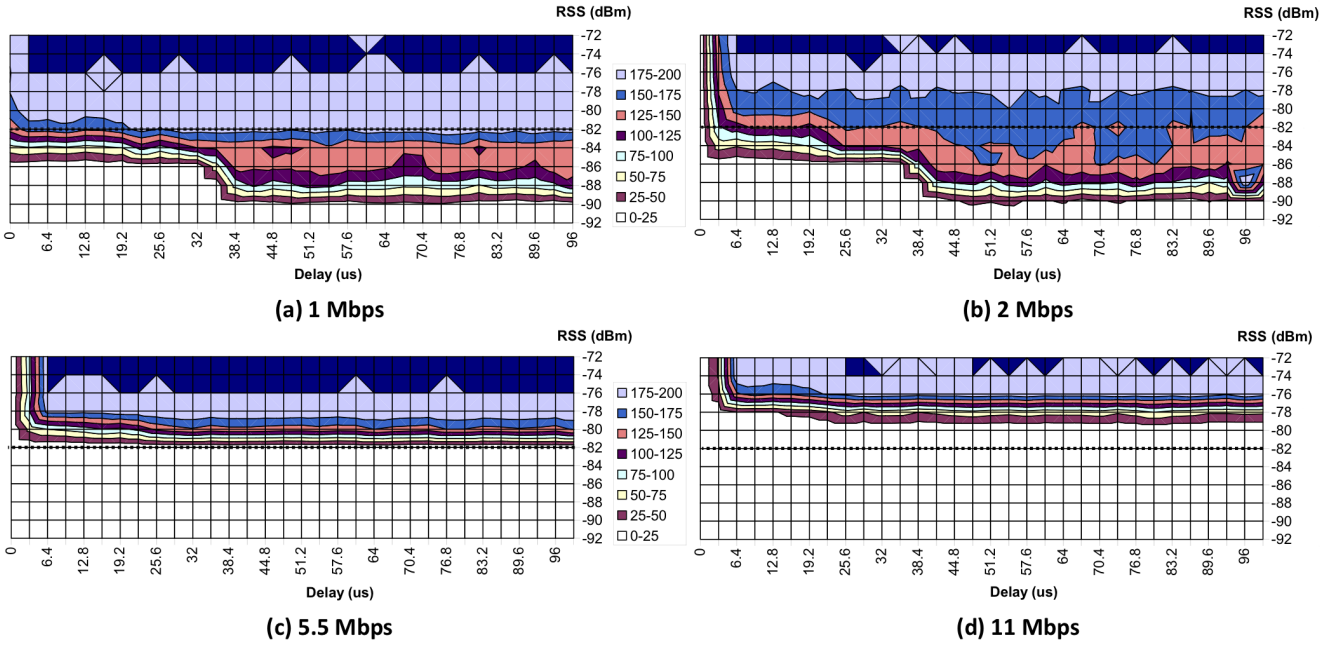


Figure 5.1: Packet capture versus interference delay (Courtesty of [9]). For the 1Mbps and 2Mbps bit-rates, interference negatively impacts the reception probability to a greater degree when it starts during the preamble.

in packet success rate behavior when the interference delay is greater than 32 microseconds. For the 5.5Mbps and 11Mbps bit-rates, it is necessary to receive the first 3 microseconds of the packet without interference, after which the probability of packet receipt increases dramatically. For these higher bit-rates, it appears that the effects of preamble interference are not substantial past these first 3 microseconds, thus it is likely that payload interference dominates preamble interference effects in these cases.

While the CMU Wireless Emulator interference tests measured in this thesis do not have a timing accuracy precise enough to measure and corroborate with the exact preamble interference results from [9], they can be used to generally determine whether both tests produce similar results. To do this, the results from Figure 5.1 were referred with interference delay values greater than 46 microseconds, and was compared to corresponding CMU Emulator data gathered in this thesis. The results in Figure 5.1 are constant for nearly 50 microseconds (46 to 96 microseconds), and therefore can be accurately compared to the

CMU Emulator interference tests gathered in this thesis, despite the timing inaccuracies shown in these experiments (see Chapter 3).

Figure 5.2 shows the result of comparing the data in [9] and the CMU Emulator interferences tests observed in this thesis, when the interference offset delay was equal to roughly 96 microseconds. For curves from the [9] results, the raw data was consulted, and an average was determined for all interference offset delay values between 44.8 microseconds and 96 microseconds, which consist of approximately 3,600 packets per bit-rate and signal to interference ratio combination. In the CMU Emulator interference tests, five experiments were performed for every bit-rate and signal to interference ratio combination. These experiments were compared and averaged, to insure that results are accurate and repeatable. The resulting data was then averaged over all interference offset delay values ranging from 60 microseconds to 120 microseconds, which consist of roughly 1,080 packets per bit-rate and signal to interference ratio combination.

The results match reasonably between the two sources, with packet success rate curves that generally corroborate for all bit-rates. This is an important result, especially since the methods to generate the data for both sources were extremely varied, in two areas. The first methodology difference is in the experiment setup. For the results in [9], a special FPGA function was made available that changed the channel behavior on the order of microseconds, where the interference results collected in this thesis do not use this function. The second difference is in a slight modification to the experiment. For the results in [9], all bit-rates were interfered with a 1Mbps signal, while in the results from this thesis, all bit-rates were interfered with a signal of the same bit-rate. This provides evidence that differing bit-rate interference behaves similarly to same bit-rate interference. Finally, the corroboration of both sets of data ensures that the CMU Emulator behavior has remained consistent, and that the preamble interference results determined from [9] remain accurate.

5.3 ns-3 Preamble Interference Additions

In order to allow for preamble interference functionality in ns-3, it is recommended that the results from [9] be added as a lookup table. This lookup table would resolve events with single source interference. While it is feasible that multiple source interference would occur

in a wireless network system, this single source interference contributes to the majority of interference cases, and as a result this lookup table would provide a beneficial first step towards the improvement of the preamble interference modeling in ns-3. Appendix B displays the raw data from the interference results in [9], in the form that the lookup table would be integrated. There are two axis for the table: signal to interference ratio, and interference offset delay (up to the length of the packet preamble and header). In this table's implementation, the assumption is that noise is low (these tests were conducted with the transmitter ranging from -92 dBm to -72 dBm, where the noise floor is approximately -100 dBm). Cases when noise is low (approximately equal or below the noise floor), the current ns-3 noise as interference model would be used to calculate packet error ratio.

The addition of this lookup table should be implemented as an alternate interference-helper class in ns-3, to allow for the easy selection of this lookup table based interference modeling approach, and the current ns-3 approach. Since ns-3 currently does nothing to resolve preamble interference, it is clear that the addition of this data would improve the simulator.

5.4 Conclusion

While preamble interference is perhaps the most complicated aspect of interference to calculate in reality, ns-3 currently does nothing to determine the behaviors of this interference. To remedy this, a lookup table from the results gathered in [9] is presented, and recommended for implementation in ns-3. This lookup table has been roughly validated with additional CMU Emulator interference tests, and its addition would prove to beneficially change the preamble interference modeling in ns-3.

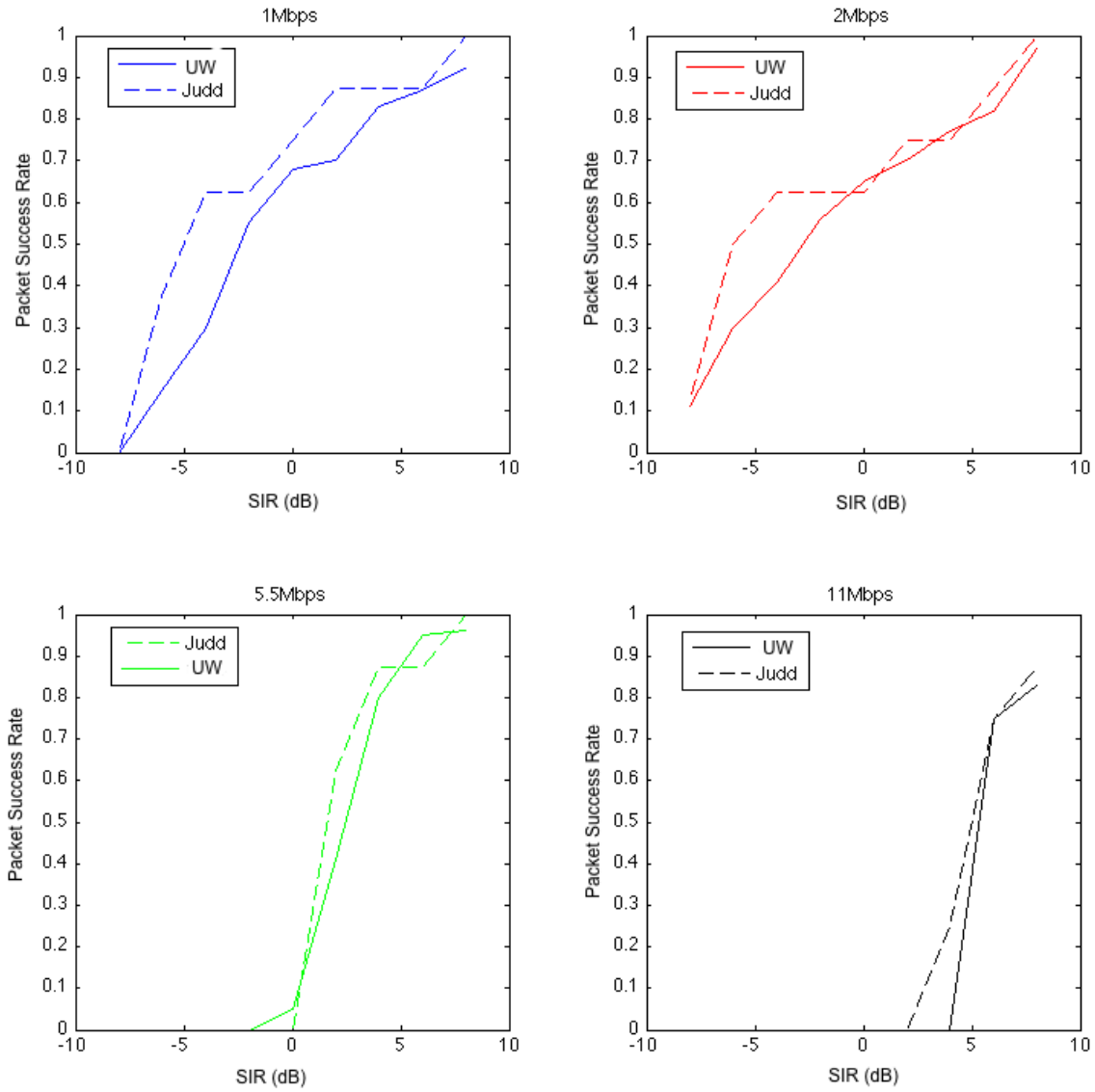


Figure 5.2: Packet success rate versus signal to interference ratio compared between the CMU Emulator interference tests gathered in this thesis (referred to as 'UW') and the results from [9] (referred to as 'Judd'). These packet success rates were measured when interference offset delay was varied between roughly 60 to 120 microseconds for the UW results and 44.8 to 96 microseconds for the Judd results.

Chapter 6

PAYLOAD INTERFERENCE MODELING

6.1 Introduction

Payload Interference refers to the interference that occurs when a receiver is decoding a packet payload. The occurrence of this type of interference is generally more likely than preamble interference, especially because a packet payload bit length is typically orders of magnitude greater than the packet preamble bit length. Because of this, modeling payload interference is perhaps the most important aspect of creating an accurate interference simulator. Many network simulators use either a threshold based approach to interference modeling, or treat the interference as additional noise. Both methods do not necessarily model interference accurately, so it is important to determine the specific conditions that the current interference model is inaccurate. This chapter studies where modeling interference as noise is appropriate, and where a new model must be implemented, as well as suggesting a lookup table to be used for modeling the most common interference case: single source payload interference.

6.2 Treating Interference as Noise

The ns-3 Network Simulator currently calculates payload interference by treating all interference as noise, and determining bit error rates and packet error rates due to common noise behavior equations for 1Mbps, 2Mbps, 5.5Mbps, and 11Mbps. In order to determine the effectiveness of this implementation, it becomes necessary to specify in what situations this is an adequate approximation. To provide this information, the Simulink Interference Model was initially consulted. Data was gathered on packet error rates as a function of signal to noise ratio and signal to interference ratio, for the 1Mbps bit-rate. The number of bits that the interference occurred over was set to 8192 bits, and results were averaged for tests with interference offset delay values ranging over 1 microsecond. Results from this

experiment were then compared to results from interference tests on ns-3, with equivalent settings for SNR, SIR, and bits that interference occurred over. The differences in packet error rates from these two implementations is shown as a function of both SNR and SIR, in Figure 6.1.

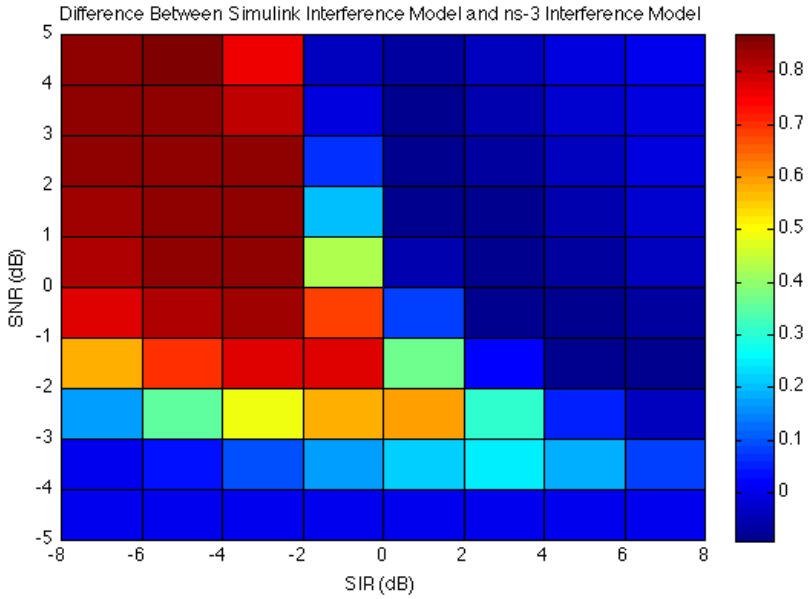


Figure 6.1: Heat map showing the difference in packet error rate between the Simulink Interference Model and the ns-3 interference implementation of treating interference as noise, tested at 1Mbps. The colors refer to the determined ns-3 packet error rate, subtracted by the packet error rate from the Simulink Interference Model, at each SIR and SNR value. The ns-3 implementation matches the packet error rate calculations as described in Equations 2.1 and 2.2.

This figure shows that when SNR is very low (-4 dB and below), there is little difference in tests with an actual interference source, or with approximating the interference as noise. As SNR is increased, however, the difference between the two implementations becomes large, especially at low SIR. The ns-3 interference model exhibits very high packet error rate values in the high SNR, low SIR region (where there is a low level of noise and a high level of interference), however the Simulink interference model shows relatively low packet error rate values in this region. The differences between the two implementations in these high SNR, low SIR cases suggest that it is important for this region to be studied further.

As SIR is increased, and is greater than 0 dB, the difference between implementations is lessened, for all SNR. When SIR is sufficiently high, both the ns-3 interference and Simulink interference model responses are very similar. The heat map shows that for 1Mbps, the current ns-3 interference model corroborates with the Simulink interference model either when signal to interference ratio is sufficiently high (greater than 0 dB in this case), or when signal to noise ratio is sufficiently low (smaller than -4 in this case), and for most other cases, the two models do not match.

6.3 Single Source Interference Experimental Results

It is clear that there are specific cases where the ns3 interference model results must be studied further. To provide alternate data for one of these cases, the case of single source interference in a low noise channel, the CMU Wireless Emulator was used. Interference tests from the emulator were used to compare against the ns-3 interference as noise implementation, as well as to provide a lookup table as an alternative interference model to more accurately model single source interference in network simulators.

Appendix A shows figures that describe the results of the CMU Emulator tests, grouped by bit-rate. Each figure represents a certain pathloss combination (between the transmitter to receiver channel and the interference to receiver channel), with the difference in pathloss between the two channels equaling the signal to interference ratio.

For 1Mbps, the payload interference behavior of these figures increases gradually, with the packet success rate proportionately increasing as the delay offset of the interference packet increases. This behavior is similar for 2Mbps, however the packet success rate slopes are much shallower for most signal to interference ratios. In 5.5Mbps, packet success rate remains relatively constant as interference delay offset is increased, with large changes only occurring at very low or very high interference delay offsets. At 11Mbps, the slope of the packet success rate increases as the interference offset delay increases. This only occurs for one signal to interference ratio combination, and with higher signal to interference ratios, the packet success rate is very high. The behavior of the packet success rate slopes for all bit-rates except 11Mbps do not match the exponential behavior resulting from modeling interference as noise. In ns-3, the packet success rate increases exponentially as interference

offset delay is increased, for all bit-rates. The only evidence of this type of behavior is seen in the 11Mbps experiment, when signal to interference ratio equals 4 dB.

Additionally, for low bit-rates (1Mbps and 2Mbps), the average packet success rate transition from low to high, as signal to interference ratio is increased, is very gradual (the transition occurs over a 10 dB range). This contrasts to results from ns-3 tests, where this transition occurs quickly (approximately a 3 dB range). At 5.5Mbps and 11Mbps, the average packet success rate transition from low to high occurs quickly, but the packet success rate values differ by a large amount from the CMU Emulator interference tests and the ns-3 interference model.

Figure 6.2 shows the results of a comparison between the CMU Emulator interference experiments and the ns-3 interference tests. In these graphs, each point represents the packet success rate averaged over all interference offset delay values. From this figure, it is clear that the ns-3 interference model is generally inaccurate for the signal to interference range of -8 dB to 8 dB, and when signal to noise ratio is high. This shows the importance of using the data from the CMU Emulator as a lookup table, as there is a great difference between the current ns-3 interference model, and the hardware CMU Emulator tests.

6.4 ns-3 Payload Interference Additions

As shown in this chapter, there are many areas of interference where the current ns-3 implementation of treating interference as noise is insufficient. To fix one such very common scenario, the case of single source interference in a low noise channel, it is suggested that the data from the CMU Wireless Emulator interference tests be used as a lookup table. This lookup table is available in Appendix A, and it operates in a similar fashion to the preamble interference lookup table suggested in Chapter 5. The table serves as an alternate implementation to the current ns3 interference model. Additionally, the lookup table is used only for the case of single source, same channel interference, with the current ns-3 noise as interference model to be used for all other cases (until similar data is generated for other cases of interference that cannot be treated as noise). The addition of this lookup table significantly improves the ns-3 interference implementation for this interference case.

6.5 Conclusion

The majority of the ns-3 interference simulation occurs in payload interference modeling. Despite the sharp focus on payload interference, the current ns-3 implementation of treating interference as noise is not accurate for all interference cases. These cases have been investigated, with results showing that single source interference scenarios when signal to interference ratio is low and signal to noise ratio is high do not corroborate with interference experiments. There are cases when the ns-3 implementation does match with these interference experiments however, and evidence is shown that suggests when signal to noise ratio is sufficiently low, or when signal to interference ratio is sufficiently high, then both models match. To improve the interference modeling in ns-3, data is gathered on single source interference from the CMU Wireless Emulator, and saved as a lookup table. This lookup table improves the single source interference modeling of ns-3 dramatically.

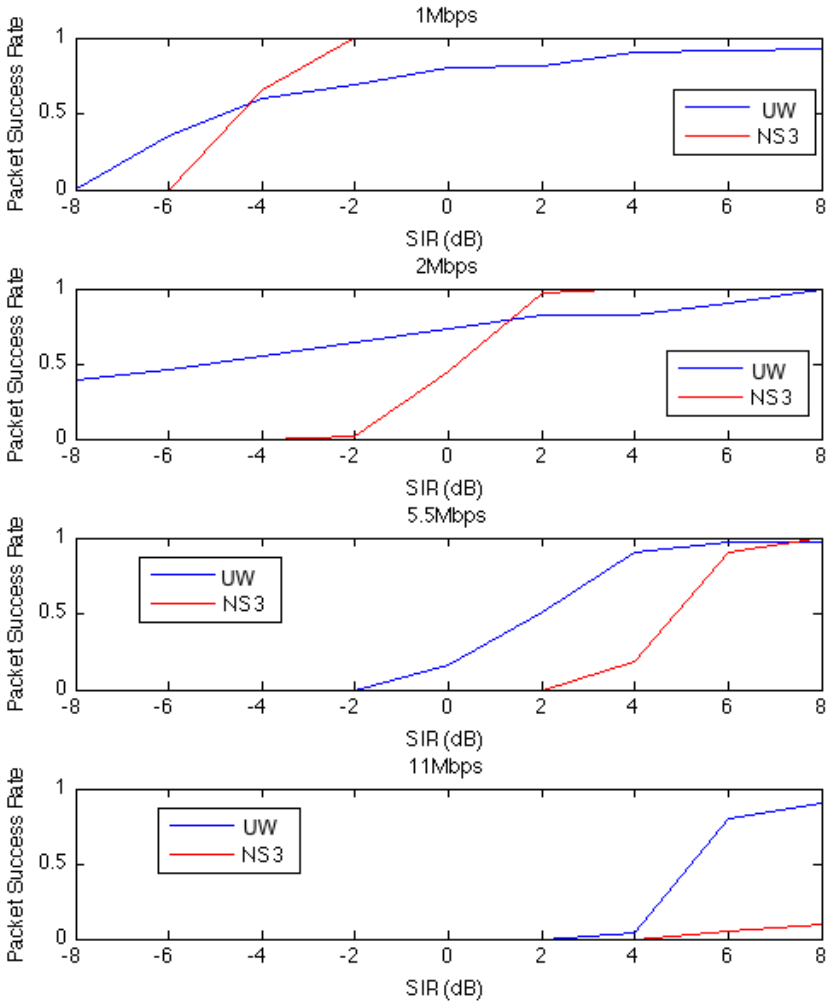


Figure 6.2: Packet success rate versus signal to interference ratio compared between the CMU Emulator interference tests, and the ns-3 interference tests. Each graph represents a unique 802.11b bit-rate, and at each point represents the packet success rate averaged over all interference delay values. The signal strength is kept constant at -70 dBm for all tests (noise floor is approximately -100 dBm). The ns-3 packet success rate values match the mathematical calculations from Equations 2.1-2.8.

Chapter 7

CONCLUSION

7.1 Conclusions

Current research on the subject of 802.11b receiver behavior in interference conditions is limited, and does not provide an in-depth view of these behaviors. This thesis provides a deeper understanding of the effects of interference by investigating both hardware and software experiments for interference. Results from the Carnegie Mellon University wireless emulator were gathered to be used as a lookup table in future error rate models for interference implementations in network simulators. Also, a Simulink interference model was created in order to obtain an understanding of exactly why and where interference modeling contrasts from modeling a receiver in clear channel conditions. As the effects of noise are lessened in a system, the various characteristics of the specific hardware receiver begin to dominate the model, which is difficult to simulate in software. Finally, interference modeling changes are recommended for the open source network simulator ns-3. These changes include a simple physical layer change in the receiver state machine, lookup tables for the calculation of single source interference, and an analysis to determine in which cases the current ns-3 payload interference model is acceptable, and in which cases it must be improved.

It is not feasible for a single thesis to completely explore the entire subject of 802.11b interference. Instead, by focusing on the most common interference case (single source, same channel interference), this thesis provides important insight into interference characteristics, as well as presents a viable interference model to be used in network simulators. With the given interference model, the error rate implementations in many network simulators would be much more accurate, and would more closely match hardware.

7.2 Future Work

The work in this thesis can be extended in many directions. First, the CMU emulator lookup table used for network simulators can be improved by adding interference cases other than single source, same channel, same bit-rate interference. Additionally, interference effects from sources other than 802.11b transmitters can be investigated. For example, there is widespread use of bluetooth, and often within range of 802.11b receivers (most laptops have the ability to transmit bluetooth and 802.11). The effects of bluetooth interference are likely different than those of 802.11b interference, and this thesis did not cover this subject. In order to improve the Simulink interference model, a focus could center on investigating the behaviors of various 802.11b hardware receivers. A study on bit synchronization and drift in 802.11b hardware receivers would contribute considerably to the Simulink interference model efforts. Finally, the practices in this thesis could be used to study other types of receivers, such as 802.11a/g/n, and could investigate the effects of interference on packet reception for these types of wireless receivers.

BIBLIOGRAPHY

[1] Carnegie mellon university wireless emulator. `http://www.cs.cmu.edu/~emulator/`.

[2] The ns-2 network simulator wiki. `http://nsnam.isi.edu/nsnam/index.php/Main_Page`.

[3] The ns-3 network simulator. `http://www.nsnam.org/`.

[4] Simulink - simulation and model-based design.

[5] Jan Boer. Direct sequence spread spectrum physical layer specification ieee 802.11. *Lucent Technologies*, 1996.

[6] IEEE LAN/MAN Standards Committee. Part 15.2: Coexistence of wireless personal area networks with other wireless devices operating in unlicensed frequency bands, August 2003.

[7] Mustafa Ergen. Ieee 802.11 tutorial. `http://www.eecs.berkeley.edu/~ergen/docs/ieee.pdf`, 2002.

[8] Ramakrishna Gummadi, David Wetherall, Ben Greenstein, and Srinivasan Seshan. Understanding and mitigating the impact of rf interference on 802.11 networks. In *In Proc. of Sigcomm07*, pages 385–396, 2007.

[9] Glenn Judd and Peter Steenkiste. Characterizing 802.11 wireless link behavior. *Wireless Networks*, 2008.

[10] Andrzej Kochut, Arunchandar Vasan, A. Udaya Shankar, and Ashok Agrawala. Sniffing out the correct physical layer capture model in 802.11b. *12th IEEE International Conference on Network Protocols*, 2004.

[11] Hui Ma, Eiman Alotaibi, and Sumit Roy. Analysis and simulation model of physical carrier sensing in ieee 802.11 mesh networks. *OPNETWORK Conference*, August 2006.

[12] Guangyu Pei and Tom Henderson. Validation of ns-3 802.11b phy model. `http://www.nsnam.org/~pei/80211b.pdf`, 2009.

[13] John G. Proakis. *Digital Communications*. McGraw-Hill, 2001.

[14] Michael B. Pursley and Thomas C. Royster. Properties and performance of the ieee 802.11b complementary-code-key signal sets. *IEEE TRANSACTIONS ON COMMUNICATIONS*, 57(2):440–449, February 2009.

[15] Vinay Sridhara, Hweechul Shin, and Stephan Bohacek. Performance of 802.11b/g in the interference limited regime. *Computer Communications*, 2008.

[16] Shravan Kumar Surineni. Software defined radio (sdr) based implementation of ieee 802.11 wlan baseband protocols. 2004.

Appendix A

CMU WIRELESS EMULATOR SINGLE SOURCE INTERFERENCE EXPERIMENTAL RESULTS AND LOOKUP TABLE

Following are the results from the CMU Wireless Emulator interference experiments. The graphs in each figure are calculated by finding the timestamp differences between each packet collision, as described in Section 3.3.4. Each figure represents a specific 802.11b bit-rate, and each graph on the figures details the packet success rate behavior given a specific signal to interference ratio, which ranges from -8 dB to 8 dB.

Following the visual result graphs are the lookup tables recommended to be integrated with the ns-3 payload interference implementation. Each table gives data for a specific bit-rate, where signal to interference ratio and interference offset delay are input.

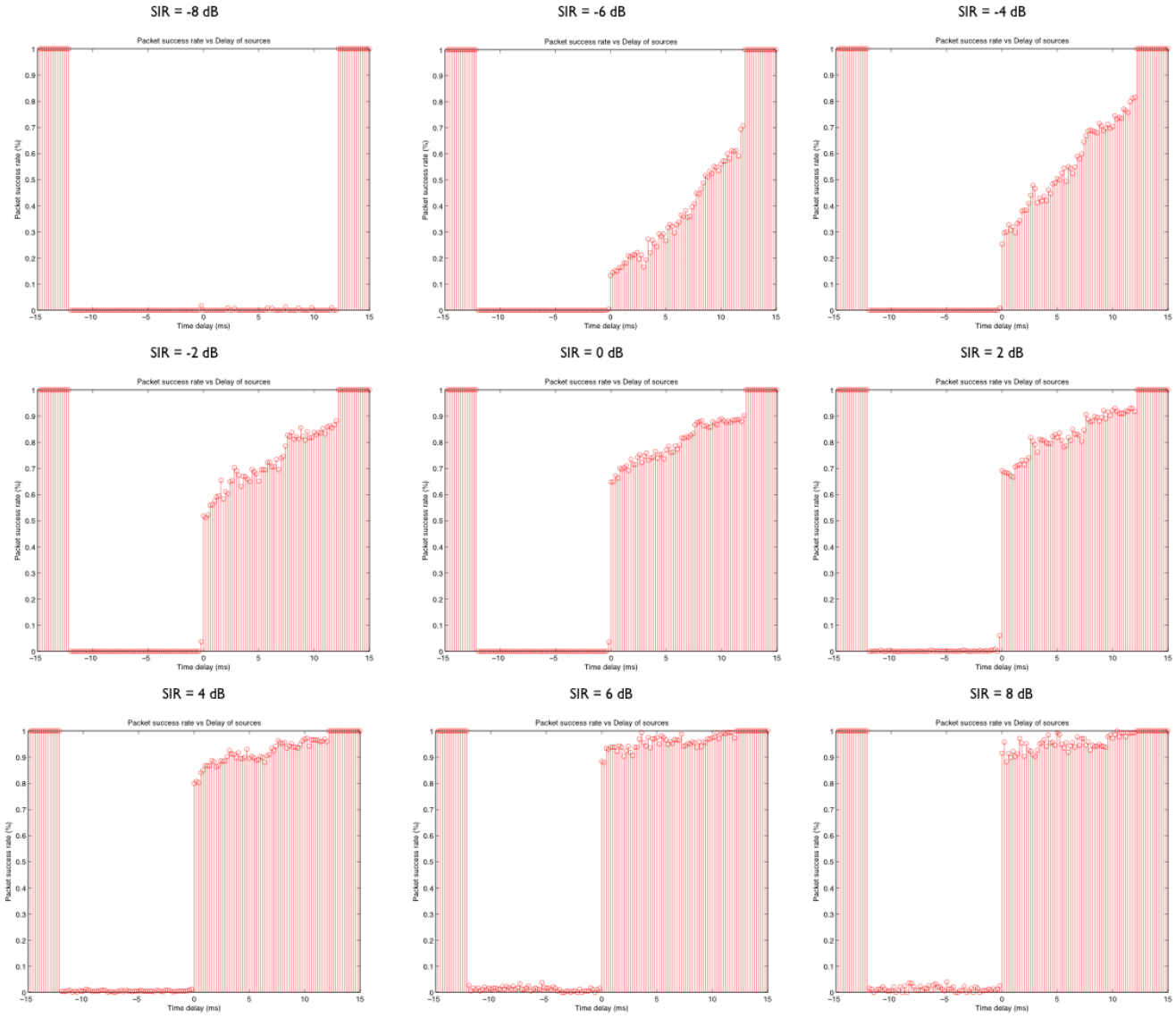


Figure A.1: CMU Wireless Emulator Interference Results for 1 Mbps

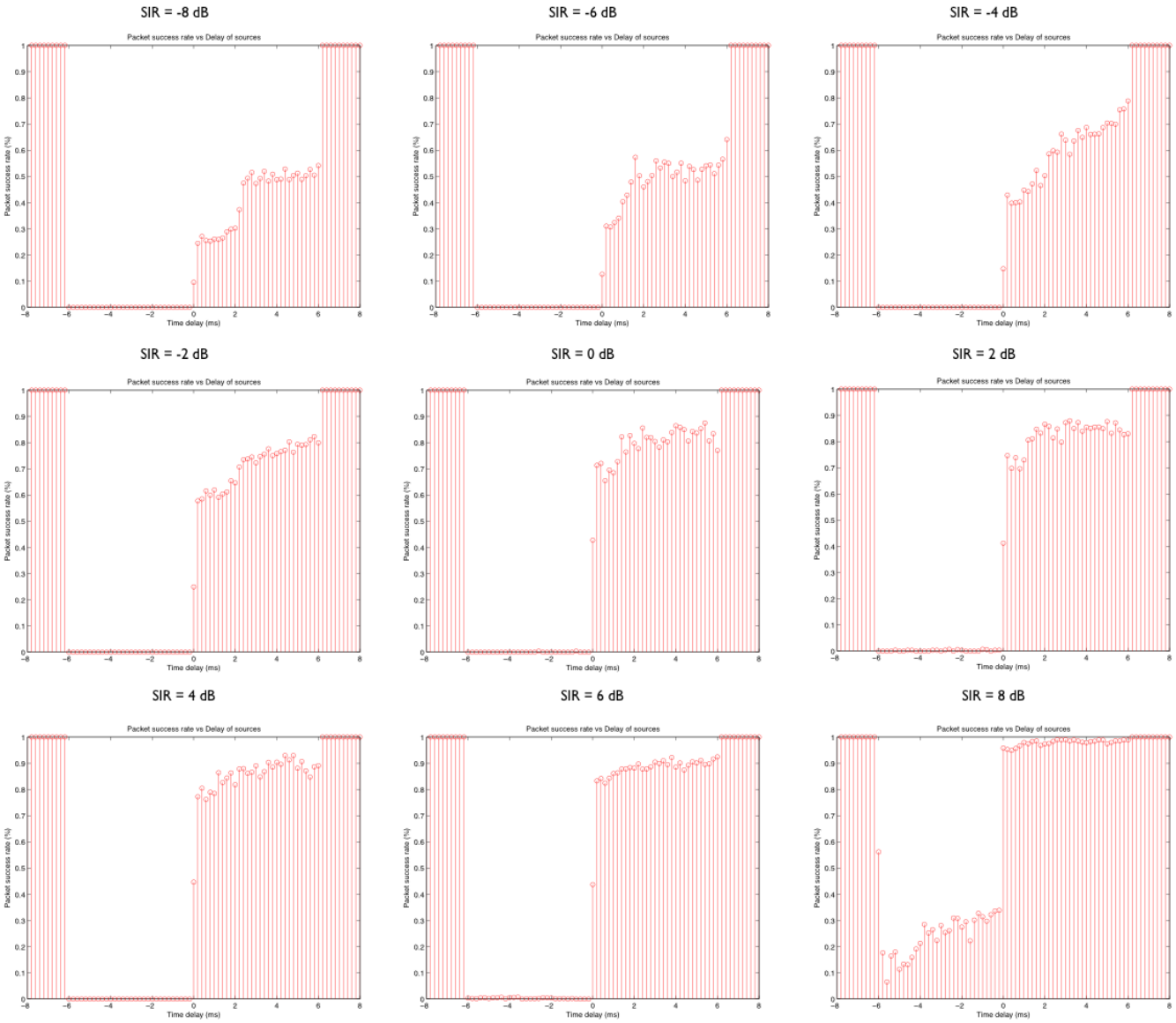


Figure A.2: CMU Wireless Emulator Interference Results for 2 Mbps

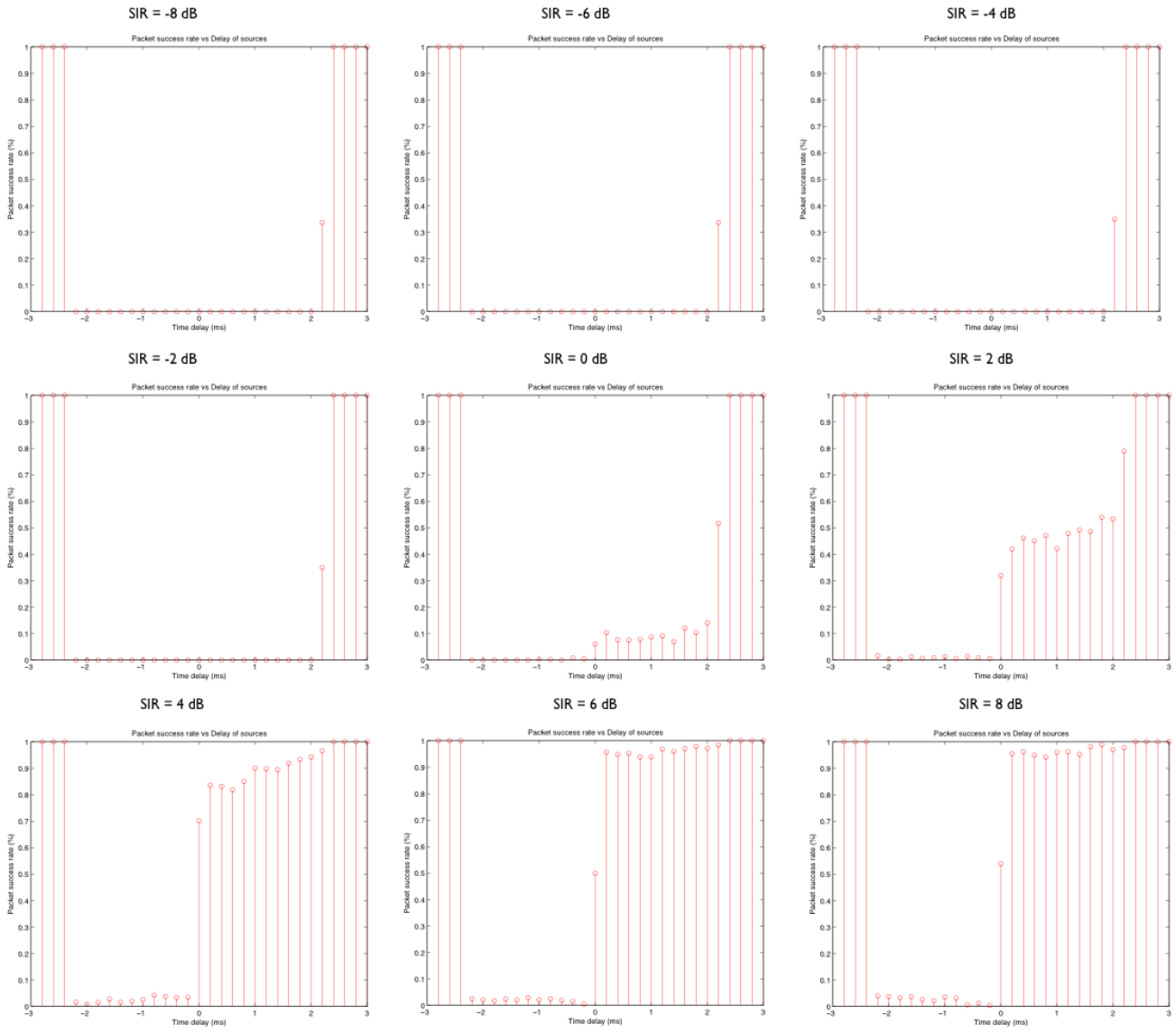


Figure A.3: CMU Wireless Emulator Interference Results for 5.5 Mbps

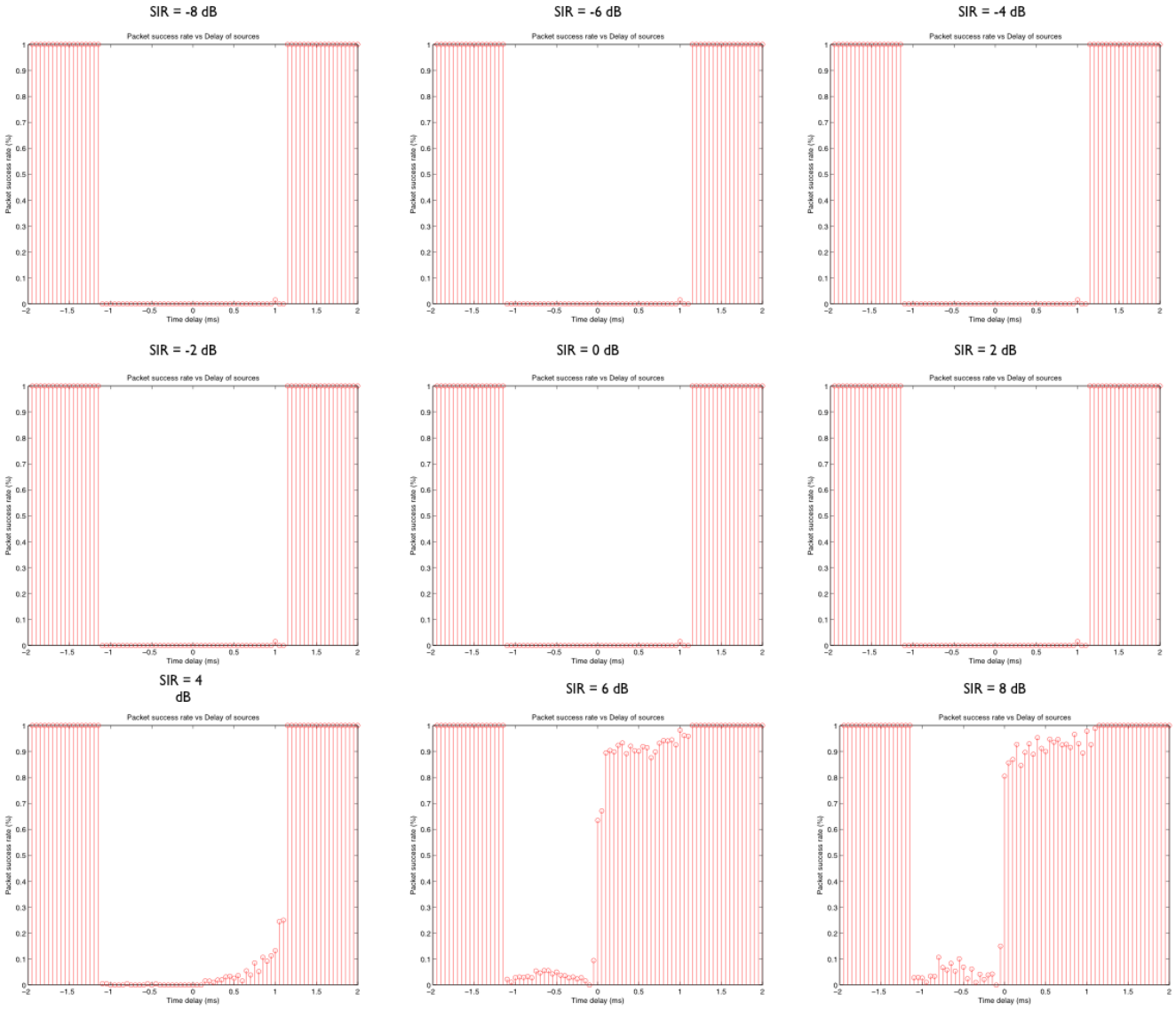


Figure A.4: CMU Wireless Emulator Interference Results for 11 Mbps

SIR (dB) — Payload Interference Lookup Table: Packet Success Rate 1Mbps

Interference Offset Delay (milliseconds)

	8	6	4	2	0	−2	−4	−6	−8
0	0.9152	0.8837	0.7801	0.6909	0.5325	0.5173	0.2538	0.1346	0
0.4	0.8815	0.9341	0.8278	0.6836	0.9080	0.5216	0.2994	0.1529	0
0.8	0.8976	0.9375	0.8617	0.6718	0.9215	0.5616	0.3060	0.1639	0
1.2	0.9021	0.9385	0.8592	0.7054	0.9014	0.5912	0.2976	0.1831	0
1.6	0.9712	0.9430	0.9025	0.7133	0.9061	0.6541	0.3424	0.2103	0
2	0.9034	0.9026	0.9185	0.7130	0.9120	0.6105	0.3827	0.2119	0
2.4	0.9275	0.9415	0.9072	0.7405	0.9372	0.6489	0.4092	0.2227	0
2.8	0.9091	0.9064	0.9171	0.8032	0.9540	0.7030	0.4783	0.2137	0.0071
3.2	0.9254	0.9385	0.8921	0.7613	0.9662	0.6729	0.4109	0.1950	0
3.6	0.9550	0.9935	0.8763	0.8078	0.9438	0.6690	0.4177	0.2217	0
4	0.9402	0.9423	0.8823	0.7981	0.9451	0.6581	0.4185	0.2568	0
4.4	0.9542	0.9355	0.8879	0.7957	0.9498	0.6962	0.4457	0.2944	0
4.8	0.9452	0.9704	0.9104	0.8296	0.9577	0.6779	0.4879	0.2938	0
5.2	0.9863	0.9503	0.9161	0.8352	0.9555	0.6943	0.4989	0.3167	0
5.6	0.9350	0.9750	0.9132	0.7815	0.9509	0.6938	0.5432	0.3220	0
6	0.9184	0.9487	0.9198	0.8178	0.9761	0.7239	0.5492	0.3288	0
6.4	0.9556	0.9521	0.9227	0.8487	0.9715	0.7069	0.5225	0.3658	0
6.8	0.9771	0.9653	0.9376	0.8310	0.9809	0.6954	0.5891	0.3830	0
7.2	0.9710	0.9877	0.9442	0.8036	0.9659	0.7451	0.5974	0.3617	0
7.6	0.9441	0.9441	0.9388	0.9062	0.9838	0.8269	0.6663	0.4105	0
8	0.9255	0.9505	0.9417	0.8784	0.9826	0.8375	0.6893	0.4468	0
8.4	0.9400	0.9545	0.9407	0.8986	0.9901	0.8200	0.6826	0.4886	0
8.8	0.9490	0.9573	0.9449	0.8800	0.9821	0.8552	0.7146	0.5113	0
9.2	0.9408	0.9581	0.9438	0.9205	0.9762	0.8076	0.6871	0.5255	0
9.6	0.9784	0.9708	0.9464	0.9154	0.9752	0.8168	0.7117	0.5498	0
10	0.9716	0.9744	0.9599	0.9203	0.9787	0.8375	0.7029	0.5599	0
10.4	1	0.9935	0.9651	0.9185	0.9683	0.8367	0.7316	0.5729	0
10.8	0.9923	0.9877	0.9664	0.9097	0.9753	0.8517	0.7345	0.5830	0
11.2	0.9781	0.9880	0.9514	0.9200	0.9793	0.8577	0.7647	0.6095	0
11.6	0.9915	0.9940	0.9681	0.9288	0.9688	0.8553	0.7978	0.5924	0.0108
12	0.9923	0.9724	0.9619	0.9173	0.9805	0.8814	0.8145	0.7087	0

Figure A.5: 1Mbps lookup table for payload interference.

SIR (dB)

Payload Interference Lookup Table: Packet Success Rate 2Mbps

Interference Offset Delay (milliseconds)

	8	6	4	2	0	−2	−4	−6	−8
0	0.9715	0.1030	0.1566	0.1486	0.0956	0	0	0	0
0.2	0.9630	0.8427	0.8286	0.7727	0.7594	0.5926	0.4561	0.3234	0.2185
0.4	0.9545	0.8435	0.8148	0.6923	0.7398	0.5841	0.3941	0.3122	0.2721
0.6	0.9480	0.8575	0.7172	0.6645	0.6119	0.6244	0.3936	0.3317	0.2385
0.8	0.9532	0.8430	0.8161	0.7163	0.7023	0.6035	0.3560	0.3410	0.2429
1	0.9814	0.8675	0.8036	0.7222	0.6484	0.6349	0.4044	0.3911	0.2522
1.2	0.9796	0.8582	0.8817	0.7949	0.6750	0.6094	0.4396	0.3938	0.2541
1.4	0.9818	0.8699	0.8211	0.8000	0.8175	0.5816	0.4483	0.4845	0.2608
1.6	0.9767	0.8848	0.8596	0.8405	0.7660	0.5898	0.5238	0.5510	0.3106
1.8	0.9798	0.8786	0.8586	0.8489	0.8143	0.6565	0.4462	0.4455	0.3064
2	0.9673	0.8846	0.7684	0.8562	0.8145	0.6518	0.4629	0.4860	0.2892
2.2	0.9682	0.9133	0.8351	0.8693	0.8028	0.6819	0.5596	0.5082	0.3264
2.4	0.9813	0.8705	0.8485	0.8101	0.8222	0.7326	0.5707	0.4851	0.4864
2.6	0.9901	0.8809	0.8692	0.8380	0.8207	0.7146	0.5879	0.5430	0.5274
2.8	0.9908	0.8631	0.8378	0.8302	0.7970	0.7342	0.6810	0.5083	0.5402
3	0.9954	0.9122	0.8922	0.8874	0.8298	0.7426	0.6237	0.5700	0.4600
3.2	0.9856	0.8976	0.8280	0.8645	0.7660	0.7500	0.5882	0.5450	0.5039
3.4	0.9868	0.9189	0.8641	0.8630	0.8130	0.7340	0.6995	0.5538	0.5379
3.6	0.9774	0.8822	0.8870	0.9214	0.7652	0.7786	0.6837	0.5026	0.4955
3.8	0.9685	0.9171	0.9063	0.8514	0.8239	0.7182	0.6284	0.5860	0.5268
4	0.9779	0.8869	0.9388	0.8551	0.8492	0.7380	0.7047	0.4492	0.4961
4.2	0.9740	0.8940	0.8952	0.8535	0.8538	0.7629	0.6648	0.5860	0.4727
4.4	0.9913	0.8520	0.8991	0.8414	0.8493	0.7646	0.6308	0.5147	0.5164
4.6	0.9836	0.9062	0.8980	0.8521	0.8056	0.8111	0.6720	0.5568	0.4665
4.8	0.9953	0.9063	0.9468	0.8750	0.8385	0.7435	0.6495	0.5025	0.5034
5	0.9693	0.8996	0.8764	0.8466	0.8333	0.8005	0.6804	0.5533	0.5167
5.2	0.9740	0.9173	0.9083	0.8154	0.8527	0.8060	0.7232	0.5489	0.4775
5.4	0.9856	0.8979	0.8932	0.8819	0.9044	0.8074	0.7283	0.5285	0.5388
5.6	0.9897	0.9111	0.8200	0.8353	0.8527	0.8282	0.7486	0.5248	0.5270
5.8	0.9911	0.9202	0.8980	0.8478	0.8414	0.8128	0.7287	0.5261	0.5011
6	0.9897	0.9214	0.8889	0.8562	0.7941	0.8290	0.8021	0.6526	0.5209

Figure A.6: 2Mbps lookup table for payload interference.

Payload Interference Lookup Table: Packet Success Rate 5.5Mbps

Interference Offset Delay (milliseconds) \ SIR (dB)	8	6	4	2	0	−2	−4	−6	−8
0	0	0	0.6649	0.3333	0.0745	0	0	0	0
0.1	0.9837	0.9099	0.7717	0.3736	0.0654	0	0	0	0
0.2	0.9640	0.9533	0.8526	0.4930	0.0968	0	0	0	0
0.3	0.9429	0.9556	0.8490	0.3556	0.0737	0	0	0	0
0.4	0.9706	0.9466	0.8343	0.3944	0.0460	0	0	0	0
0.5	0.9487	0.9348	0.8144	0.4396	0.1013	0	0	0	0
0.6	0.9505	0.9505	0.8211	0.4032	0.0549	0	0	0	0
0.7	0.9735	0.9441	0.8444	0.5556	0.0986	0	0	0	0
0.8	0.9619	0.9464	0.8703	0.4231	0.0685	0	0	0	0
0.9	0.9464	0.9366	0.8670	0.5128	0.0575	0	0	0	0
1	0.9516	0.9307	0.8920	0.4306	0.0506	0	0	0	0
1.1	0.9709	0.9270	0.9029	0.4026	0.1505	0	0	0	0
1.2	0.9600	0.9590	0.9172	0.5205	0.1034	0	0	0	0
1.3	0.9587	0.9638	0.8808	0.4658	0.0779	0	0	0	0
1.4	0.9316	0.9481	0.8882	0.4819	0.0854	0	0	0	0
1.5	0.9694	0.9662	0.8793	0.5333	0.0256	0	0	0	0
1.6	0.9831	0.9674	0.9207	0.4177	0.1000	0	0	0	0
1.7	0.9646	0.9738	0.9061	0.5484	0.1139	0	0	0	0
1.8	0.9922	0.9847	0.9195	0.4237	0.1481	0	0	0	0
1.9	0.9823	0.9697	0.9329	0.5698	0.0864	0	0	0	0
2	0.9694	0.9699	0.9444	0.4933	0.1268	0	0	0	0
2.1	0.9739	0.9693	0.9529	0.5556	0.1351	0	0	0	0

Figure A.7: 5.5Mbps lookup table for payload interference.

Payload Interference Lookup Table: Packet Success Rate 11Mbps

Interference Offset Delay (milliseconds) \ SIR (dB)	8	6	4	2	0	−2	−4	−6	−8
0	0.8046	0.6343	0	0	0	0	0	0	0
0.05	0.8687	0.8944	0	0	0	0	0	0	0
0.1	0.8462	0.8984	0.0153	0	0	0	0	0	0
0.15	0.9294	0.9328	0.0193	0	0	0	0	0	0
0.2	0.9529	0.9203	0.0314	0	0	0	0	0	0
0.25	0.9000	0.9005	0.0267	0	0	0	0	0	0
0.3	0.9355	0.9146	0.0160	0	0	0	0	0	0
0.35	0.9250	0.8985	0.0387	0	0	0	0	0	0
0.4	0.9157	0.9420	0.0521	0	0	0	0	0	0
0.45	0.9302	0.9440	0.0919	0	0	0	0	0	0
0.5	0.9783	0.9822	0.1329	0	0	0	0	0	0
0.55	0.9886	0.9588	0.2500	0	0	0	0	0	0
0.6	0.9773	0.9641	0.9138	0	0	0	0	0	0
0.65	0.9677	0.9623	0.9767	0	0	0	0	0	0
0.7	0.9494	0.9449	0.9578	0	0	0	0	0	0
0.75	1	0.9509	0.9594	0	0	0	0	0	0
0.8	0.9579	0.9388	0.9719	0	0	0	0	0	0
0.85	0.9875	0.9504	0.9659	0	0	0	0	0	0
0.9	0.9583	0.9518	0.9688	0	0	0	0	0	0
0.95	0.9773	0.9525	0.9759	0	0	0	0	0	0
1	0.9691	0.9572	0.9474	0	0	0	0	0	0
1.05	0.9506	0.9761	0.9595	0	0	0	0	0	0
1.1	1	1	1	0	0	0	0	0	0

Figure A.8: 11Mbps lookup table for payload interference.

Appendix B

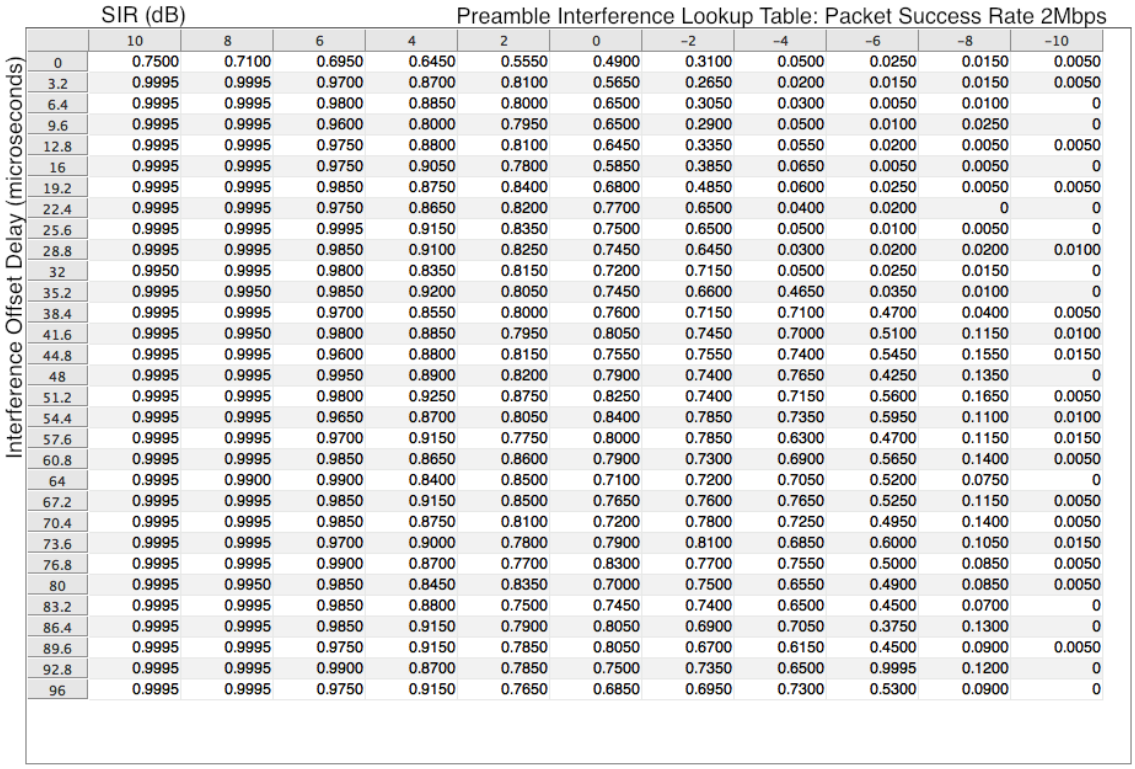
PREAMBLE INTERFERENCE LOOKUP TABLE

The lookup tables derived from the raw data in the preamble interference tests from [9] are displayed here. These results are recommended to be implemented in the ns-3 Network Simulator as a lookup table in the case of single source interference, when interference occurs during a packet preamble.

Preamble Interference Lookup Table: Packet Success Rate 1Mbps

Interference Offset Delay (microseconds) \ SIR (dB)	10	8	6	4	2	0	−2	−4	−6	−8	−10
0	0.8750	0.8850	0.8700	0.8800	0.8150	0.6700	0.3200	0.0550	0.0400	0.0350	0.0300
3.2	0.9995	0.9995	0.9995	0.9700	0.9600	0.8100	0.3600	0.0400	0.0500	0.0150	0.0150
6.4	0.9995	0.9995	0.9995	0.9800	0.9400	0.8050	0.3350	0.0250	0.0400	0.0050	0.0050
9.6	0.9995	0.9995	0.9995	0.9950	0.9400	0.8450	0.3500	0.0350	0.0100	0.0350	0.0050
12.8	0.9995	0.9995	0.9995	0.9700	0.9250	0.7750	0.3150	0.0150	0.0300	0.0300	0.0400
16	0.9995	0.9995	0.9950	0.9995	0.9400	0.7850	0.3400	0.0500	0.0150	0.0050	0.0100
19.2	0.9995	0.9995	0.9995	0.9950	0.9400	0.8400	0.3750	0.0050	0.0500	0.0100	0.0050
22.4	0.9995	0.9995	0.9995	0.9750	0.9700	0.9250	0.5450	0.0500	0.0100	0.0250	0.0150
25.6	0.9995	0.9995	0.9995	0.9950	0.9450	0.8650	0.4500	0.0400	0.0050	0.0200	0.0100
28.8	0.9995	0.9995	0.9950	0.9900	0.9650	0.9350	0.5350	0.0250	0.0100	0	0.0350
32	0.9995	0.9995	0.9995	0.9700	0.9400	0.9250	0.6250	0.0250	0.0150	0.0100	0.0050
35.2	0.9995	0.9995	0.9995	0.9900	0.9850	0.9300	0.6450	0.3100	0.0300	0.0250	0.0350
38.4	0.9995	0.9995	0.9995	0.9950	0.9600	0.9200	0.6700	0.6650	0.4150	0.0700	0.0250
41.6	0.9995	0.9995	0.9995	0.9900	0.9600	0.9100	0.6450	0.6400	0.5250	0.0350	0.0100
44.8	0.9995	0.9995	0.9995	0.9950	0.9550	0.9250	0.5950	0.6500	0.4450	0.0750	0.0050
48	0.9995	0.9995	0.9950	0.9800	0.9700	0.8850	0.6150	0.6450	0.4800	0.1050	0.0050
51.2	0.9995	0.9995	0.9995	0.9750	0.9450	0.9150	0.6300	0.6450	0.5550	0.0750	0.0100
54.4	0.9995	0.9995	0.9995	0.9700	0.9800	0.8950	0.7300	0.6900	0.5350	0.0600	0.0100
57.6	0.9995	0.9995	0.9995	0.9950	0.9400	0.8850	0.6450	0.6700	0.4500	0.0550	0.0050
60.8	0.8750	0.9995	0.9950	0.9750	0.9700	0.9300	0.6650	0.6750	0.4550	0.0950	0.0250
64	0.9995	0.9995	0.9950	0.9800	0.9750	0.9150	0.6700	0.6450	0.4400	0.1050	0.0150
67.2	0.9995	0.9995	0.9995	0.9900	0.9350	0.9200	0.6400	0.5900	0.4000	0.0900	0.0200
70.4	0.9995	0.9995	0.9995	0.9800	0.9600	0.9200	0.6450	0.5950	0.4000	0.1050	0.0200
73.6	0.9995	0.9995	0.9995	0.9800	0.9750	0.9500	0.6750	0.7400	0.4350	0.1150	0.0050
76.8	0.9995	0.9995	0.9995	0.9650	0.9650	0.9400	0.7000	0.6950	0.4350	0.0800	0.0550
80	0.9995	0.9995	0.9950	0.9900	0.9750	0.9150	0.5950	0.6700	0.3650	0.0600	0.0250
83.2	0.9995	0.9995	0.9995	0.9950	0.9750	0.9300	0.6700	0.7050	0.4350	0.0550	0.0150
86.4	0.9995	0.9995	0.9995	0.9850	0.9550	0.8950	0.6800	0.6450	0.4200	0.0350	0.0100
89.6	0.9995	0.9995	0.9995	0.9700	0.9650	0.9350	0.6800	0.5950	0.4250	0.0700	0.0150
92.8	0.9995	0.9995	0.9950	0.9850	0.9850	0.9100	0.6550	0.6950	0.4700	0.0350	0.0150
96	0.9995	0.9995	0.9995	0.9950	0.9900	0.9250	0.6250	0.6600	0.5950	0.0650	0.0300

Figure B.1: 1Mbps lookup table for preamble interference. Data courtesy of [9].

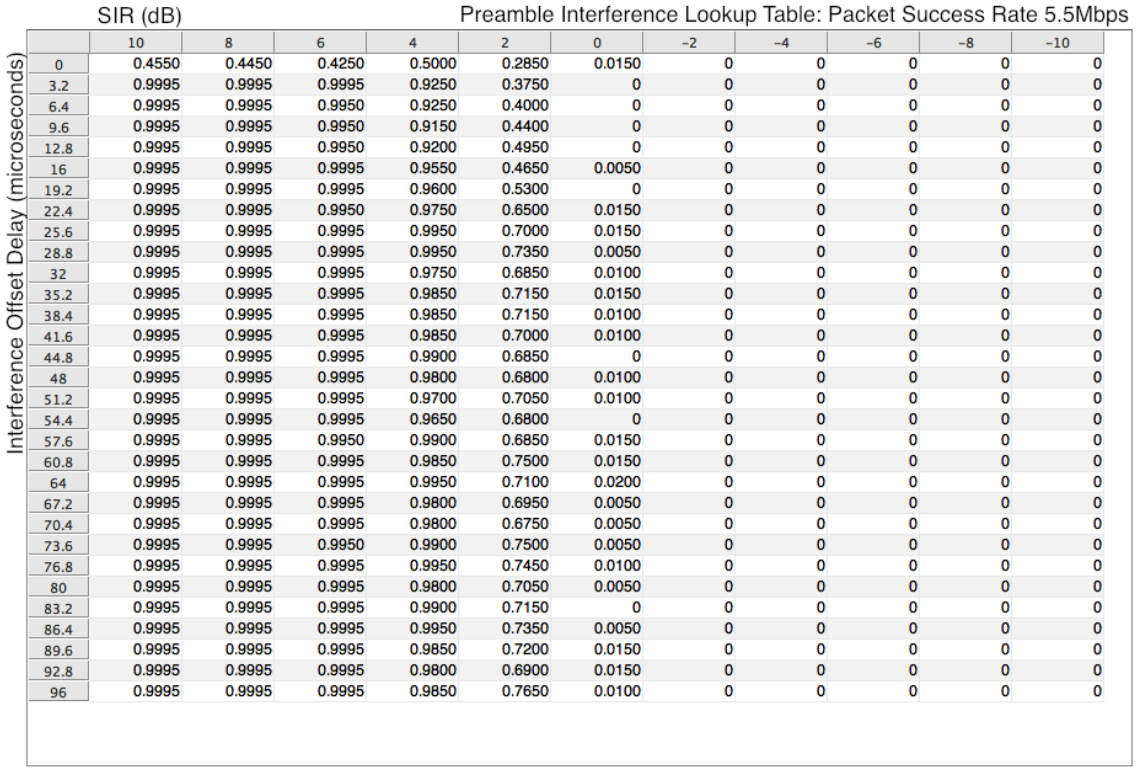


SIR (dB) — Preamble Interference Lookup Table: Packet Success Rate 2Mbps

Interference Offset Delay (microseconds)

	10	8	6	4	2	0	−2	−4	−6	−8	−10
0	0.7500	0.7100	0.6950	0.6450	0.5550	0.4900	0.3100	0.0500	0.0250	0.0150	0.0050
3.2	0.9995	0.9995	0.9700	0.8700	0.8100	0.5650	0.2650	0.0200	0.0150	0.0150	0.0050
6.4	0.9995	0.9995	0.9800	0.8850	0.8000	0.6500	0.3050	0.0300	0.0050	0.0100	0
9.6	0.9995	0.9995	0.9600	0.8000	0.7950	0.6500	0.2900	0.0500	0.0100	0.0250	0
12.8	0.9995	0.9995	0.9750	0.8800	0.8100	0.6450	0.3350	0.0550	0.0200	0.0050	0.0050
16	0.9995	0.9995	0.9750	0.9050	0.7800	0.5850	0.3850	0.0650	0.0050	0.0050	0
19.2	0.9995	0.9995	0.9850	0.8750	0.8400	0.6800	0.4850	0.0600	0.0250	0.0050	0.0050
22.4	0.9995	0.9995	0.9750	0.8650	0.8200	0.7700	0.6500	0.0400	0.0200	0	0
25.6	0.9995	0.9995	0.9995	0.9150	0.8350	0.7500	0.6500	0.0500	0.0100	0.0050	0
28.8	0.9995	0.9995	0.9850	0.9100	0.8250	0.7450	0.6450	0.0300	0.0200	0.0200	0.0100
32	0.9950	0.9995	0.9800	0.8350	0.8150	0.7200	0.7150	0.0500	0.0250	0.0150	0
35.2	0.9995	0.9950	0.9850	0.9200	0.8050	0.7450	0.6600	0.4650	0.0350	0.0100	0
38.4	0.9995	0.9995	0.9700	0.8550	0.8000	0.7600	0.7150	0.7100	0.4700	0.0400	0.0050
41.6	0.9995	0.9950	0.9800	0.8850	0.7950	0.8050	0.7450	0.7000	0.5100	0.1150	0.0100
44.8	0.9995	0.9995	0.9600	0.8800	0.8150	0.7550	0.7550	0.7400	0.5450	0.1550	0.0150
48	0.9995	0.9995	0.9950	0.8900	0.8200	0.7900	0.7400	0.7650	0.4250	0.1350	0
51.2	0.9995	0.9995	0.9800	0.9250	0.8750	0.8250	0.7400	0.7150	0.5600	0.1650	0.0050
54.4	0.9995	0.9995	0.9650	0.8700	0.8050	0.8400	0.7850	0.7350	0.5950	0.1100	0.0100
57.6	0.9995	0.9995	0.9700	0.9150	0.7750	0.8000	0.7850	0.6300	0.4700	0.1150	0.0150
60.8	0.9995	0.9995	0.9850	0.8650	0.8600	0.7900	0.7300	0.6900	0.5650	0.1400	0.0050
64	0.9995	0.9900	0.9900	0.8400	0.8500	0.7100	0.7200	0.7050	0.5200	0.0750	0
67.2	0.9995	0.9995	0.9850	0.9150	0.8500	0.7650	0.7600	0.7650	0.5250	0.1150	0.0050
70.4	0.9995	0.9995	0.9850	0.8750	0.8100	0.7200	0.7800	0.7250	0.4950	0.1400	0.0050
73.6	0.9995	0.9995	0.9700	0.9000	0.7800	0.7900	0.8100	0.6850	0.6000	0.1050	0.0150
76.8	0.9995	0.9995	0.9900	0.8700	0.7700	0.8300	0.7700	0.7550	0.5000	0.0850	0.0050
80	0.9995	0.9950	0.9850	0.8450	0.8350	0.7000	0.7500	0.6550	0.4900	0.0850	0.0050
83.2	0.9995	0.9995	0.9850	0.8800	0.7500	0.7450	0.7400	0.6500	0.4500	0.0700	0
86.4	0.9995	0.9995	0.9850	0.9150	0.7900	0.8050	0.6900	0.7050	0.3750	0.1300	0
89.6	0.9995	0.9995	0.9750	0.9150	0.7850	0.8050	0.6700	0.6150	0.4500	0.0900	0.0050
92.8	0.9995	0.9995	0.9900	0.8700	0.7850	0.7500	0.7350	0.6500	0.9995	0.1200	0
96	0.9995	0.9995	0.9750	0.9150	0.7650	0.6850	0.6950	0.7300	0.5300	0.0900	0

Figure B.2: 2Mbps lookup table for preamble interference. Data courtesy of [9].



SIR (dB) — Preamble Interference Lookup Table: Packet Success Rate 5.5Mbps

Interference Offset Delay (microseconds)

	10	8	6	4	2	0	−2	−4	−6	−8	−10
0	0.4550	0.4450	0.4250	0.5000	0.2850	0.0150	0	0	0	0	0
3.2	0.9995	0.9995	0.9995	0.9250	0.3750	0	0	0	0	0	0
6.4	0.9995	0.9995	0.9950	0.9250	0.4000	0	0	0	0	0	0
9.6	0.9995	0.9995	0.9950	0.9150	0.4400	0	0	0	0	0	0
12.8	0.9995	0.9995	0.9950	0.9200	0.4950	0	0	0	0	0	0
16	0.9995	0.9995	0.9995	0.9550	0.4650	0.0050	0	0	0	0	0
19.2	0.9995	0.9995	0.9995	0.9600	0.5300	0	0	0	0	0	0
22.4	0.9995	0.9995	0.9950	0.9750	0.6500	0.0150	0	0	0	0	0
25.6	0.9995	0.9995	0.9995	0.9950	0.7000	0.0150	0	0	0	0	0
28.8	0.9995	0.9995	0.9995	0.9950	0.7350	0.0050	0	0	0	0	0
32	0.9995	0.9995	0.9995	0.9750	0.6850	0.0100	0	0	0	0	0
35.2	0.9995	0.9995	0.9995	0.9850	0.7150	0.0150	0	0	0	0	0
38.4	0.9995	0.9995	0.9995	0.9850	0.7150	0.0100	0	0	0	0	0
41.6	0.9995	0.9995	0.9995	0.9850	0.7000	0.0100	0	0	0	0	0
44.8	0.9995	0.9995	0.9995	0.9900	0.6850	0	0	0	0	0	0
48	0.9995	0.9995	0.9995	0.9800	0.6800	0.0100	0	0	0	0	0
51.2	0.9995	0.9995	0.9995	0.9700	0.7050	0.0100	0	0	0	0	0
54.4	0.9995	0.9995	0.9995	0.9650	0.6800	0	0	0	0	0	0
57.6	0.9995	0.9995	0.9950	0.9900	0.6850	0.0150	0	0	0	0	0
60.8	0.9995	0.9995	0.9995	0.9850	0.7500	0.0150	0	0	0	0	0
64	0.9995	0.9995	0.9995	0.9950	0.7100	0.0200	0	0	0	0	0
67.2	0.9995	0.9995	0.9995	0.9800	0.6950	0.0050	0	0	0	0	0
70.4	0.9995	0.9995	0.9995	0.9800	0.6750	0.0050	0	0	0	0	0
73.6	0.9995	0.9995	0.9950	0.9900	0.7500	0.0050	0	0	0	0	0
76.8	0.9995	0.9995	0.9995	0.9950	0.7450	0.0100	0	0	0	0	0
80	0.9995	0.9995	0.9995	0.9800	0.7050	0.0050	0	0	0	0	0
83.2	0.9995	0.9995	0.9995	0.9900	0.7150	0	0	0	0	0	0
86.4	0.9995	0.9995	0.9995	0.9950	0.7350	0.0050	0	0	0	0	0
89.6	0.9995	0.9995	0.9995	0.9850	0.7200	0.0150	0	0	0	0	0
92.8	0.9995	0.9995	0.9995	0.9800	0.6900	0.0150	0	0	0	0	0
96	0.9995	0.9995	0.9995	0.9850	0.7650	0.0100	0	0	0	0	0

Figure B.3: 5.5Mbps lookup table for preamble interference. Data courtesy of [9].

SIR (dB) — Preamble Interference Lookup Table: Packet Success Rate 11Mbps

Interference Offset Delay (microseconds)	10	8	6	4	2	0	−2	−4	−6	−8	−10
0	0.2650	0.2650	0.1950	0.0350	0	0	0	0	0	0	0
3.2	0.9950	0.9800	0.7300	0.1200	0	0	0	0	0	0	0
6.4	0.9950	0.9650	0.8050	0.1100	0	0	0	0	0	0	0
9.6	0.9800	0.9600	0.7600	0.1100	0.0050	0	0	0	0	0	0
12.8	0.9900	0.9750	0.7650	0.1600	0	0	0	0	0	0	0
16	0.9900	0.9800	0.8200	0.1700	0	0	0	0	0	0	0
19.2	0.9900	0.9900	0.8900	0.2250	0	0	0	0	0	0	0
22.4	0.9995	0.9995	0.9050	0.3100	0	0	0	0	0	0	0
25.6	0.9995	0.9995	0.9650	0.2500	0	0	0	0	0	0	0
28.8	0.9950	0.9995	0.9500	0.3100	0	0	0	0	0	0	0
32	0.9995	0.9900	0.9400	0.3050	0	0	0	0	0	0	0
35.2	0.9995	0.9950	0.9250	0.2500	0	0	0	0	0	0	0
38.4	0.9950	0.9995	0.9400	0.2750	0	0	0	0	0	0	0
41.6	0.9950	0.9950	0.9450	0.3100	0	0	0	0	0	0	0
44.8	0.9995	0.9995	0.9600	0.3300	0	0	0	0	0	0	0
48	0.9995	0.9950	0.9500	0.3050	0	0	0	0	0	0	0
51.2	0.9995	0.9995	0.9450	0.2900	0	0	0	0	0	0	0
54.4	0.9995	0.9950	0.9700	0.3200	0	0	0	0	0	0	0
57.6	0.9995	0.9995	0.9350	0.2950	0	0	0	0	0	0	0
60.8	0.9995	0.9950	0.9450	0.3000	0	0	0	0	0	0	0
64	0.9995	0.9950	0.9400	0.3000	0	0	0	0	0	0	0
67.2	0.9995	0.9950	0.9450	0.3200	0	0	0	0	0	0	0
70.4	0.9950	0.9995	0.9600	0.2700	0	0	0	0	0	0	0
73.6	0.9995	0.9950	0.9450	0.3100	0	0	0	0	0	0	0
76.8	0.9995	0.9995	0.9500	0.3750	0	0	0	0	0	0	0
80	0.9995	0.9950	0.9350	0.3250	0	0	0	0	0	0	0
83.2	0.9995	0.9995	0.9700	0.3100	0	0	0	0	0	0	0
86.4	0.9995	0.9950	0.9350	0.2700	0	0	0	0	0	0	0
89.6	0.9950	0.9995	0.8950	0.2700	0	0	0	0	0	0	0
92.8	0.9995	0.9950	0.9750	0.2800	0	0	0	0	0	0	0
96	0.9995	0.9995	0.9300	0.2750	0	0	0	0	0	0	0

Figure B.4: 11Mbps lookup table for preamble interference. Data courtesy of [9].