Reduction in Coexistent WLAN Interference Through Statistical Traffic Management

Pablo Maximiliano Robert

Dissertation submitted to the Faculty of the Virginia Polytechnic Institute and State University in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Electrical Engineering

Jeffrey H. Reed, Chair
William H. Tranter
Scott F. Midkiff
Luiz A. DaSilva
C. Patrick Koelling

April 17, 2003
Blacksburg, Virginia

Keywords: WLAN, WPAN, Traffic, Interference, Coexistence, Middleware, IEEE 802.11b, Bluetooth, IEEE 802.15

Copyright 2003, Pablo Maximiliano Robert
Reduction in Coexistent WLAN Interference Through Statistical Traffic Management

Pablo Maximiliano Robert

(ABSTRACT)

In recent years, an increasing number of devices have been developed for operation in the bands allocated by the Federal Communications Commission (FCC) for license-free operation. Given the rules governing devices in these bands, it is possible for interference created by these devices to significantly reduce the overall capacity of these bands. Two such protocols are Bluetooth and IEEE 802.11b. Several methods have been presented in the literature for managing interference between these two devices. However, these approaches are generally not practical, since they either require the purchase of specialized hardware or do not comply with the current versions of existing protocols. In this dissertation, an approach is presented that is not only backwards-compatible, but requires the algorithm to be implemented in only a small subset of the devices operating in the local environment for the coexistence algorithm to function properly. An analytical solution for this coexistence approach when applied to generic networks is presented. A method is also presented for the backwards-compatible integration of some medium access control (MAC) protocols into Bluetooth devices. A case study of the Bluetooth/IEEE 802.11b coexistence problem is presented in this dissertation, as well as a proposed coexistence mechanism, collision-based multiple access (CBMA). A form of adaptive frequency hopping (AFH) is presented in this dissertation, as well as a combined CBMA/AFH strategy. The CBMA algorithm is shown be able to significantly reduce the impact of a Bluetooth link on an IEEE 802.11b link. The AFH algorithm is shown to have comparable performance to the CBMA algorithm. A combined CBMA/AFH algorithm presented, is shown to not only have an impact on the IEEE 802.11b link that is not greater than the CBMA-only implementation, but the Bluetooth link throughput is shown to be significantly greater than either the CBMA or AFH implementation alone.
This work is dedicated to the Comicbook Store Guy and Professor Frink for showing me how to find the road less traveled and to Margaret Cline for making the search worthwhile.
I would like to acknowledge the many people who have made this dissertation possible. First, I would like to thank my advisor, Jeffrey H. Reed. Both his technical guidance and his technical editing were invaluable contributions to this dissertation. Luiz A. DaSilva’s contribution to modeling the network traffic and the analytical framework were also invaluable. I would like to thank William H. Tranter for his insight into the simulation of wireless networks. Next, I would like to thank Scott F. Midkiff and C. Patrick Koelling for their support of this research. I would like to thank America Online, and especially Kris Gabor, Michaela Barnes, and Amy Hale, for making this dissertation possible. I would also like to thank my parents and siblings for being there for me over the years. Finally, I would like to thank Matthew Valenti, Carl Fossa, Roberto Conte, Raqibul Mostafa, James Hicks, Srikathyayani Srikantheswara, Jody Neel, Michael Cating, and Theresa Roeder for their friendship and support over the last several years.

This work has been funded by the America Online Wireless Home Networking Technologies Fellowship, the Bradley Fellowship, and the MPRG Affiliates Program.
## Contents

1 Introduction ........................................ 1

2 Background ........................................ 4

   2.1 Wireless Communications .......................... 4

         2.1.1 FCC Part 15 Rules .......................... 6

   2.2 Protocols ........................................ 7

         2.2.1 Bluetooth .................................... 8

         2.2.2 IEEE 802.11b ................................. 16

   2.3 Previous Work .................................... 26

         2.3.1 Backoff/traffic conditioning mechanisms ....... 27

         2.3.2 WLAN interference mitigation ................... 29

         2.3.3 IEEE 802.15 Part 2 .......................... 35

   2.4 Models ........................................... 36

         2.4.1 Environment .................................. 37

         2.4.2 Radio Channel ................................. 38

         2.4.3 Data Traffic .................................. 43
2.5 Summary ................................................................. 50

3 Coexistence Analysis and Mitigation .............................. 52

3.1 Network Description .................................................. 52

3.1.1 Statistical Description of Network Traffic .................... 53

3.1.2 Statistical Description of Network Traffic Example ............ 55

3.1.3 Channel Statistical State ......................................... 56

3.1.4 Channel Statistical State Example .............................. 59

3.2 Coexistence Issue .................................................... 61

3.2.1 Collision Awareness .............................................. 62

3.2.2 Collision Awareness Example ................................... 65

3.3 Summary .............................................................. 67

4 Coexistence Mechanism ................................................. 68

4.1 Collision Avoidance .................................................. 68

4.1.1 Case 1 - Short Packets with Maximum Throughput ............ 70

4.1.2 Case 2 - Long Packets with Maximum Throughput ............ 71

4.1.3 Case 3 - Long Packets with Variable Data Traffic Rate ....... 74

4.2 Discussion ............................................................ 75

4.3 Summary .............................................................. 76

5 Integration .............................................................. 77

5.1 Transparency .......................................................... 78
6 Practical Case Study

6.1 Introduction ........................................ 87
6.2 Approach ............................................... 88
   6.2.1 Insertion Interface ............................. 89
   6.2.2 Susceptibility to Errors ......................... 90
   6.2.3 Use Paradigm .................................... 94
6.3 Case Study Scenario ................................ 95
   6.3.1 Scaling .......................................... 98
   6.3.2 Interference Solution ............................ 106
6.4 Simulation ............................................ 108
   6.4.1 Device Models ................................... 108
   6.4.2 Metrics .......................................... 111
   6.4.3 Convergence .................................... 113
   6.4.4 Validation ...................................... 113
6.5 Algorithm Performance on IEEE 802.11b .......... 117
   6.5.1 Parameter Sensitivity ........................... 124
   6.5.2 Multiple Devices ................................. 137
6.6 Realistic Implementation ............................... 144
   6.6.1 Error Detection ................................. 144
6.6.2 Moving Scenario ........................................ 146
6.7 Algorithm Enhancements .................................. 152
  6.7.1 AFHH - Closed form .................................. 153
  6.7.2 AFH/CBMA Combination ............................... 155
6.8 Summary .................................................... 162

7 Conclusion .................................................. 163
  7.1 Future Work ............................................... 167
    7.1.1 IEEE 802.11 Implementation ........................ 167
    7.1.2 Rate-varying IEEE 802.11 ............................ 167
    7.1.3 Parameter Estimation ................................. 168
    7.1.4 Protocol Estimation ................................. 169
    7.1.5 Multi-source Behavior ............................... 170
    7.1.6 Channel Estimation ................................. 171
  7.2 Suggested Publications .................................. 171
List of Figures

2.1 Generic Network ................................................. 9
2.2 Bluetooth Piconet .................................................. 11
2.3 Piconet TDD scheme .............................................. 11
2.4 Piconet ARQ scheme ............................................... 13
2.5 Piconet ARQ scheme ............................................... 14
2.6 Independent Basic Service Set .................................. 17
2.7 Basic Service Set .................................................. 18
2.8 Extended Service Set .............................................. 18
2.9 IEEE 802.11 MAC Frame (MPDU) ............................... 20
2.10 IEEE 802.11 PPDU ............................................... 20
2.11 IEEE 802.11 RTS/CTS ........................................... 21
2.12 Detection of data prevents CTS reply to RTS reception (shaded ellipses indicate stations’ range) ........................................... 23
2.13 IEEE 802.11 Contention Mechanism ............................ 24
2.14 IEEE 802.11 Error Recovery ............................... 25
2.15 Fractal example .................................................. 48
3.1 Layout of network $N_1$ .................................................. 53
3.2 Packet traffic delivered over a single link ............................ 53
3.3 Probability of a new collision at time $k$ after the original collision assuming the start of the packet is known ......................... 59
3.4 Validation of analysis of probability that the channel is occupied at time $k$ after the beginning of a packet ........................................ 60
3.5 Layout of networks $N_1$ and $N_2$ ...................................... 61
3.6 Probability of collision between $N_1$ and $N_2$ at time $k$ after collision .... 67
4.1 Layout of network negotiation structure ................................. 69
4.2 Simulated throughput for case 1 ....................................... 71
4.3 Simulated efficiency for case 1 ........................................ 72
4.4 Simulated throughput for case 2 ....................................... 73
4.5 Simulated efficiency for case 2 ........................................ 73
4.6 Simulated throughput for case 3 ....................................... 74
4.7 Simulated efficiency for case 3 ........................................ 75
5.1 Comparison of DH1 vs. AUX1 packet link ............................ 79
5.2 Hybrid ARQ Algorithm .................................................... 82
5.3 Blind Adaptive ARQ applied to Bluetooth system compared with DH1 and DM1 packets (AWGN) ........................................ 85
6.1 Bluetooth/IEEE 802.11b Robustness to Noise ........................ 93
6.2 Basic TCP Principle ....................................................... 95
6.22 Impact of Different On Periods in IEEE 802.11b Source on Bluetooth Link Delay .................................................................................................................. 132
6.23 IEEE 802.11b Throughput for Traffic Source with Different Off Period .......... 134
6.24 Increased Queuing Delay Due to Shorter Off Period ................................ 134
6.25 Effect of different IEEE 802.11 source Off times on Bluetooth throughput . . 135
6.26 Effect of different IEEE 802.11 source Off times on Bluetooth delay .......... 136
6.27 Throughput of multiple Bluetooth devices when implementing CBMA Algo-

rithm in the Presence of IEEE 802.11b Device ............................................ 139
6.28 Delay of multiple Bluetooth devices when implementing CBMA Algorithm in the Presence of IEEE 802.11b Device ................................................. 140
6.29 IEEE 802.11b Throughput in the Presence of Multiple Bluetooth Devices . . 140
6.30 IEEE 802.11b Throughput in the Presence of Multiple Bluetooth Devices Not Implementing CBMA ................................................................. 141
6.31 Bluetooth Throughput for Multiple Bluetooth Devices in the Presence of IEEE 802.11b Link when Bluetooth Devices are Not Implementing CBMA . . 142
6.32 Bluetooth Delay for Multiple Bluetooth Devices in the Presence of IEEE 802.11b Link when Bluetooth Devices are Not Implementing CBMA . . 143
6.33 Throughput of IEEE 802.11b device in the presence of Bluetooth link when comparing different Bluetooth device separations ........................................ 147
6.34 Throughput of Bluetooth device in the presence of IEEE 802.11b link when comparing different Bluetooth device separations ................................. 149
6.35 Latency of Bluetooth device in the presence of IEEE 802.11b link when com-
paring different Bluetooth device separations .............................................. 150
6.36 Comparison of range of Bluetooth link when CBMA algorithm is implemented with when it is not implemented ........................................ 152
6.37 Frequency assignment of Master and Slave Bluetooth devices .................. 153
6.38 IEEE 802.11b throughput when Bluetooth implements combined AFH/CBMA 157
6.39 Bluetooth throughput when implementing combined AFH/CBMA in the presence of an IEEE 802.11b link ........................................ 158
6.40 Bluetooth delay when implementing combined AFH/CBMA in the presence of an IEEE 802.11b link ........................................ 159
6.41 IEEE 802.11b throughput when Bluetooth devices are implementing combined AFH/CBMA and the Bluetooth devices use an error trigger ........ 160
6.42 Bluetooth throughput when implementing combined AFH/CBMA using errors as a trigger ......................................................... 161
6.43 Bluetooth latency when implementing combined AFH/CBMA using errors as a trigger ......................................................... 161
List of Tables

2.1 Key Terms and Concepts ............................................. 5
2.2 Data rate associated with each Bluetooth packet type ............... 16
2.3 Suggested methods to mitigate interference between Bluetooth and IEEE 802.11b ................................................................. 30
2.4 IEEE 802.15 ................................................................. 35
4.1 Statistical-Backoff Algorithm ........................................... 69
5.1 Bluetooth Hybrid ARQ .................................................. 81
6.1 CBMA Implementation .................................................. 107
6.2 Throughput Performance with Overlapping Piconets .................... 138
6.3 AFH/CBMA implementation .......................................... 156
6.4 Comparison of CBMA, AFH, and AFH/CBMA over all simulated distances . 158
7.1 Dissertation Original Contributions ................................... 166
# Glossary

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACK</td>
<td>Acknowledgment</td>
</tr>
<tr>
<td>ACL</td>
<td>Asynchronous Connection-Less</td>
</tr>
<tr>
<td>AFH</td>
<td>Adaptive frequency hopping</td>
</tr>
<tr>
<td>ARQ</td>
<td>Automatic retransmission request</td>
</tr>
<tr>
<td>ATM</td>
<td>Asynchronous transfer mode</td>
</tr>
<tr>
<td>AUX1</td>
<td>Auxiliary packet</td>
</tr>
<tr>
<td>AWGN</td>
<td>Additive white gaussian noise</td>
</tr>
<tr>
<td>BSS</td>
<td>Basic service set</td>
</tr>
<tr>
<td>CBMA</td>
<td>Collision based multiple access</td>
</tr>
<tr>
<td>CRC</td>
<td>Cyclic redundancy code</td>
</tr>
<tr>
<td>CSMA/CA</td>
<td>Carrier sensing multiple access with collision avoidance</td>
</tr>
<tr>
<td>CTS</td>
<td>Clear to send</td>
</tr>
<tr>
<td>CW</td>
<td>Contention window (IEEE 802.11)</td>
</tr>
<tr>
<td>CW</td>
<td>Continuous wave (RF measurements)</td>
</tr>
<tr>
<td>DBPSK</td>
<td>Differential binary phase shift keying</td>
</tr>
<tr>
<td>DCF</td>
<td>Difference in center frequencies</td>
</tr>
<tr>
<td>DH1</td>
<td>Data packet, high speed, 1 slot</td>
</tr>
<tr>
<td>DH3</td>
<td>Data packet, high speed, 3 slot</td>
</tr>
<tr>
<td>DH5</td>
<td>Data packet, high speed, 5 slot</td>
</tr>
<tr>
<td>DIFS</td>
<td>Distributed (coordination function) interframe space</td>
</tr>
<tr>
<td>DM1</td>
<td>Data packet, medium speed, 1 slot</td>
</tr>
<tr>
<td>DM3</td>
<td>Data packet, medium speed, 3 slot</td>
</tr>
<tr>
<td>DM5</td>
<td>Data packet, medium speed, 5 slot</td>
</tr>
<tr>
<td>DQPSK</td>
<td>Differential quadrature phase shift keying</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>DS</td>
<td>Distribution system</td>
</tr>
<tr>
<td>DSS</td>
<td>Distribution system service</td>
</tr>
<tr>
<td>DSSS</td>
<td>Direct sequence spread spectrum</td>
</tr>
<tr>
<td>EIFS</td>
<td>Extended interframe space</td>
</tr>
<tr>
<td>ESS</td>
<td>Extended service set</td>
</tr>
<tr>
<td>FCC</td>
<td>Federal communications commission</td>
</tr>
<tr>
<td>FEC</td>
<td>Forward error correction</td>
</tr>
<tr>
<td>FER</td>
<td>Frame Error Rate</td>
</tr>
<tr>
<td>FHSS</td>
<td>Frequency hopping spread spectrum</td>
</tr>
<tr>
<td>FIFO</td>
<td>First in first out</td>
</tr>
<tr>
<td>FSK</td>
<td>Frequency shift keying</td>
</tr>
<tr>
<td>GFSK</td>
<td>Gaussian frequency shift keying</td>
</tr>
<tr>
<td>HCI</td>
<td>Host controller interface</td>
</tr>
<tr>
<td>IBSS</td>
<td>Independent basic service set</td>
</tr>
<tr>
<td>IEEE</td>
<td>Institute of Electrical and Electronic Engineers</td>
</tr>
<tr>
<td>IFS</td>
<td>Interframe space</td>
</tr>
<tr>
<td>IP</td>
<td>Internet protocol</td>
</tr>
<tr>
<td>ISM</td>
<td>Industrial, scientific, and medical</td>
</tr>
<tr>
<td>kb</td>
<td>kilo-bit (1000 bits)</td>
</tr>
<tr>
<td>kB</td>
<td>kilo-byte (1024 bytes)</td>
</tr>
<tr>
<td>LAN</td>
<td>Local area network</td>
</tr>
<tr>
<td>LMP</td>
<td>Link management protocol</td>
</tr>
<tr>
<td>LOS</td>
<td>Line of sight</td>
</tr>
<tr>
<td>MAC</td>
<td>Medium access control</td>
</tr>
<tr>
<td>MPDU</td>
<td>MAC Protocol Data Unit</td>
</tr>
<tr>
<td>MPRG</td>
<td>Mobile and portable radio research group</td>
</tr>
<tr>
<td>PCS</td>
<td>Personal communication system</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------</td>
</tr>
<tr>
<td>PDA</td>
<td>Personal digital assistant</td>
</tr>
<tr>
<td>PDU</td>
<td>Protocol data unit</td>
</tr>
<tr>
<td>PHY</td>
<td>Physical layer</td>
</tr>
<tr>
<td>PLCP</td>
<td>Physical Layer Convergence Protocol</td>
</tr>
<tr>
<td>PN</td>
<td>Pseudo-noise</td>
</tr>
<tr>
<td>PPDU</td>
<td>PLCP protocol data unit</td>
</tr>
<tr>
<td>RSSI</td>
<td>Received signal strength indicator</td>
</tr>
<tr>
<td>RTS</td>
<td>Request to send</td>
</tr>
<tr>
<td>SDMA</td>
<td>Space division multiple access</td>
</tr>
<tr>
<td>SIFS</td>
<td>Short interframe space</td>
</tr>
<tr>
<td>SNR</td>
<td>Signal to noise ratio</td>
</tr>
<tr>
<td>STA</td>
<td>Station</td>
</tr>
<tr>
<td>TCP</td>
<td>Transport control protocol</td>
</tr>
<tr>
<td>TDMA</td>
<td>Time division multiple access</td>
</tr>
<tr>
<td>TDD</td>
<td>Time division duplex</td>
</tr>
<tr>
<td>UUID</td>
<td>Universal unique identifier</td>
</tr>
<tr>
<td>WG</td>
<td>Working group</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless local area network</td>
</tr>
<tr>
<td>WPAN</td>
<td>Wireless personal area network</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

The recent trend towards nomadic computing in office environments has been based on the proliferation of wireless local area networks (WLAN). These networks can take multiple forms - the most popular ones use the license-free spectrum allocated in the United States by the Federal Communications Commission (FCC) - since they require no investment in costly spectrum licenses. However, since these networks operate over unlicensed bands, there is a high probability that more than one network will coexist in any one region at any time, causing mutual interference. This network coexistence issue has recently received a large amount of attention, most of which has been centered around the activities of the IEEE 802.15 Part 2 effort [1].

If two or more independently operated systems share the same spectrum and attempt to operate at the same time, interference will occur. This problem has the potential to cause significant degradation in performance to large and expensive wireless infrastructure. Two main methods have been suggested to handle this issue: channel orthogonality and traffic control.

The channel orthogonality concept revolves around avoiding channels that are being used by some other system. Occupied channel avoidance through adaptive frequency hopping can
provide reductions in interference levels, since interference can be directly avoided. However, for this approach to be applied to older devices, it requires the retrofitting of hardware, which is impractical. Furthermore, adaptive frequency hopping is currently applicable only to devices operating at 2.4 GHz, not 915 MHz or 5.7 GHz [2]. FCC rules would need to change to allow adaptive frequency hopping at 915 MHz or 5.7 GHz.

Several methods for dealing with the coexistence issue have been discussed in the available literature. Chapter 2 of this dissertation provides some basic background information on the relevant aspects of both wireless communications and networking and is followed by a literature review of the coexistence mechanisms available in the public literature today. In this dissertation, a new statistical collision avoidance traffic control algorithm is presented. The motivation for seeking a statistical approach for collision avoidance is that the algorithm can be applied to layers above the Medium Access Control (MAC), allowing adherence to existing standards without modification of the hardware. Furthermore, the device using the statistical traffic control method treats all other devices operating in the vicinity as random processes, regardless of their makeup or application. This means that a device using this collision avoidance mechanism will be able to deal with both complex and simple systems. The foundation for this algorithm, an analysis of network traffic in statistical terms, is presented in Chapter 3.

The analysis in the first part of Chapter 3 is expanded to include the statistical collision avoidance algorithm. The performance of this algorithm is tested in simulation and the performance of the statistical back-off algorithm is compared to both a best effort and an exponential back-off method in Chapter 4, where the statistical collision avoidance algorithm shows an increase in efficiency of roughly 25%. An increase in power efficiency was seen for all traffic models used, but the magnitude of this increase in efficiency varied significantly among the different cases. The cost of this increase in efficiency is a penalty in throughput in the statistical collision avoidance system when compared to the best effort case. However, when compared to the exponential back-off case, the statistical collision avoidance system proves to have both higher throughput and higher power efficiency.
A theoretical analysis of the coexistence problem provides insight into resolving the problem but does not conclusively prove that the solution I propose is viable in a real system. To determine the viability of such an approach, two technologies are selected to test out the coexistence problem: Bluetooth and IEEE 802.11b. Chapter 5 presents a method that can be used to bypass some of the control features of the Bluetooth baseband controller, leaving such features under the control of the application. This methodology is test-proven in Chapter 5 with a Hybrid ARQ algorithm, which is based on previous published work.

In Chapter 6, the algorithm presented in Chapter 4 is adapted to fit an on-off traffic source based on a heavy-tailed distribution and an environment populated by IEEE 802.11b and Bluetooth devices. The algorithm’s viability is tested in a perfect detection scenario. The algorithm’s robustness is also tested in reference to changes in traffic pattern assumptions, detection thresholds, and the existence of multiple devices using the same algorithm in the local environment. Given that the coexistence algorithm presented in this dissertation is based on injecting an artificial delay when frames suffer collisions, manifested as bit errors, the penalty in decreased range for a Bluetooth device is determined through simulation.

One of the interesting aspects of the method presented for the implementation of the coexistence mechanism is that it can be integrated to an adaptive frequency hopping mechanism for Bluetooth. The performance of this algorithm when implemented on a Bluetooth device is investigated, as well as a combination of the delay-based coexistence algorithm with the adaptive frequency-hopping coexistence algorithm.

Finally, Chapter 7 presents a summary of this dissertation, including key contributions, suggested future work, publications to date, and suggested publications.
Chapter 2

Background

2.1 Wireless Communications

This dissertation covers the wireless local area network (WLAN) behavior. To have the background necessary to understand the work presented in this dissertation, several aspects of WLAN systems need to be covered. However, the full understanding of WLAN requires the understanding of radio channels, radio interfaces, link protocols, and traffic models. Given the makeup of the audience that is most likely to read this dissertation, the presentation of some information would be redundant, if not outright boring. Therefore, there are several aspects that will not be covered in this dissertation. These aspects are listed in Table 2.1. If any of these topics are not clear to the reader, the following references are suggested for further reading: [3][4].

Beyond the background knowledge that is expected, all information that is required to understand the work presented in this dissertation is presented in this chapter. This information includes unlicensed band device regulations, the structure of both Bluetooth and IEEE 802.11b, the IEEE 802.15 Part 2 channel model, and data traffic models.
Table 2.1: Key Terms and Concepts

<table>
<thead>
<tr>
<th>Layer</th>
<th>Technology</th>
</tr>
</thead>
<tbody>
<tr>
<td>PHY</td>
<td>Direct Sequency Spread Spectrum (DSSS)</td>
</tr>
<tr>
<td>PHY</td>
<td>Frequency Hopping Spread Spectrum (FHSS)</td>
</tr>
<tr>
<td>PHY</td>
<td>Channel filter</td>
</tr>
<tr>
<td>PHY</td>
<td>Receiver Noise Temperature</td>
</tr>
<tr>
<td>PHY</td>
<td>Additive White Gaussian Noise (AWGN)</td>
</tr>
<tr>
<td>PHY</td>
<td>Differential Binary Phase Shift Keying (DBPSK)</td>
</tr>
<tr>
<td>PHY</td>
<td>Differential Quaternary Phase Shift Keying (DQPSK)</td>
</tr>
<tr>
<td>PHY</td>
<td>Gaussian Minimum Shift Keying (GMSK)</td>
</tr>
<tr>
<td>PHY</td>
<td>Large- and Small-scale fading</td>
</tr>
<tr>
<td>MAC</td>
<td>Forward Error Correction (FEC)</td>
</tr>
<tr>
<td>MAC</td>
<td>Packet synchronization mechanisms</td>
</tr>
<tr>
<td>MAC</td>
<td>Pseudo-noise (pn) sequence</td>
</tr>
</tbody>
</table>
2.1.1 FCC Part 15 Rules

The Federal Communications Commission, which releases rules under Title 47 of the United States Federal Code, governs several aspects of non-military communications, including rules concerning the use of spectrum. Of interest in this work is Part 15 of the FCC rules, which govern the use of unlicensed spectrum, which is now limited to 915 MHz (902-928 MHz), 2.4 GHz (2400-2483.5 MHz), and 5.7 GHz (5725-5850 MHz). It should be noted that, ironically, there are some licensed users of the unlicensed spectrum. For example, the 2.4 GHz band overlaps with spectrum allocated to the amateur radio band for television broadcasts. Therefore, the Part 15 rules state that "Part 15 devices may not cause any harmful interference to authorized services and must accept any interference that may be received [5]." Given these constraints, devices operating under Part 15 rules were originally limited to the use of spread spectrum, either FHSS or DSSS, since transmissions would either linger in a single channel for short periods of time, in FHSS, or would have low power densities, in DSSS. However, at the time that these rules were set, commercial spread spectrum technology was limited in capabilities, and the data rates achievable today were not envisioned. Therefore, new rules have been set by the FCC that maintain the low-interference nature of spread spectrum yet allow the deployment of more advanced systems. The Part 15 rules of interest concern FHSS and DSSS, which will be described in the following sections.

Rules for FHSS

As of July 2001, the rules governing the 2.4 GHz band are different than those governing the 915 MHz and 5.7 GHz bands. Since the systems of interest in this work are designed to operate at the 2.4 GHz band, only those rules will be discussed.

A FHSS system is required to use at least 15 non-overlapping channels, regardless of the bandwidth of the signal. This means that a device using FHSS and channel bandwidths of 5 MHz is required to use a total of 75 MHz out of the total 83.5 MHz of available spectrum,
while a device using 1 MHz channels must use at least 15 MHz of the available bandwidth. However, since devices using less than 1 MHz channel bandwidths can completely occupy a full channel under use by a device using wider bandwidths, devices using channels of 1 MHz or less must use adaptive frequency hopping to avoid undue interference on other devices. It should also be noted that devices using channel bandwidths of 1 MHz or less are not required to use adaptive frequency hopping if they use a hopping scheme that covers the whole unlicensed band. One last restriction is the transmit power. Devices using FHSS with channel bandwidth greater than 1 MHz are limited to 125 mW. Devices with channel bandwidth less than 1 MHz but with adaptive frequency hopping are also limited to a transmit power of 125 mW. Finally, all other FHSS devices are allowed to a transmit power of up to 1 W.

Rules for DSSS

The rules for DSSS are simpler than those governing FHSS systems. Devices using DSSS are, like FHSS, limited to transmit powers of 1 W. The spreading sequence is required to provide a minimum of 10 dB processing gain. Initially, the measurement of the 10 dB processing gain was performed by applying a continuous wave (CW) signal in the passband of the device and measuring the performance of the equipment. However, this method does not necessarily provide an indication of the system’s real processing gain. Therefore, the Part 15 rules state that a Gaussian noise signal be used as the reference signal against which the processing gain of the system is measured.

2.2 Protocols

There are several layers that need to be implemented in a system to provide full functionality to a network. The main focus of this work is the 2.4 GHz ISM unlicensed band, providing a framework for the types of devices that are the target of this work. The ISM band can support
devices ranging from simple garage door openers and cordless phones to sophisticated devices integrated into a large network infrastructure, such as IEEE 802.11 variations. Given the wide variety of devices that can be supported in this band, it is difficult to narrow down the services to a set that is manageable. Therefore, this work will concentrate on minimizing the impact of a sophisticated network on a simple service provided over a simple standard such as Bluetooth. The algorithm presented in this work to minimize interference on a network is capable of having a significant impact on a more sophisticated systems implementing IP (Internet Protocol) and TCP (Transport Control Protocol), resulting in timeouts and other link impediments. Although the impact of the wireless channel on the upper protocols is an active area of research [6][7], the impact of the algorithm presented in this work on these sophisticated protocols is left open for future research.

The networking overview presented in this chapter covers the fundamentals of links and concentrates on the packetization structure and error recovery mechanism inherent to the MAC (medium access control) layer of the system. The two protocols that will be described in sufficient detail to provide this background are Bluetooth and IEEE 802.11. Suggested further reading in networks includes [8][9].

A communications network can be defined as a set of devices that are communicating with each other, where each device in the network can be considered a single node, as seen in Figure 2.1. The establishment and termination of that link is beyond the scope of this work, but the air interface and the packetization scheme that Bluetooth and IEEE 802.11 follow are described in sufficient detail in the following sections.

### 2.2.1 Bluetooth

Bluetooth is a wireless personal area network (PAN) specification that is designed by the Bluetooth SIG (Special Interest Group), a group of over 2000 companies, to provide very inexpensive connectivity between devices that are separated by short distances, normally under 10 meters. The raw data rate of Bluetooth devices was specified to be 1 Mbps,
Figure 2.1: Generic Network

providing a solution for most short distance applications. The actual maximum data rate that a Bluetooth device can deliver is significantly lower than 1 Mbps, but more details on the system data rate will be discussed in a later section of this overview.

The Bluetooth specifications cover the radio, baseband processing, link management, link control and protocol adaptation, service discovery, and a suite of protocols that are designed to cover some specific applications. This Bluetooth overview will cover the sections that are of direct interest to this dissertation, the radio and baseband specifications. A full discussion of the Bluetooth specifications is beyond the scope of this work, but the full specifications are available at [10], and there are several descriptions of the standard such as [11]. Unless otherwise stated, this description is based on [10].

Radio Specifications

Bluetooth devices operate in the unlicensed ISM (Industrial, Scientific, and Medical) band at 2.4 GHz, which is regulated by the FCC Part 15 rules, which are discussed in Section
2.1.1. While the ISM band has been allocated throughout the world, its specific position is country-dependent. However, the United States and most of Europe have allocated the space between 2400 and 2483.5 MHz for this band. Bluetooth devices use 79 channels within this band, each occupying 1 MHz.

Bluetooth uses FHSS and a TDD (time division duplex) scheme for its link. The device controlling the link, the identity of which is discussed in the Section 2.2.1, determines the pn-sequence offset used in the frequency-hopping scheme and also sets the network clock. The nominal hopping rate used is 1600 hops per second, but this rate can change depending on the packet type being transmitted. The pn-sequence used to determine hopping scheme is a cyclic code of length $2^{27} - 1$. Within a single link, Bluetooth is a slotted system, where slots are 625 $\mu$s long, with a single packet being transmitted per slot. When a new packet is transmitted or when a packet needs to be retransmitted, a new channel is selected. TDD is implemented by alternating the transmission slots used by the devices in the network.

The modulation used is binary GMSK, where the symbol rate for the channel is 1 Msymbol per second, yielding a raw bit rate of 1 Mbps.

**Baseband Specifications**

Bluetooth devices are arranged in a master/slave configuration, where a master controls all the aspects of the link with the slave, including the channel selection and the time slots in which the slave is expected to transmit information. The establishment of a master/slave relationship yields a piconet, where a piconet is a connection between a master and one or more slave devices, as seen in Figure 2.2. The seed value for the pn-sequence used to selected the channel in the FHSS scheme is controlled by the master, thereby maintaining control over the current channel used by all the devices in the piconet. The TDD scheme is also controlled by the master device, where the traffic in a sample link is seen in Figure 2.3.

Bluetooth is a packet system, where data is split into small segments and transported as an
Figure 2.2: Bluetooth Piconet

Figure 2.3: Piconet TDD scheme
individual piece of data. The amount of data that each packet can transport depends on the packet length, but in Bluetooth it will never be more than 344 bytes. Bluetooth can support several packet types for both synchronous and asynchronous links. The synchronous packets have strict slot requirements and are designed to transport voice, but the description of this type of packet is beyond the scope of this dissertation, since the focus of this work is data applications, not voice. Asynchronous packets are designed to transport data. There are two basic types of asynchronous packets, high-speed and medium-speed, which are called DHx and DMx, respectively, where x is the number of slots that the packet is occupying; the concept of multi-slot packets is discussed later.

The only difference between the DHx and DMx packets is the FEC used in the payload. DMx packets use a \((15, 10)\) shortened Hamming code on the payload, allowing the packet to system to correct up to two bit errors for every fifteen bits sent. Since five bits of redundant information are added to each 10 bits of raw data, the data rate of the system is \(2/3\) of the maximum data rate. The DHx packets, on the other hand, have no FEC on the payload of the packet. Since no redundant data is added to this packet, the maximum raw data rate is relatively high.

As mentioned above, Bluetooth packets can span multiple slots. There are three packet sizes: 1, 3, and 5, making the names for the packets Dn1, Dn3, and Dn5, respectively. Each packet, regardless of the number of slots it occupies, is transmitted over a single channel. Therefore, if a link consists of packets whose length is greater than a single slot, the hopping rate of the Bluetooth system will be lower than the system’s nominal 1600 hops per second rate.

Since packet systems treat each packet as a separate entity, packets need to be composed of more than just a payload, allowing information such as the destination address to be included in the packet. Bluetooth packets include a 72-bit access code and a 54-bit header in the beginning of each packet. The access code is used for synchronization, DC offset compensation, and identification, where the identification is a piconet ID, simplifying the
mixing of multiple piconets in an environment. The header contains link-specific information, such as the device piconet address, packet identifiers, and link error recovery information. Note that the header uses a rate 1/3 repetition code as protection.

While Bluetooth packets can contain FEC in the payload, there is no real guarantee that the packet was delivered. To provide this guarantee, Bluetooth employs an ARQ (automatic transmission request) algorithm, where each packet that is sent is acknowledged. By using this method, if a packet is acknowledged with a negative acknowledgment due to an unrecoverable number of errors in the payload, or if the packet is not acknowledged due to packet loss, then the sender can retransmit the information. Thus, a guarantee exists that if the packet can arrive at the destination, it will eventually arrive at that destination. Figure 2.4 shows the operation of such an ARQ scheme.

All Bluetooth packets have a similar structure, seen in Figure 2.5. As seen in Figure 2.5, the Bluetooth packet has a 72-bit access code followed by a 54-bit header, which in turn is followed by the packet payload. The access code is a bit sequence that is unique to the piconet and that is used by the members of the piconet to perform synchronization. By performing a correlation operation with the received packet, the receiver can determine the beginning of the packet. The construction of this access code is based on the Master device’s UUID (Universal Unique Identifier), so it is guaranteed to be unique to the piconet. Also, the access code is constructed such that a minimum Hamming distance of 6 is maintained, giving the receiver the capability of discerning whether the packet is from the appropriate piconet with a level of certainty that can be calculated.
The packet header consists of 54 bits. However, the packet header uses a \( r = 1/3 \) repetition code, so in reality the packet header contains only 18 bits of actual information. Given that the packet’s access code has the dual purpose of performing synchronization and identifying the piconet to which the packet belongs, the task of the header is to perform some housekeeping functions as well as determining the identity of the destination device. The header contains 3 bits to be used for identifying the destination device, which means that up to 8 devices can be uniquely identified. Instead of allowing 8 devices to be controlled by a single Master device, the address space is set such that 7 slaves can be individually addressed, leaving the all-zeros address as a multicast address.

Finally, the packet payload contains anywhere between 240 and 2745 bits. This payload value is misleading, because packets will generally contain a 1- or 2-byte payload header that performs some further housekeeping. Beyond this, the variation in payload length is a function of the number of slots that the packet occupies. In this dissertation, the packets of choice occupy a single slot. Assuming a 1-byte payload header in a single-slot packet, the number of bits per packet is

\[
l_{BI} = 72 + 54 + 240 = 366
\]

The Bluetooth packet is expected to generally extend over 366 bits, which at 1Mbps translates to 366\( \mu s \). Given that the Bluetooth slot occupies 625\( \mu s \), that means that spectral load of Bluetooth can be described as

\[
load_{BI} = \frac{366\mu s}{625\mu s} = 58.6\% \tag{2.2}
\]
where the spectral load is defined as the proportion of the allocated channel time that is used for actual radiation transmission. The idle time between packets is generally used as oscillator settling time for the frequency hopping architecture. While 259µs sounds like a long time, it is important to remember that Bluetooth is designed to be implemented with inexpensive low-quality components.

The spectral efficiency of Bluetooth can be calculated based on the amount of useful data that is transmitted. Assuming the use of single-slot packets that carry no CRC but still need the 1-byte payload header, the packet usable payload is then \(240 - 8 = 232\) bits, which as 1Mbps translates to 232µs, giving a spectral efficiency of

\[
eff_{Bl} = \frac{232\mu s}{625\mu s} = 37.7\% \quad (2.3)
\]

This spectral efficiency assumes that the packets transport data in both the uplink and the downlink and the the packet contains a full load. If data is to be transported in only one direction, the slotting structure would still require the full slot for the acknowledgement, which leads to a simplex spectral efficiency of

\[
eff_{simplex} = \frac{232\mu s}{1250\mu s} = 18.6\% \quad (2.4)
\]

It should be noted that in some cases, the packet will not contain a full load, reducing the efficiency event further. Since there is overhead inherent to the transmission of Bluetooth packets, and since some packets use FEC, each packet type can support a maximum real data throughput that is lower than Bluetooth’s nominal data rate of 1 Mbps, as seen in Table 2.2. Note that this maximum throughput assumes that no re-transmissions occur.

It should be noted that the data rates shown in Table 2.2 are those for asymmetric links, where the data rate shown is that for the link direction using the higher data rate. The AUX 1 packet can be useful in leveraging other capabilities of Bluetooth, since the packet transports a payload that uses no FEC and contains no error detection mechanism (implemented as a
Table 2.2: Data rate associated with each Bluetooth packet type

<table>
<thead>
<tr>
<th>Packet type</th>
<th>Maximum data rate (kbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DM1</td>
<td>108.8 (symmetric)</td>
</tr>
<tr>
<td>DM3</td>
<td>387.2 (asymmetric)</td>
</tr>
<tr>
<td>DM5</td>
<td>477.8 (asymmetric)</td>
</tr>
<tr>
<td>DH1</td>
<td>172.8 (symmetric)</td>
</tr>
<tr>
<td>DH3</td>
<td>585.6 (asymmetric)</td>
</tr>
<tr>
<td>DH5</td>
<td>723.2 (asymmetric)</td>
</tr>
</tbody>
</table>

CRC), so no ARQ scheme is used with this packet. When using the AUX1 packet, data rates of up to 185.6 kbps are possible. It should be noted that the data rates mentioned are the maximum possible data rate, and in practice the rates would probably be lower, depending on the environment, layout of the devices, and quality of the device implemented.

This concludes the discussion of the Bluetooth specifications. There are several aspects of Bluetooth that were not discussed, but these aspects, while important, are beyond the scope of this description.

2.2.2 IEEE 802.11b

Like Bluetooth, the IEEE 802.11 standard covers the description of a sophisticated system, and its full description is beyond the scope of this work. The sections of interest in this dissertation are parts of the air interface, the packet structure, and the multiple access strategy. As in the Bluetooth description, Section 2.2.1, this description will cover selected areas of the radio and baseband sections of the system. Unless otherwise stated, the information in this section is based on [12].

The IEEE 802.11b standard is a subset of the IEEE 802.11 standard, designed to provide
WLAN (wireless local area network) connectivity. In turn, the IEEE 802.11 standard is part of the IEEE 802 suite of standards, designed to provide LAN/MAN (metropolitan area network) standards.

The IEEE 802.11 standard provides coverage based on either an independent service set (IBSS) for communication between at least two devices, a basic service set (BSS) for access to a fixed infrastructure through an access point (similar to a base station in cellular system), or an extended service set (ESS), which is designed to “create a wireless network of arbitrary size or complexity.” Examples of IBSS, BSS, and ESS are shown in Figures 2.6, 2.7, and 2.8, respectively, where “STA n” is device n. The service paradigm that the IEEE 802.11 standard is intended to support is essentially a wireless version of Ethernet (IEEE 802.3), where machines such as desktop computers use a wireless interface instead of a wire to access network services.

Radio Specifications

The original IEEE 802.11 standard offers two basic radio interface methods, FHSS and DSSS, but their full explanation is beyond the scope of this description. The air interface of interest is IEEE 802.11b, an extension of the IEEE 802.11 standard. IEEE 802.11b has received significant levels of interest for office connectivity, since it provides relatively high data rates. The main difference between IEEE 802.11 and IEEE 802.11b is the air interface, which is described in the next paragraph.
Figure 2.7: Basic Service Set

Figure 2.8: Extended Service Set
IEEE 802.11b uses a high-speed DSSS at the air interface, and is compliant with the Part 15 FCC rules, since it is designed to operate in the 2.4 GHz ISM band. The modulation used by IEEE 802.11b is a combination of DBPSK (differential binary phase shift keying) and DQPSK (differential quadrature phase shift keying), depending on which part of the packet is being transmitted. The data rate at which the packet is transmitted is 1 Mbps, 5.5 Mbps, or 11 Mbps, depending on which part of the packet is being transmitted and the mode being used. The different parts of the packet and their corresponding data rate are discussed in the baseband specifications, Section 2.2.2. Compared to the original IEEE 802.11, which supports rates of 1 or 2 Mbps, IEEE 802.11b provides a significant improvement in raw performance.

The chip rate is 11 MHz for the IEEE 802.11b signal, providing a bandwidth of 22 MHz. No frequency hopping mechanism is used in this standard, allowing the device to remain parked on a specific channel indefinitely.

**Baseband Specifications**

The IEEE 802.11 MAC layer, which is common to all variants of IEEE 802.11, contains several details concerning multicast and unicast transmissions. However, given the scope of this dissertation, the primary concern in unicast, where a single device is transmitting to another specific device. Multicast transmissions, where a single device transmits to a number of devices is beyond the scope of this work.

The main concern of this work is the frame format and multiple-access technique used in IEEE 802.11. IEEE 802.11 frames are composed of three sections: MAC header, frame body, and frame check sequence. The MAC header contains information such as the length of the frame and the destination address, the frame body contains the actual data the frame is transporting, and the frame check sequence is a 32-bit CRC used to perform error detection on the delivered frame.
A burst of data that needs to be sent over the IEEE 802.11b link is the MAC service data unit (MSDU). This MSDU is fragmented into segments that can be managed by the MAC, resulting in a series of MAC protocol data units (MPDUs) that are sent in a series to transfer a whole MSDU. Since IEEE 802.11 supports multiple air interfaces, a convergence layer is needed beyond the MAC frame. The convergence layer consists of an extra data set added to the beginning of the frame. Figure 2.9 shows the lengths of these different sections in bits. The IEEE 802.11b Physical Layer Convergence Protocol (PLCP) is used to bring together the a specific physical layer with the IEEE 802.11 standard as a whole. The PLCP Protocol Data Unit (PPDU), shown in Figure 2.10 is the packet that is transmitted over the air; the length of each section is shown in bits.

As mentioned in the Radio Specification section of the IEEE 802.11 description, different sections of the PPDU are transmitted in different ways. The PPDU preamble, which provides a synchronization field and a frame delimiter, and the PPDU header, which provides information such as signaling and length descriptions, are transmitted using 1 Mbps DBPSK. The MPDU, which in the PPDU is called PSDU, or PLCP service data unit, is transmitted using DBPSK or DQPSK, providing 5.5 Mbps or 11 Mbps service, respectively.

Multiple access in IEEE 802.11 is performed using CSMA/CA (carrier sensing multiple access with collision avoidance). As the name implies, CSMA/CA senses the presence of a carrier, through either physical layer sensing or a virtual carrier sense mechanism, where link messaging is observed to determine whether or not another device is transmitting. By using a sensing algorithm, the device can avoid collisions with other devices operating locally.
When the amount of data arriving for transmission at an IEEE 802.11 device exceeds a specific threshold, IEEE 802.11 uses a combination of request-to-send (RTS) and clear-to-send (CTS) messages to determine whether a data frame can be transmitted. This mechanism forms the basis of IEEE 802.11’s virtual carrier sense mechanism. It should be noted that the threshold that is used to determine whether or not the virtual carrier sense mechanism should be invoked is not defined by the standard. Instead, this threshold is determined by the implementation, although it is usually left up to the user as a device property. By listening to this exchange, devices detecting the transmission of the CTS message know that another devices will transmit soon, thus invoking the collision avoidance mechanism. Physical sensing is physical layer-dependent, and therefore varies between the different implementations, but the goal is to detect other transmissions and to invoke the collision avoidance mechanism.

As seen in Figure 2.11, the exchange of a packet is followed by an acknowledgement, guaranteeing that the data was received correctly. There is a short period of time between the packets in the RTS/CTS/Data/ACK exchange called Short Interframe Space (SIFS), which is defined as

\[
SIFS = RxRFDelay + RxPLCPDelay + MACProcDelay + RxTxTurnaroundTime \\
= 10\mu s
\]
Collision Avoidance

One of the key aspects of the IEEE 802.11 standard is that it uses a carrier sensing algorithm combined with a collision avoidance structure. The two methods for carrier sensing, virtual and physical carrier sensing, are briefly described in the previous section. This section discusses the collision avoidance mechanism that is triggered either when a transmission is detected or when a collision occurs.

At all times, any active IEEE 802.11 device is listening to the traffic being exchanged over the channel that it occupies. When data is transferred between two devices, the data header contains a marker indicating the length of the whole data set to be transferred, regardless of the number of frames over which the data will be transported. When devices that are not the destination device listen to traffic, they decode the packet header and are aware of the expected traffic load over the channel. This traffic monitoring has two effects. First, when data arrives for transmission at the device that is listening to traffic, the device can backoff until it thinks that no traffic is currently being transmitted, without needing to resort exclusively to a physical carrier sense mechanism. The other effect pertains to the RTS/CTS exchange. Since a device knows that data is being transmitted in the local area, it may receive an RTS request from a device that was outside the range of the data exchange, as seen in Figure 2.12. In this case, the device that is within the range of the data exchange is aware that the channel is occupied and does not make a CTS reply to the RTS request.

When data arrives at the transmission queue, the IEEE 802.11 device performs a carrier sense event. This carrier sense procedure is split into two distinct parts. First, the virtual carrier sense mechanism is triggered. In the virtual carrier sense part of the carrier sensing event, the IEEE 802.11 device determines whether or not the channel is occupied right now by monitoring the traffic in the local environment. If the virtual carrier sense is negative, the physical carrier sense mechanism is then performed. The physical carrier sense mechanism is not described by the IEEE 802.11 standard, so it is left up to the implementation to determine how the physical carrier sensing is performed. Since this is an implementation-
Figure 2.12: Detection of data prevents CTS reply to RTS reception (shaded ellipses indicate stations’ range)
Figure 2.13: IEEE 802.11 Contention Mechanism

specific aspect of the device, its implementation can lead to a competitive advantage for the manufacturer, so implementation details are not readily available in the open literature. One common aspect of all the implementations should be that the device can determine whether or not the amount of energy in the channel goes beyond a certain threshold. If the physical carrier sense algorithm does not detect a signal on the channel, then the data can be sent. However, if either the virtual or physical carrier sense algorithms detect a busy channel, then the contention mechanism is triggered.

The IEEE 802.11 contention algorithm is based on the idea that, when a contention occurs, the device that detected the contention should be able to, on average, be able to wait until the channel clears before transmitting. Figure 2.13 shows the basic mechanism.

As seen in Figure 2.13, a contention, described as a “Busy” state, leads to two separate delays in the link. The first delay is the DIFS delay, DCF Interframe Space (DIFS), where DCF is the Distributed Coordination Function, the logic that coordinates every active device that belongs to a BSS in a functional network. The DIFS is defined as

\[ DIFS = SIFS + 2 \times Slot = 50\mu s \]  

(2.6)

where a slot time in IEEE 802.11b is 20\mu s. Note that the DIFS value is longer than the SIFS value. By ensuring that the SIFS time is less than the DIFS time, which is less than the minimum amount of time another device will wait until the retransmission, the RTS/CTS/Data/ACK exchange maintains an atomic (indivisible) structure, ensuring that no other device that can hear the exchange will interrupt the process. After the DIFS time,
Figure 2.14: IEEE 802.11 Error Recovery

The device will wait a length of time bound by the contention window (CW). The contention window is a number between 31 and 1023 slots, which translates into a window that is between $620\mu s$ and $20460\mu s$. The actual time that the delay lasts is a random number between 0 and the current CW value. If, while the device is waiting the amount of time described by the CW-derived random number is interrupted by a transmission, where the interruption is triggered by carrier sense mechanism, the amount of time expired is stored. After the channel is no longer busy, the contention mechanism is started again with the DIFS delay, with the exception that the CW-derived waiting period is the length of time that expired before the device’s CW-derived delay was interrupted. After the CW-derived delay expires, the device goes through the RTS/CTS exchange and, if this exchange is successful, the data is transmitted. If the RTS/CTS exchange is not successful, then the busy-channel mechanism is started again. However, this time the CW value is set to double the CW value used in the last iteration. Regardless of the number of failed transmissions, this CW value is not to exceed 1023 slots. If the data is successfully transmitted, the CW value is set to the minimum value, which in this case is 31 slots.

The mechanism when either the data packet or the acknowledgement is received incorrectly is similar to the contention mechanism. The error mechanism is seen in Figure 2.14.

As seen in Figure 2.14, when a data packet or the acknowledgement is received incorrectly, essentially the same behavior that occurs during contention is triggered. After waiting a length of time equal to the DIFS time, there is a delay equal to a random number between 0 and the current CW value. After this delay, the RTS/CTS exchange is implemented, after which, if successfully completed, the rest of the data that was to be sent in the interrupted
exchange is finally transmitted. One of the main differences between error recovery and contention is what occurs to other devices.

When another device listens to an IEEE 802.11 link between two other devices and detects an error in that link, it will wait a total of EIFS before transmitting, where EIFS is defined as

$$EIFS = SIFS + bACK + hPLCP + DIFS = 364\mu s$$  \hspace{1cm} (2.7)

where SIFS and DIFS are the defined delays, $bACK$ is the time to transmit the ACK packet, 112$\mu s$, and $hPLCP$ is the transmission time for both the PLCP preamble and header, 192$\mu s$.

Since the channel is busy when the IEEE 802.11 device detects the other devices’ failed packet transmission, after the EIFS time the CW-derived delay will be incurred. The added EIFS time guarantees that the link over which the error occurred will be able to terminate the data exchange. This exchange allows all the devices in the local environment to be re-synchronized.

The behavior that should be noted from IEEE 802.11 is that its links contain a certain level of structure. Regardless of the length of the data segment arriving at the IEEE 802.11 device for transmission, the data sent over the air will have some structure. After a packet has been corrupted, the behavior of the link can be predicted with a certain level of reliability. This known structure will become significant in understanding the coexistence mechanism presented in Chapter 6.

### 2.3 Previous Work

The issue of WLAN/PAN interference has received significant levels of attention in both industry and academia. At the core of this attention is the belief that when multiple types of WLAN and/or PAN devices operate in the same environment simultaneously, levels of
interference will be such that performance will become seriously degraded. Since this is a relatively new area of study, there is a wide variety in estimates as to the level of degradation that is likely. Another key issue is that there has been little work done in the characterization of the expected traffic patterns for WLAN and WPAN; this problem is especially acute with systems such as Bluetooth, which provide a whole new type of service. However, all the studies agree that some level of degradation will occur, and this level, depending on device location and traffic patterns, can be significant [13][14][15][16][17][18]. An important point to note is that, given the limited availability of devices, most work available today is primarily based on simulation rather than on experimentation, although some of the publications have simulation work that is backed by experiments.

Given the approach suggested in this work, discussed in Chapter 4, there are two major areas that require a review of the current available work: backoff/traffic conditioning mechanisms and WLAN interference mitigation. The review of backoff/traffic conditioning mechanisms will cover the principal methods discussed in the publicly-available literature that attempt to deal with network congestion in wireless environments. The review of WLAN interference mitigation will cover the publicly-available information for dealing with interference in WLAN.

2.3.1 Backoff/traffic conditioning mechanisms

This dissertation presents an analysis of the traffic placed in a channel by a service, followed by the description and simulation of an algorithm designed to take advantage of the optimum re-transmission time derived from the initial analysis; this is a form of adaptive back-off or traffic conditioning, depending on where in the protocol stack this concept is implemented. The concept of adaptive feedback is not new and has been presented in different forms by several researchers [19][20][21]. It should be noted that the approaches presented in publications are limited in two respects: they are limited to a single system and, although backoff policy parameters may change, they preserve a static stochastic profile.
Throughout the seventies and eighties, the concept of shared medium was based on the concept of ALOHA. This concept proved to be very powerful and is today used in several standards, including Ethernet. For wireless media, improvements to ALOHA were presented in publications, including a version that uses a carrier-sensing mechanism [22]; this work led to the concept of creating an exponential-backoff policy as the standard approach for dealing with congested channels. IEEE 802.11 uses a combination carrier sensing mechanisms and an exponential-backoff contention window to achieve a functional multiple-access mechanism [12].

The exponential backoff window received a significant amount of attention, given that this window determines a significant part of the idle time of the transmitter between successful packet transmissions in IEEE 802.11. Different approaches were attempted by a variety of researchers for managing the contention between different devices on a network [23]. Approaches involving the selection of different contention window sizes as a function of the number of nodes present in the local environment have been suggested [24]. Also, backoff times not necessarily following an exponential series have been suggested, including a geometric backoff [25], linear, Mu-Law, and Step-function backoff [26]. It should also be noted that a p-persistent method for selecting the transmission time during the contention window could be used by these algorithms.

Every one of the listed publications is based on research on a single system. In the eighties, research tended to focus on ALOHA systems, and in the nineties research tended to focus on IEEE 802.11 systems. None of the published work addresses the use of traffic conditioning or dynamic backoff to deal with systems spanning multiple standards. The concept of network coexistence has been explored in several publications, but the policies discussed were limited to dealing with issues related with specific standards rather than the issue as a generic problem related to specific standards.

While the use of a dynamic backoff algorithm has been explored by several researchers, the basic model used in the backoff policy never changes its general form - be it an exponential
or geometric backoff policy, the basic form of it is static. This dissertation explores the use of a dynamic policy that changes the form of the backoff (or traffic conditioning) policy to match the state of the local network using a statistical basis.

2.3.2 WLAN interference mitigation

There are two major factors limiting the availability of information on this topic. First, relative to the publication of this document, it is a new topic, where the issue has become a major concern only in the past twelve to eighteen months, limiting the number of publications addressing this issue. Second, there is a strong interest in the commercial sector to solve this problem, therefore, the specific solutions to this problem are only discussed in general terms, where the specific solution is often proprietary. While there are several publications that discuss the severity of this problem [13][14][15][16][17][18], only a few resources were found that address these issues. Furthermore, these publications deal with the explicit issue of interference between Bluetooth and IEEE 802.11b. It should be noted that the unlicensed bands allow a multitude of devices to operate, where the only restriction is that the air interface operate within some spread spectrum parameters. Therefore, while the approaches suggested to date provide a framework for dealing with this issue in the short term, they do not necessarily address the long term needs of devices operating in the unlicensed bands.

Several strategies have been suggested to deal with the issue of Bluetooth/IEEE 802.11b interference. These strategies are summarized in Table 2.3. Each of these approaches is discussed in more detail in the following sections.

Adaptive or Restricted Frequency Hopping

This method is one of the most promising approaches for minimizing interference. Since the device is capable of detecting which channels are being used and can adaptively avoid those channels, the goal of little (due to adjacent channel interference) or no interference
Table 2.3: Suggested methods to mitigate interference between Bluetooth and IEEE 802.11b

- Adaptive or restricted frequency hopping
- Switching between protocols
- Hardware-level interference mitigation
- System-wide solutions
- Smart antennas
- Dynamic channel assignment
- Traffic conditioning
- Nothing

should be achievable. However, there are two basic problems when using this approach. First, data collection becomes an issue, since it is unknown which other device is actively using that channel. For example, if the algorithm were designed to integrate Bluetooth in an environment where IEEE 802.11b is deployed, and instead of deploying the device in an environment with IEEE 802.11b, it was deployed in a strictly-Bluetooth (without adaptive frequency hopping) environment, then the adaptive device would unnecessarily avoid channels. Since IEEE 802.11b is parked on a specific band, collisions seen by a Bluetooth piconet mean that the piconet is currently operating over occupied channels, signaling the adaptive frequency hopping algorithm to avoid the occupied channel. However, if the Bluetooth piconet was operating nearby other piconets (not using adaptive frequency hopping), then a collision would lead the piconet with adaptive frequency hopping to erroneously conclude that another IEEE 802.11b network is currently parked on that channel. Second, if three IEEE 802.11b links were active, then the bulk of the spectrum would be occupied, severely restricting the spectrum available to the adaptively-hopping device. Third, any new Bluetooth device employing the adaptive frequency hopping algorithm would, by definition, need to interface with legacy Bluetooth devices (versions 1.0 and 1.1). Since the older devices have no adaptive frequency hopping option, all the devices within the piconet would be forced to
employ the full 79 channels listed in the Bluetooth specifications.

Switching Between Protocols

There are two basic approaches that can be taken with protocol switching. First, there is a hardware switch, where, when two different radios are colocated, only one of the radios is operational at any one point in time. This approach can have some significant problems, due to operational issues concerning entry and exit from a network, leading to network performance problems. These problems can be alleviated through the use of signaling, thus broadcasting the device’s temporary cessation of operation.

Second, a software switch can be used, where each MAC layer is sent information in a TDMA (time division multiple access) fashion. This approach suffers from similar problems, since the device may be taken off line at an inappropriate time, causing network problems. Furthermore, since the software is expected to operate in a non-real-time device such as a desktop computer, the timing cannot be controlled very accurately, introducing additional latency (delay) and jitter (latency variation).

It should be noted that this approach applies only in the instance in which two devices are operating within the same platform. It does not address the case in which the devices are near each other but are not under the control of a single computer.

Hardware-level interference mitigation

This approach employs active cancellation techniques available today in conventional radio implementation. Using this method, active signal cancellation can be employed, since both of the transmitted signals are known to the device. As in the "Switching Between Protocols" approach, this approach covers only one of the devices, and addresses only interference resulting from the colocated devices. It does not address signals originating at other devices.
System-Wide Solutions

A system-wide solution requires the use of some software agent that manages the devices operating in a local network [27]. The agent can manage issues such as traffic flow, restricting some devices from transmitting when it is known that the transmission will cause interference, and can also be used in spectrum management. The key technical restriction to this approach is that the agent is required to maintain control over all the devices in the network, regardless of which protocol they follow. This means that any device in the ISM band needs to register itself with the local server and it needs to be able to support the agent software. This is not a realistic approach, since some devices may be too simple to support such software. In the cases where the device is not too simple to support this software, the device user may be unwilling to include this agent into the protocol stack. Regardless of the actual solution, the concept of cross-layer optimization in a network such as Bluetooth can yield improvements in system performance [28].

Smart Antennas

From a purely performance perspective, this approach holds a lot of promise. Through the use of smart antennas, it is possible to steer a beam in the direction of the desired user, while at the same time nulling out sources of interference [29]. Significant improvements in networks using CSMA/CA, such as IEEE 802.11, have been shown when smart antennas are used [30]. However, there are several issues concerning this approach.

First, the goal of most devices operating in the ISM band is to maintain a low price point. Smart antennas require the use of multiple RF chains, which can drive the cost of the equipment beyond acceptable levels.

Second, smart antennas generally require a separation of $\lambda/2$ (half a wavelength), which at 2.4 GHz amounts to a distance of 6.3 cm, or 2.48 in. As the number of elements in the antenna array increases, narrower beams are possible and a larger number of sources
of interference can be nulled out. However, given that mobile Bluetooth and IEEE 802.11 devices are relatively small, this separation can make the hardware bulky.

Even though there are several drawbacks to using smart antennas in mobile WLAN or PAN devices, they could provide significant improvements in performance in fixed infrastructure such as IEEE 802.11 access points. Given that the number of access points in a network is smaller than the number of mobile devices, and given that the access points are fixed, the drawbacks inherent to small antennas could decrease in importance significantly.

**Dynamic Channel Assignment**

The concept of channel assignment has been a key one in the deployment of sophisticated networks such as cellular and PCS networks. It follows logically that this concept could be expanded to WLAN. Several researchers have focused on this concept to improve the spectral efficiency of a network in networks using ATM [31][32], WLAN implementing HIPERLAN/2 [33], and WLAN using TDMA (time division multiple access)[34], and timing control of the Master device in a Bluetooth piconet [35]. The best aspect of this approach is that a coordinated spectral planning policy improves the overall capacity of a given band. However, a coordinated approach to channel assignment requires all the devices sharing the band to agree on the channels to use. While this approach is possible for homogenous environments such as an IEEE 802.11b WLAN, this approach is impractical for bands occupied by devices deployed by different service providers.

**Traffic Conditioning and Packet Fragmentation**

The concept of modifying a traffic pattern to minimize the probability of collision when applied explicitly to Bluetooth and IEEE 802.11b has been presented by other researchers [15]. The concept presented in [15] is to match bursts of Bluetooth packets to a scale similar to that seen in IEEE 802.11b packets. The goal of this approach is to essentially mimic
the structure of an IEEE 802.11b packet using a set of Bluetooth packets. Through this approach, it is possible to improve the performance of an IEEE 802.11b network between 19% and 29%, depending on the level of delay that the Bluetooth devices are willing to accept. Given that traffic conditioning will add a certain amount of latency to a network link, there will exist issues when multiple types of traffic are mixed into the network, causing scheduling problems [36].

Packet fragmentation is the process by which a large packet is broken up into smaller pieces to match the conditions existing at the lower layers, regardless of whether due to the underlying protocol or current channel conditions. There has been some promising work aimed at performing IEEE 802.11 packet fragmentation to match the bursty conditions of the channel [37]. This work could be expanded to account for bursty conditions of the channel due not just to fading and other impairments, but to manage channel interference.

The concept of packet scheduling, or traffic conditioning, combined into a system using smart antennas to establish a form of SDMA (space division multiple access) has also been explored [38]. Several of the concepts explored, including traffic conditioning, operate on a single layer of the system. Given the isolation inherent to system layers, most of these approaches are complementary, and can be matched to provide a solution.

**Nothing**

This is the simplest option. Even though performance degradation is expected, the network manager or the users of the network may be unwilling to employ any of the suggested strategies. Instead, they may prefer to accept the degradation in performance.

The presented strategies cover the approaches presented to date for dealing with interference issues in WLAN/PAN operating within interference range of each other. A combination of these strategies is possible [39] - such a combination is the core of the approach of companies such as Mobilian with their TrueRadio solution [40]. However, as seen in the summary of the
approaches, not all of these strategies are capable of addressing the WLAN/PAN interference issue in a more generic fashion, where instead of Bluetooth-IEEE 802.11b issues, it becomes a broader technology issue. The following chapter discusses a method that deals with this issue in a more generic fashion, where the solution can be applied to a wide variety of networks.

### 2.3.3 IEEE 802.15 Part 2

The IEEE is in the process of integrating Bluetooth systems into the IEEE 802 series of networking standards. The effort is called IEEE 802.15. The standardization effort is broken into four sections, listed in Table 2.4.

<table>
<thead>
<tr>
<th>Part</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bluetooth</td>
</tr>
<tr>
<td>2</td>
<td>Coexistence (with IEEE 802.11)</td>
</tr>
<tr>
<td>3</td>
<td>High-speed WPAN</td>
</tr>
<tr>
<td>4</td>
<td>Low-speed (inexpensive) WPAN</td>
</tr>
</tbody>
</table>

Part 2 of the IEEE 802.15 has a direct impact on this dissertation, since that part deals with coexistence. The official recommendation of the IEEE 802.15 is to implement frequency hopping in all future versions of Bluetooth. This approach has been proven appropriate in the avoidance of interference, as long as IEEE 802.11 devices are not occupying all the available channels. As stated in Section 2.3.2, this approach is not backwards compatible, since a legacy Bluetooth system cannot support adaptive FHSS, and, unlike the method presented in this dissertation, Adaptive FHSS requires spectral estimation. However, the Adaptive FHSS approach has become the central focus for the recommendations of the IEEE 802.15 Part 2 WG specifications [41].

The method suggested by the IEEE 802.15 Part 2 WG is to re-map the pn-sequence associ-
ation to specific physical channels. This re-mapping function is based on the identification of “bad” channels in the local environment. Once these “bad” channels are selected, the Bluetooth hopper avoids these channels. Of course, a certain interaction would be needed between the Master and Slave Bluetooth devices to determine which channels are “good” and which are “bad.” This exchange is performed by the LMP (link management protocol), which manages the link beyond just the PHY and MAC functionality. Furthermore, Bluetooth’s host controller interface (HCI) needs to be modified to include the new device commands that are necessary to implement the adaptive frequency hopping functionality.

One method that was suggested by the IEEE 802.15 Part 2 WG for managing this problem is loosely based on the performance of adaptive frequency hopping through selective delays in packet transmission [42].

The idea presented in [42] is to predict when a packet that the Master device will transmit is in a channel that is available for transmission. There are two problems with this approach. First, it is designed for DM1 packets, and not only is the slave packet’s transmitted frequency not taken into account, which can create interference in the Slave device’s local environment, but re-transmissions due to the error recovery mechanism could overlap with the IEEE 802.11b occupied channel. It should be noted that an adaptive frequency hopping method similar to this one was created concurrently and is presented in this dissertation as part of the research. The adaptive frequency hopping method presented in this dissertation does not suffer from the same problems that arise in [42].

2.4 Models

The models used to simulate a system play a key role in determining the validity of the results. In this dissertation, models for two different aspects of system behavior have been implemented, the radio channel and the data traffic. Before the specific models are discussed, the environment of interest needs to be reviewed.
2.4.1 Environment

There are two different sets of simulations presented in this dissertation. An analytically-friendly case and a more realistic case study using Bluetooth and IEEE 802.11b as the two different protocols for which coexistence needs to be achieved.

In the analytically-friendly set of simulations, a generic packet system is used to validate the analytical derivation of the algorithm that is the core of this dissertation, as well as to determine the performance of the basic network when the algorithm is implemented. In this case, the environment is dictated by the parameters that are tested. The primary goal of this set of simulations is to test the performance of the algorithm with respect to the traffic model only. In the analysis that is presented in Chapter 3 the algorithm’s structure is derived from only the traffic model of the “weak” system, i.e. the system that is most likely to suffer from the effects of interference, regardless of the likelihood that a collision would result in the corruption of a packet. To evaluate the theoretical validity of this approach, the only parameter that is needed is a traffic model that can be supported by both the analysis and the simulation. Thus, the radio functionality of the system, including path loss and other radio effects, can be completely described by the trivial transfer function $H(f) = 1$, where the radio interface is completely eliminated from the system. Because of these reasons, the radio channel of the analytically-friendly system are not discussed in Section 2.4.2, while the traffic used for testing this system is discussed in Section 2.4.3.

On the other hand, for a realistic case study a larger number of factors need to be considered. In this case, two distinct protocols need to be simulated, Bluetooth and IEEE 802.11b. Because these two specific protocols are simulated, a channel model that is appropriate for the 2.4 GHz band needs to be selected, with the appropriate receiver performance, both of which are described in detail in Section 2.4.2. Furthermore, because these different systems have different likely applications, traffic models that are appropriate for each of the different types of application need to be used; the details of each of these traffic models are discussed in Section 2.4.3.
2.4.2 Radio Channel

The radio channel of the simulations in this dissertation is defined as encompassing the propagation channel (path loss, large-scale fading, small-scale fading), antenna, and the radio receiver. As shown in the Bluetooth and IEEE 802.11b descriptions, the receivers of these systems are sophisticated, and there is significant leeway in the implementation details. From a propagation standpoint, the options are also complicated. There are several papers that describe different aspects of radio propagation, giving a different way in which the signal statistics can be predicted. Because the two systems simulated are a WPAN and a WLAN, the simulations are limited to indoor environments only. Even though the environment is limited to indoor, parameters like wall location and composition make performance prediction difficult.

The problem in establishing a reasonable model for an environment populated by both Bluetooth and IEEE 802.11b is not unique to the research presented in this dissertation. The IEEE 802.15 Part 2 effort, discussed in Section 2.3.3, has looked at this problem because it, too, is trying to find some solution to the coexistence problem. As part of the effort to find a solution to the problem, the IEEE 802.15 Part 2 Working Group has come up with a model that describes the behavior of a Bluetooth receiver, an IEEE 802.11b receiver, the effects of each system’s interference on the other system, and the expected channel. To simulate results that can be measured against a common reference point, this dissertation uses the radio interfaces and channel suggested by the IEEE 802.15 Part 2 WG.

IEEE 802.15 Part 2 - Bluetooth Interface

Since Bluetooth uses a form of FSK, the link probability of bit error, \( p_e \), as a function of \( E_b/N_0 \) is described as

\[
p_e = \frac{1}{2} \left( 1 - \frac{1}{2} \sqrt{1 - \frac{4E_b}{N_0}} \right)
\]
\[ p_e = \begin{cases} 0, & \text{if } E_b/N_0 > 20dB \\ 0.5, & \text{if } E_b/N_0 < 1dB \\ \min \left( e^{\frac{-E_b}{2N_0}}, 0.5 \right), & \text{otherwise} \end{cases} \]

(2.8)

where \( N_0 \) is the receiver’s noise power spectral density, described in Section 6.4.1, added to the received interference. The minimum spectral rejection characteristics of Bluetooth devices is directed by the Bluetooth standard. However, the spectral mask does not describe the relative susceptibility of Bluetooth devices to interference from either other Bluetooth devices or IEEE 802.11b devices. According to the IEEE 802.15 Part 2 WG, the strength of the interference is described in terms of the difference in center frequencies (dcf) of the two Bluetooth devices in the following terms

\[ \text{gain}_{\text{Bluetooth-Bluetooth}} = \begin{cases} 1, & \text{if } \text{dcf} = 0MHz \\ 10^{-1.05}, & \text{if } \text{dcf} = 1MHz \\ 10^{-3.7}, & \text{if } \text{dcf} = 2MHz \\ 10^{-5}, & \text{if } \text{dcf} = 3MHz \\ 10^{-5.1}, & \text{if } \text{dcf} > 3MHz \end{cases} \]

(2.9)

where \( \text{gain}_{\text{device1-device2}} \) is the additional loss in the link budget between \textit{device1} and \textit{device2} that accounts for the difference in center frequencies between respective devices. The interference rejection of Bluetooth to IEEE 802.11b is also a function of the difference in center frequencies. The rejection according to the IEEE 802.15 Part 2 WG is described as
\[ gain_{Bluetooth-802.11} = \begin{cases} 1, & \text{if } df = 0\text{MHz} \\ 10^{-1.13}, & \text{if } df = 12\text{MHz} \\ 10^{-3.45}, & \text{if } df = 13\text{MHz} \\ 10^{-3.6}, & \text{if } 14 \leq df < 22\text{MHz} \\ 10^{-5.6}, & \text{if } df \geq 22\text{MHz} \end{cases} \]

(2.10)

**IEEE 802.15 Part 2 - IEEE 802.11b Interface**

The IEEE 802.11b receiver uses a combination of DBPSK and DQPSK for its 11 Mbps packets. Also, IEEE 802.11b uses DSSS, which also has an effect on the performance of the receiver. The probability of bit error, \( p_e \) of the IEEE 802.11b 11 Mbps receiver is

\[ p_e = \begin{cases} 0, & \text{if } E_b/N_0 > 10 \\ 0.5, & \text{if } E_b/N_0 < 0.5 \\ \frac{128}{256} \text{SER11}(E_b/N_0), & \text{otherwise} \end{cases} \]

(2.11)

where \( \text{SER11}(x) \) is defined as

\[
\text{SER11}(E_b/N_0) = \min (\text{res}, 0.99999)
\]

(2.12)

\( Q_5(x) \) is defined as
\[
Q_5(x) = \frac{e^{-x^2/2} (x^4 + 9x^2 + 8)}{\sqrt{2\pi} (x^5 + 10x^3 + 15x)}
\]  

(2.13)

The BER for the IEEE 802.11b link when the data rate is 1 Mbps is described by

\[
\begin{align*}
    p_e &= 0, & \text{if } E_b/N_0 > 10 \\
    &= 0.5, & \text{if } E_b/N_0 < 0.5 \\
    &= \min(0.5, Q_5 \left( \sqrt{11 * E_b/N_0} \right)), & \text{otherwise}
\end{align*}
\]

(2.14)

Interference rejection is a function of the device with which there is a collision; it should be noted that interference is treated as AWGN. If the source of interference is a Bluetooth device, the rejection levels, which are a function of the difference in center frequency (dcf), are set as follows,

\[
\begin{align*}
    \text{Gain}_{802.11-\text{Bluetooth}} &= 1, & \text{if } dcf \leq 11\, \text{MHz} \\
    &= 10^{-1.1}, & \text{if } dcf = 12\, \text{MHz} \\
    &= 10^{-2.97}, & \text{if } dcf = 13\, \text{MHz} \\
    &= 10^{-3}, & \text{if } 14 \leq dcf < 22\, \text{MHz} \\
    &= 10^{-4.7}, & \text{if } dcf \geq 22\, \text{MHz}
\end{align*}
\]

(2.15)

According to the IEEE 802.15 TG2 model, the complementary code keying (CCK) signalling method used by the 11Mbps and 5.5Mbps frames result in an 8dB suppression of narrowband interference. Given that Bluetooth is a narrowband source of interference, the following modification of the interference solution needs to be applied

\[
\text{Gain}_{\text{Bluetooth-802.11}} = \text{Gain}_{\text{Bluetooth-802.11}} \cdot 10^{-0.8}
\]

(2.16)
IEEE 802.15 Part 2 - Channel

The most realistic channel models are generally based on site-specific propagation models, where the layout of the local environment leads to a prediction of the received signal characteristics. Site-specific propagation prediction is usually a large-scale issue, where the two factors that are key are the distance between the transmitter and the receiver and any objects which may block the line-of-sight (LOS) of the signal. Small-scale effects, which can have a significant effect on the signal level, are very difficult to predict, and are generally described by some random variable.

There are two principal problems with highly detailed channel models. The first problem is that a site-specific model leads to answers to only the current case, which is difficult to extrapolate into more generic cases. The other problem is purely computational. In simulations of wireless links, this computational problem is overcome with computing time. However, when placed in the context of a network, the time-varying link has to be considered in terms of link traffic. Normally, when a network is simulated, the traffic load, which is usually described by a random variable, requires a runtime sufficiently long to converge using Monte Carlo simulation. However, when a variable-envelope link is added to the simulation, the simulation time needs to be sufficiently long to allow for the convergence of both the link and the traffic load, which leads to a very high simulation time. Because of these problems, most of the network simulations attempt to resolve the wireless simulation using a static (not time-varying, i.e. AWGN) channel. The IEEE 802.15 Part 2 WG [43] has approached the problem by presenting the channel as a static channel.

The minimum distance that the channel model allows is 10$cm$, therefore a distance function can be described as

\[
D(x) = \begin{cases} 
  x, & \text{if } 0.1 \leq x \\
  0.1, & \text{otherwise}
\end{cases}
\]  

(2.17)
where $x$ is the distance and $D(x)$ is the value that is used in the path loss model. The transmitted power needs to be handled in a different way, since there exist in this environment two different systems with noticeably different air interfaces. Therefore, if the “link” that is being evaluated is the “link” between the IEEE 802.11b and Bluetooth device, then

$$
P_{Tx,IEEE802.11-Bluetooth} = \frac{P_{Tx,IEEE802.11}}{(22 MHz)}
$$

(2.18)

where $P_{Tx,IEEE802.11}$ is the power transmitted by the IEEE 802.11b device and $P_{Tx,IEEE802.11-Bluetooth}$ is the power transmitted over a single Bluetooth channel. Note that the word “link” is used to described the relationship between these two types of devices because a link budget needs to be evaluated between these two types of devices even though this link is unintentional.

The transmitted power is then modified only in path loss before it is passed through the receiver algorithm. A two-mode path loss model was selected by IEEE 802.15 to predict the received signal strength, which is defined as follows

$$
P_{Rx}(d) = P_{Tx}/(d^2 \cdot 10^{4.02}), \quad \text{if } d \leq 8
$$

$$
= P_{Tx}/\left((d/8)^{3.3} \cdot 10^{5.85}\right), \quad \text{otherwise}
$$

(2.19)

### 2.4.3 Data Traffic

In this dissertation, three types of system are investigated: generic, IEEE 802.11b, and Bluetooth. Because of each of these system’s goal, the basic application that they are intended to support is different, hence the traffic load to be supported is different. This section describes both the rationale for the selection of each traffic model and the traffic model itself.
Analytically-Friendly System

The analytically-friendly system’s goal is fundamentally a proof-of-concept that has little basis on any real system. The only connection between this basic system and a real wireless network is that it uses a packet-based structure.

Furthermore, this system is designed to prove that an analysis was not only done correctly, but that the algorithm presented in this dissertation is functional. Therefore, the ideal form for the traffic for this system is one that is easily translated from the simulation domain into the analytical domain, and back again.

Given that there is no indication on what the possible application for this system is, and given that the system is supposed to have a close relationship with an analytical basis, an exponential-based traffic pattern was selected. This traffic pattern is similar to that used for Poisson statistics, which have been widely used in the prediction of traffic in systems, or more accurately the expected waiting time in a queue, for several decades [44]. This work was initially pioneered by A.K. Erlang in the early 1900’s. His work was initially based on telephony, since his theories were based on work at the Copenhagen Telephone Company in Denmark [45]. This work was the basis for traffic studies for decades. Furthermore, given that it is based on exponential distributions, it lends itself well to analysis.

Poisson arrivals are based on the idea of exponential inter-arrival times. However, this system is based on a single service sent over a single channel. Given the fact that it is a single service, instead of setting the inter-arrival time as exponential, an on-off source was selected, where the off times are described by an exponential distribution of the form

$$f_{\text{exp}} (t) = \frac{1}{\lambda} e^{-\frac{t}{\lambda}} u(t)$$  \hspace{1cm} (2.20)

The simulation structure used for this system is a clocked system, which lends itself naturally to a slotted system. Therefore, the exponential inter-arrival time was converted into a probability mass function (pmf), in terms of $k$ slots, of the form
Another benefit of using Equation 2.21 is that this off time can be reused at a later time for the creation of web-like traffic, seen in the next section. The packet size was also selected so that its analysis is kept simple. In many wireless systems, packets are of fixed size, or have a small variability in size. Therefore, for this system, a distribution of packet length was selected that is uniformly distributed over a limited range. The packet length, in terms of \( k \) slots is

\[
p_{u}(k) = \begin{cases} 
\frac{1}{4}, & 3 \leq k \leq 6 \\
\end{cases}
\]  

(2.22)

**Bluetooth**

The selection of a traffic model for the Bluetooth system is challenging, primarily from the fact that Bluetooth is the first device of the family of devices classified as wireless personal area network (WPAN). Given that Bluetooth is the first device of the WPAN class, there have not been any measurements on the expected traffic source statistics for this type of device. Therefore, the best traffic pattern that can be used for this device is based on an educated guess.

Bluetooth devices will probably be used for applications such as PDA (personal digital assistant) synchronization, light print jobs, some basic telemetry, keyboards and mice, and file transfers.

Basic telemetry, where the data transfers are mostly housekeeping, and keyboard and mice will probably yield a relatively light load to the system. A telemetry housekeeping transfer, where a data transfer occurs every few minutes, would have a negligible effect on the radio
environment, because data transfers would be rare.

A keyboard, or a mouse, may sound like it would supply a relatively heavy load. A simple analysis with the keyboard as an example can show otherwise. A single key stroke may require the transfer of two Bluetooth packets, capable of transporting a combined total of 432 bits - note that the keystroke would be an ASCII character requiring 8 bits. One packet is needed for the transfer of the keystroke, and another packet is needed for the acknowledgement. The wastefulness of the packets, and the high frequency of keystrokes, imply a heavy load. However, assuming an average typist creating a document at 30 words per minute (wpm), the update rate can be calculated.

A word is defined as 5 characters [46]. Assuming that there is a single punctuation mark between words, the rate of 30 words per minute translates to 180 characters per minute, which leads to a total of 3 characters per second, or a character every 333 ms. A Bluetooth device has a packet exchange period of 1.25 ms, where the packet exchange period is the amount of time that the channel is active to exchange a single message. Assuming that a character is transferred over a single packet exchange, and assuming that there are no re-transmissions needed for error recovery, the Bluetooth duty cycle is then

\[ C = \frac{1.25}{333} = 0.38\% \]  \hspace{1cm} (2.23)

Given a duty cycle of 0.38%, it is clear that the Bluetooth device is using the channel on a very limited basis, so the interference created by this device should be minimal.

Given the light load from some services, the sample services that are left to consider are synchronizations, light print jobs, and file transfers. All of these services have several aspects in common. First, all of these services are, from a network perspective, a purely local issue. A synchronization will generally be between a handheld device and a desktop computer, keeping the traffic from going out over a larger network. Light print jobs will also be local, since both the handheld device requesting the print job and the printer are local, since the
same person that is holding the handheld device will probably be the one that will want to retrieve the piece of paper resulting from printing. Finally, a file transfer will also be generally local, since the transfer will probably between a handheld or laptop computer and a local server.

In all of these services, the data transfer will be local, which means that all the data that needs to be transferred is immediately available, so there should really be no queue empty time, unless the transfer is complete. Furthermore, since the data transfer is local, any higher-layer data transfer assurance method, such as TCP, will not have any multi-hop route to manage, hence there should be no significant delays between individual parcels of data. Therefore, the most accurate traffic source for this type of device is an infinite traffic source, where the queue is never empty. For this reason, an infinite traffic source was selected for all the Bluetooth simulations in this dissertation.

**IEEE 802.11b**

One of the ironic aspects of IEEE 802.11b is that its traffic is probably the most complex type of traffic, yet it is the easiest to describe. An IEEE 802.11b device is essentially a wireless version of an Ethernet device. The use paradigm for these two devices is fundamentally the same; email and web browsing. Ethernet traffic has received vast amounts of attention, especially as the Internet became a popular vessel for communication.

In 1994, Leland et. al [47] established a fractal method for describing trunked ethernet traffic. Fractal traffic is one that, regardless of the scale, maintains the same fundamental shape. Figure 2.15 shows a visual example of this effect; in this case, the figure shows an iteration, the $6^{th}$ iteration in this case, of the 2-dimensional Cantor set. This particular Cantor set started as a large black square. This square was subsequently broken up into 4 equally-sized black squares, and so on. When this iterative process reaches infinity, the set is defined as a Cantor set. In the example shown in Figure 2.15, regardless of the scale on which the figure is viewed, the pattern remains constant.
Figure 2.15: Fractal example
In terms of a stochastic process, self-similarity is only parallel to the example seen in Figure 2.15. In a stochastic process, self-similarity implies that the aspect that is stable regardless of scale is some statistical characteristic.

Since this initial work attributing self-similar characteristics to Ethernet traffic, further research has shown that self-similar traffic is indeed an accurate measure of Ethernet traffic [48][49][50][51].

The generation of self-similar traffic can be complex, and several methods have been suggested for its creation [52][53][54]. A method that is well known is based on the addition of multiple on-off sources with at least one heavy-tailed distribution for either the on or the off time, or both [55][56][49].

One of the heavy-tailed distributions that has proven to provide accurate modeling is the Pareto probability mass function [57], which is defined as

\[ p_P(t) = \alpha \cdot \beta^\alpha t^{-(\alpha+1)}, 0 < \beta < t, 0 < \alpha \]  

(2.24)

where \( \alpha \) describes how heavy the tail is and \( \beta \) describes the minimum value that the function described by the Pareto distribution can take. In real terms, a heavy-tailed distribution means that the probability that a particularly long event, a long burst of data when describing the “on” period of an on/off source, is relatively high.

The Hurst parameter, which describes the relative burstiness of the distribution, is calculated from a Pareto distribution [47] as

\[ H = \frac{3 - \alpha}{2} \]  

(2.25)

Ethernet can be described by a Hurst parameter of 0.9, which means that \( \alpha \) should be set to 1.2. However, heavy-tailed distributions have some interesting characteristics. Because a heavy-tailed distribution has a significant probability that a value is high, the
variability of the process described by the distribution can be significant. The variability and, in combination with \( \beta \), the mean are both controlled by \( \alpha \). When \( 1 < \alpha < 2 \), the variance of the distribution is infinite, but the mean is finite, while when \( \alpha \leq 1 \), both the variance and the mean of the distribution are infinite. It has been shown that a Pareto source with \( \alpha < 1.5 \) leads to a very long simulation time [55].

For these convergence reasons, unless otherwise stated, the \( \alpha \) value selected for the traffic source is 1.7, which is one of the construction blocks that is used to describe web traffic. This selected value provides a traffic source that converges in a reasonable time but at the same time can still be used for the construction of self-similar traffic. The exponentially-distributed source off times are varied, depending on the relative data rate of the link.

Note that the traffic selected for this set of simulation is not self-similar. The measurements performed in environments where a large number of users share an Ethernet trunk are modeled by collecting a large number of on-off sources following the characteristics described above. Given that the environment studied in this dissertation is limited in range, the number of users is very small - in this case, just one user. In such an environment, a single on-off sourced was considered to be a reasonable description of the likely traffic. Extensions of this work into environments with much larger number of users is described in the future work section of this dissertation.

## 2.5 Summary

This chapter provides an overview of most relevant aspects of both the Bluetooth and IEEE 802.11b specifications. Furthermore, the regulations controlling the spectrum over which these two systems is described, providing clues as to the regulatory decisions that have led to the coexistence problem between systems operating simultaneously over the same unlicensed band. The problem of coexistence between Bluetooth and IEEE 802.11b is addressed in terms of strategies taken today to overcome the problem along with the corresponding reasons why
each of these solutions has limitations. Since a coexistence algorithm is presented in this dissertation, this chapter culminates with a description of both the channel and system traffic models that are used in the different simulations in this dissertation. Before the algorithm can be described, the state of a communications link in terms of a traffic source needs to be discussed. The next chapter describes an analysis of the state of the channel given an on/off traffic source.
Chapter 3

Coexistence Analysis and Mitigation

This chapter presents an analysis of a WLAN in its most basic form, as a set of packets sent over a channel separated by an amount of time. This statistical analysis initially describes a single system. Once the description of a single system is complete, another system is added to the local environment. The interaction between these systems is then analyzed, yielding a description of the traffic in one system as the function of the other system. By applying a traffic model to this analysis, an optimum retransmission time is evident. This optimum retransmission time is then used as the basis for an interference mitigation algorithm described in Chapter 4.

3.1 Network Description

Assume that there exists a network $N_1$ delivering service $S_1$ over a particular channel. $N_1$ is a packet-based network using fixed wireless channels. $N_1$ also contains only two nodes, $D_1$ and $D_2$, where in the downlink data flows from $D_1$ to $D_2$ and in the uplink data flows from $D_2$ to $D_1$. Since the channels are fixed, assign channel $C_1$ and channel $C_2$ for the downlink and the uplink, respectively. Using these parameters, we can represent $N_1$ as in Figure 3.1.
3.1.1 Statistical Description of Network Traffic

The traffic supported by $S_1$ is arbitrary and will vary widely depending on the nature of the service and the user(s) using it. However, in the most generic terms, we can describe the traffic generated by $S_1$ as a sequence of data packets delivered by $D_1$ or $D_2$, as shown in Figure 3.2.

In Figure 3.2, the state of the channel is described by three random variables: $A$, the length of the packet, $B$, the separation between packets, and $\Gamma$, the time between arrivals. As can be inferred from their definitions, the inter-arrival time can be written as $\Gamma = A + B$.

In traditional network analyses such as those modeling traffic as a Poisson random process, the inter-arrivals are often assumed to be independent, since an infinite number of services...
are generating the traffic, and two packets can arrive at essentially the same time. In the case where a single service is generating the packets, the packet arrival depends on the length of the previous packet, since a single service is generating the traffic; a new packet cannot be generated until the old packet has been sent over the link.

An arbitrary probability mass function, or pmf, can be associated with $A$; each packet will have a discrete length, and the set of lengths that can be selected for packets is finite. When delivering a service over a network, the length of each packet is based on an arbitrary decision performed by $S_1$, based on the specific needs of the application and the packet structure allowed by the underlying link manager. In this analysis, the length of each packet can be assumed to be independent. Given a slotted system assumption, an arbitrary pmf for $B$ can also be generated, since there is a discrete amount of time between packets. As in the length of the packet, the separation between packets is arbitrarily chosen by $S_1$. Like the random variable describing the length of the packet, the separation between each packet is assumed independent in this analysis. The independence assumptions are that, while the current packet length is not dependent on the length of the previous packet and the idle time between two packets is not dependent on the idle length between the previous pair of packets, the idle time after a packet is sent depends on the length of the previous packet. These assumptions apply to the analysis presented. In Chapter 6, these assumptions are reconciled with the IEEE 802.11 MAC.

The joint pmf for $A$ and $\Gamma$ can be described given the conditional probability

$$p_{(A,\Gamma)}(w,k) = P[\Gamma = k|A = w] \cdot P[A = w] \quad (3.1)$$

Given that $\Gamma = A + B$, then

$$P[\Gamma = k|A = w] = P[B = k - w] \quad (3.2)$$

Furthermore, note that random variable $B$ can only take on non-negative values, and there-
Collecting Equations 3.1 through 3.3, $p_{(A,\Gamma)}(w,k)$ can be described as

$$p_{(A,\Gamma)}(w,k) = \begin{cases} 0, & \text{if } k < w \\ p_B(k-w) \cdot p_A(w), & \text{otherwise} \end{cases}$$ (3.4)

Using the joint pmf $p_{(A,\Gamma)}(w,k)$, the pmf for $\Gamma$, $p_{\Gamma}(k)$, can be obtained as

$$p_{\Gamma}(k) = \max(A) \sum_{w=0}^{\max(A)} p_{(A,\Gamma)}(w,k)$$ (3.5)

$p_{\Gamma}(k)$ describes the inter-arrival time between packets; an example of $p_{\Gamma}(k)$ follows in the next section.

### 3.1.2 Statistical Description of Network Traffic Example

Assume the traffic random variables $A$ and $B$ can be described by probability mass functions described in 3.6 and 3.7, respectively.

$$p_A(k) = \begin{cases} \frac{1}{4}, & k = 3, 4, 5, 6 \\ 0, & \text{otherwise} \end{cases}$$ (3.6)

$$p_B(k) = \int_{k-0.5}^{k+0.5} \frac{1}{\lambda} e^{-\frac{t}{\lambda}} u(t) dt, k = 0, 1, ...$$ (3.7)

where $u(k)$ is a unit step function, $A$ is a uniform random variable, and $B$ is an exponential random variable applied to a slotted system.
\( p_r(k) \) can be calculated by evaluating Equation 3.5 using the random variables described in 3.6 and 3.7.

### 3.1.3 Channel Statistical State

Define the random process \( \chi(k) \) as the number of interarrival periods up to time \( k \); its relationship with \( \Gamma \) is shown in Equation 3.9, which is discussed later in this paper.

The probability that the channel will be occupied at any time \( k \) after the beginning of a packet can be described by the theorem of total probability as

\[
P[\text{slot } k \text{ is occupied}] = \sum_{n=0}^{\infty} P[\text{slot } k \text{ is occupied} | \chi(k) = n] P[\chi(k) = n] \quad (3.8)
\]

The term \( P[\chi(k) = n] \) describes the probability that both \( n \) interarrival periods have expired by time \( k \) and \( n + 1 \) interarrival periods have yet to expire after time \( k \), hence

\[
P[\chi(k) = n] = P \left[ \sum_{i=0}^{n} \Gamma_i \leq k, \sum_{i=0}^{n+1} \Gamma_i > k \right] \quad (3.9)
\]

The expression \( \sum_{i=0}^{n} \Gamma_i \) can be considered as a single random process, defined as

\[
\hat{\Gamma}_n = \sum_{i=0}^{n} \Gamma_i \quad (3.10)
\]

Since \( \Gamma_i \)'s are independent random variables, the pmf of the sum of \( n \) random variables \( \Gamma_i \) can be described as the convolution

\[
p_{\hat{\Gamma}_n} = p_{\Gamma_1} \ast \ldots \ast p_{\Gamma_n} \quad (3.11)
\]

By replacing the random variable defined in Equation 3.10 into Equation 3.9, Equation 3.12 can be stated as
The expression $P[\text{slot } k \text{ is occupied}|\chi(k) = n]$ can be stated as

$$P[\text{slot } k \text{ is occupied}|\chi(k) = n] = P[A + \hat{\Gamma}_n \geq k|\hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k]$$  \hspace{1cm} (3.12)

By using the definition of conditional probability,

$$P[A + \hat{\Gamma}_n \geq k|\hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k] = \frac{P[A + \hat{\Gamma}_n \geq k, \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k]}{P[\chi(k) = n]}$$  \hspace{1cm} (3.14)

The term $P[A + \hat{\Gamma}_n \geq k, \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k]$ can be expressed as

$$P[A + \hat{\Gamma}_n \geq k, \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k] = P[A + \hat{\Gamma}_n \geq k, \hat{\Gamma}_{n+1} > k|\hat{\Gamma}_n \leq k] P[\hat{\Gamma}_n \leq k]$$  \hspace{1cm} (3.15)

Since $\Gamma = A + B$, Equation 3.15 can be stated as

$$P[A + \hat{\Gamma}_n \geq k, \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k] = P[A + \hat{\Gamma}_n \geq k, A + B + \hat{\Gamma}_n > k|\hat{\Gamma}_n \leq k] P[\hat{\Gamma}_n \leq k]$$  \hspace{1cm} (3.16)

which, by inspection, can be simplified to

$$P[A + \hat{\Gamma}_n \geq k, \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k] = P[A + \hat{\Gamma}_n \geq k|\hat{\Gamma}_n \leq k] P[\hat{\Gamma}_n \leq k]$$  \hspace{1cm} (3.17)

By the definition of conditional probability and the theorem on total probability, the conditional probability $P[A + \hat{\Gamma}_n \geq k|\hat{\Gamma}_n \leq k]$ can be stated as

$$P[A + \hat{\Gamma}_n \geq k|\hat{\Gamma}_n \leq k] = \sum_{i=0}^{k} P[A + \hat{\Gamma}_n \geq k|\hat{\Gamma}_n \leq k, \hat{\Gamma}_n = i] P[\hat{\Gamma}_n = i|\hat{\Gamma}_n \leq k]$$  \hspace{1cm} (3.18)
By inspection, \( P \left[ A + \hat{\Gamma}_n \geq k \mid \hat{\Gamma}_n \leq k, \hat{\Gamma}_n = i \right] \) is solved as

\[
P \left[ A + \hat{\Gamma}_n \geq k \mid \hat{\Gamma}_n \leq k, \hat{\Gamma}_n = i \right] = P \left[ A + \hat{\Gamma}_n \geq k \right] = P \left[ A + i \geq k \right] = P \left[ A \geq k - i \right]
\]

By applying the definition of conditional probability and simplifying the result, the equation \( P \left[ \hat{\Gamma}_n = i \mid \hat{\Gamma}_n \leq k \right] \) is solved by

\[
P \left[ \hat{\Gamma}_n = i \mid \hat{\Gamma}_n \leq k \right] = \frac{P \left[ \hat{\Gamma}_n \leq k \mid \hat{\Gamma}_n = i \right] P \left[ \hat{\Gamma}_n = i \right]}{P \left[ \hat{\Gamma}_n \leq k \right]} = \frac{P \left[ \hat{\Gamma}_n = i \right]}{P \left[ \hat{\Gamma}_n \leq k \right]}
\]

By applying Equations 3.19 and 3.20 to Equation 3.18, the solution for \( P \left[ A + \hat{\Gamma}_n \geq k \mid \hat{\Gamma}_n \leq k \right] \) can be described as

\[
P \left[ A + \hat{\Gamma}_n \geq k \mid \hat{\Gamma}_n \leq k \right] = \sum_{i=0}^{k} \frac{P \left[ A \geq k - i \right] P \left[ \hat{\Gamma}_n = i \right]}{P \left[ \hat{\Gamma}_n \leq k \right]}
\]

Applying Equation 3.21 to Equation 3.16 yields

\[
P \left[ A + \hat{\Gamma}_n \geq k, \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k \right] = \sum_{i=0}^{k} \frac{P \left[ A \geq k - i \right] P \left[ \hat{\Gamma}_n = i \right]}{P \left[ \hat{\Gamma}_n \leq k \right]} P \left[ \hat{\Gamma}_n \leq k \right]
\]

which, by cancelling terms, yields

\[
P \left[ A + \hat{\Gamma}_n \geq k, \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k \right] = \sum_{i=0}^{k} P \left[ A \geq k - i \right] P \left[ \hat{\Gamma}_n = i \right]
\]

Combining Equations 3.23 and 3.13 yields the expression

\[
P \left[ \text{slot } k \text{ is occupied} \mid \chi (k) = n \right] = \frac{\sum_{i=0}^{k} P \left[ A \geq k - i \right] P \left[ \hat{\Gamma}_n = i \right]}{P \left[ \chi (k) = n \right]}
\]
Combining Equations 3.24 and 3.8 yields the probability that the channel is occupied at any time \( k \) after the beginning of a packet:

\[
P[\text{slot } k \text{ is occupied}] = \sum_{n=0}^{\infty} \sum_{i=0}^{k} P[A \geq k - i] P[\hat{\Gamma}_n = i]
\]

(3.25)

### 3.1.4 Channel Statistical State Example

A numerical solution of \( P[\text{slot } k \text{ is occupied}] \) was generated by applying the example of \( p_{\Gamma}(k) \) presented in Section 3.1.2, yielding the numerical solution seen in Figure 3.3.

Figure 3.3: Probability of a new collision at time \( k \) after the original collision assuming the start of the packet is known.

Figure 3.3 shows the probability of a collision after \( k \) slots broken down into the joint probability of collision after \( k \) slots and \( \chi(k) = n \) seen in Equation 3.8. The effect of each term on the probability of collision as a function of time is evident. While the separation between
lobes is a function of random variable $B$, the spreading of the lobe is a function of the random variable $A$.

To validate the example seen in Figure 3.3, a Monte Carlo simulation was executed. This simulation used a slotted system. The random variables describing the traffic are those seen in Section 3.1.2. Once the traffic was generated, the network was allowed to run, and the results of the simulation were collected; these results are presented in Figure 3.4.

![Graph showing simulation and analysis results](image)

Figure 3.4: Validation of analysis of probability that the channel is occupied at time $k$ after the beginning of a packet

As seen in Figure 3.4, the simulation results closely match the analytical results. Figure 3.4 shows the probability that the channel will be occupied at any point of time $k$ after the beginning of a packet. Figure 3.4 also shows that, for this specific traffic set, there is one point at which the probability of observing a packet on the channel is at a minimum - the sensitivity of this minimum point is discussed in detail in Chapter 6. In the next section of this paper, this fact will be leveraged to minimize the interference between two different types of networks using the same band at the same time.
3.2 Coexistence Issue

Assume that a new network is added to the network configuration shown in Figure 3.1. Call this new network \( N_2 \); this network contains two devices, \( D_3 \) and \( D_4 \). \( D_3 \) sends data to \( D_4 \) over channel \( C_3 \) and \( D_4 \) sends data to \( D_3 \) over channel \( C_1 \). Note that now \( N_1 \) and \( N_2 \) share channel \( C_1 \). Finally, assume that \( D_1 \) transmits significantly more power than \( D_4 \), such that \( D_2 \) cannot hear transmissions by \( D_4 \) while \( D_3 \) can hear transmissions by \( D_1 \). Figure 3.5 shows the new network configuration.

To provide a starting point for the analysis, assume that both \( N_1 \) and \( N_2 \) operate in a slotted system; to simplify the analysis of the networks, the slots in each network are synchronized with respect to each other. Furthermore, assume that traffic generated by \( N_1 \) can occupy more than one slot, while traffic generated by \( N_2 \) occupies a single slot. Such a packet structure is similar to that seen in systems like Bluetooth [10]. A slot is defined as the smallest unbreakable unit of the packet - in practical terms, a slot can be a single bit or some other unit whose length is significantly less than the packet length, reducing the impact of the slotting assumption.

While this is a constrained example, it presents a concept that can be applied to a more sophisticated case. In a practical application of this approach, each bit in the packet can be
considered to be a single slot. In the algorithm presented in the next section, a rough form of synchronization is performed that is based on packet collisions. A packet in a typical system will contain several hundred or several thousand bits. If the single-bit slot assumption is applied, the symbol scale is significantly lower than the packet scale. When the scale between slots and the synchronization algorithm is as large as that described above, then an error in slot synchronization of 0.5 slots, or a complete lack of slot synchronization, would yield an error in packet synchronization stemming from slot mis-synchronization which is several orders of magnitude smaller than the packet scale.

3.2.1 Collision Awareness

Traffic observed by $N_2$ can be characterized by $P[\text{slot } k \text{ is occupied}]$, where an example of this characterization is seen in Figure 3.4. $P[\text{slot } k \text{ is occupied}]$ takes as a reference point the starting point of the packet; however, if $N_2$ is used to observe the traffic in $N_1$ and $N_2$ does not contain the capability to perform carrier-sensing, then the only way to detect the existence of a packet is to detect a collision. In general terms regarding $P[\text{slot } k \text{ is occupied}]$, a collision provides a rough form of synchronization between $N_1$ and $N_2$. This level of synchronization is rough, since packets generated by $N_1$ can occupy more than one slot and the collision could have happened anywhere over the length of the packet. Given that a collision occurred, the location within the packet with which it collided can be described by a random variable. This random variable is a function of the packet length; we select the random variable $\Theta$ as describing the slot with which traffic from $N_2$ has collided with a packet from $N_1$.

A new symbol is created to describe a collision between packets generated by $N_1$ and packets generated by $N_2$ in slot $k$, $\Psi(k)$. Given that a collision has occurred between $N_1$ and $N_2$, the probability of a new collision for the next packet sent by $N_2$ at slot $k$, $P[\Psi(k)|\Psi(0)]$, is then calculated by using $P[\Theta = x]$ as a condition for an offset of $x$ to the probability that $P[\text{slot } k \text{ is occupied}]$; by summing the different cases together, the probability that a packet will be in the channel if a transmission is made $k$ slots after a collision occurred in slot $x$ is
then

\[ P[\Psi(k)|\Psi(0)] = \max(A) \sum_{x=0}^{\max(A)} P[\text{slot } t+x \text{ is occupied}] P[\Theta = x] \] (3.26)

Since a collision has occurred with some slot in the system, \( P[\text{slot } t+x \text{ is occupied}] \) is more than just an offset version of Equation 3.25. The collision between the two systems means that the random sequence \( \Gamma_n \) now needs to be modified. Recall that \( \Gamma = A + B \) and that the expression given in Equation 3.25 assumes that the point of reference is the beginning of the current packet. Since a collision has occurred, the point of reference is now some slot beyond the beginning of the packet. Therefore, the first interarrival period is now defined as

\[ \tilde{\Gamma} = \tilde{A} + B \] (3.27)

where \( \tilde{A} \) is a random variable describing the rest of the first packet. Applying this concept to the example on this paper, if a collision occurs on the fourth slot of a packet, then the only possible values for \( A \) are 4, 5, or 6. Given the new set of possible values for \( A \), it then follows that a new random variable is needed, \( \tilde{A} \). Using the new definition for the first interarrival period, the sum of \( n \) interarrival periods where the first period is \( \tilde{\Gamma} \) is defined as

\[ \tilde{\Gamma}_n = \tilde{\Gamma} + \sum_{i=0}^{n-1} \Gamma_i \] (3.28)

and where the pmf of \( \tilde{\Gamma} \) is defined as

\[ p_{\tilde{\Gamma}_n} = p_{\tilde{\Gamma}} * p_{\Gamma} * \ldots * p_{\Gamma} \] (3.29)

A direct substitution of \( \tilde{\Gamma}_n \) for \( \hat{\Gamma}_n \) is only possible in most of the terms in Equation 3.25. The proof follows:
Start with the joint probability that $\hat{\Gamma}_n + A \geq k$ and $\hat{\Gamma}_n \leq k$ and $\hat{\Gamma}_{n+1} > k$ parallel to Equation 3.16 and apply the definition of conditional probability.

$$P \left[ \hat{\Gamma}_n + A \geq k, \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k \right] = P \left[ \hat{\Gamma}_n + A \geq k | \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k \right] P \left[ \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k \right]$$

(3.30)

By applying the definition of total probability, $P \left[ \hat{\Gamma}_n + A \geq k | \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k \right]$ can be solved by

$$P \left[ \hat{\Gamma}_n + A \geq k | \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k \right] = \sum_{i=0}^{k} P \left[ \hat{\Gamma}_n + A \geq k | \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k, \hat{\Gamma}_n = i \right] P \left[ \hat{\Gamma}_n = i | \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k \right]$$

(3.31)

where $P \left[ \hat{\Gamma}_n = i | \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k \right]$ is solved by

$$P \left[ \hat{\Gamma}_n = i | \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k \right] = \frac{P \left[ \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k | \hat{\Gamma}_n = i \right] P \left[ \hat{\Gamma}_n = i \right]}{P \left[ \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k \right]}$$

(3.32)

Given that $\hat{\Gamma}_{n+1} = \hat{\Gamma}_n + A + B = \hat{\Gamma}_n + \Gamma$, it is clear that $P \left[ \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k | \hat{\Gamma}_n = i \right]$

$$P \left[ \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k | \hat{\Gamma}_n = i \right] = P \left[ \hat{\Gamma}_n + A + B > k | \hat{\Gamma}_n = i \right] = P \left[ i + \Gamma > k \right]$$

(3.33)

Applying the result of Equation 3.33 to Equation 3.32 yields the simplified result

$$P \left[ \hat{\Gamma}_n = i | \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k \right] = \frac{P \left[ i + \Gamma > k \right] P \left[ \hat{\Gamma}_n = i \right]}{P \left[ \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k \right]}$$

(3.34)

The expression $P \left[ \hat{\Gamma}_n + A \geq k | \hat{\Gamma}_n \leq k, \hat{\Gamma}_{n+1} > k, \hat{\Gamma}_n = i \right]$ can be simplified to
\( P[\Gamma_n + A \geq k | \Gamma_n \leq k, \Gamma_{n+1} > k, \Gamma_n = i] = \frac{P[i + A \geq k | i + \Gamma > k]}{P[i + \Gamma > k]} = \frac{P[i + A \geq k]}{P[i + \Gamma > k]} \) \((3.35)\)

By collecting terms, the expression \( P[\Gamma_n + A \geq k, \Gamma_n \leq k, \Gamma_{n+1} > k] \) can be simplified to

\[
P[\Gamma_n + A \geq k, \Gamma_n \leq k, \Gamma_{n+1} > k] = \sum_{i=0}^{k} \left( \frac{P[i + A \geq k]}{P[i + \Gamma > k]} \right) P[\Gamma_n = i] = \sum_{i=0}^{k} P[i + A \geq k] P[\Gamma_n = i]
\]

\((3.36)\)

Equation 3.36 is valid for \( n > 0 \) since, for the \( n = 0 \) term, the packet in question is not \( A \), but \( \bar{A} \), requiring a modified expression for the \( n = 0 \) term. Given this modification and the result derived in Equation 3.36, Equation 3.26, the probability that the slot \( k \) is occupied after a collision given that slot \( x \) in the packet is where the collision occurred is then

\[
P[\Psi(k) | \Psi(0)] = \sum_{x=0}^{\max(A)} \sum_{i=0}^{k} P[\bar{A} \geq k - i] P[\Gamma_n = i] + \sum_{n=1}^{\infty} \sum_{i=0}^{k} P[A \geq k - i] P[\Gamma_n = i] P[\Theta = x]
\]

\((3.37)\)

### 3.2.2 Collision Awareness Example

Solving \( P[\Theta = x] \) in Equation 3.26 depends on the random variable that describes packet length in \( N_1 \). However, based on the example shown on this paper, a solution for \( P[\Theta = x] \)
can be attained by first determining the probability that the slot with which $N_2$ has collided has length $y$.

The random variable $A$ returns values from the set of values $S$, where in the example shown, this set includes $\{3, 4, 5, 6\}$. Given that each packet size can occur with equal probability, then

$$P[\text{length of packet in collision}=y] = \begin{cases} \sum_{y \text{ belongs to } S} \frac{1}{\text{all values of set } S}, & y \text{ belongs to } S \\ 0, & \text{otherwise} \end{cases}$$

(3.38)

Given that the probability that the slot with which $N_2$ has collided is length $y$ is calculated, then, collecting the analysis to this point, the probability that the collision occurred with slot $\Theta$ is then determined as

$$P[x = \Theta] = \sum_{i = \min(A)}^{\max(A)} \frac{P[\text{packet length}=\Theta]}{i}, i \geq \Theta, \Theta > 0$$

(3.39)

Evaluating $P[\Psi(k) | \Psi(0)]$ using the same traffic model used in Section 3.1.2 yields the probability of collision for $N_2$ as a function of time after the last collision, seen in Figure 3.6. A Monte Carlo simulation was performed to validate the analytical result of $P[\Psi(k) | \Psi(0)]$. As seen in Figure 3.6, simulation matches the analytical results.

As seen in Figure 3.6, the probability of collision during retransmission can be lowered by a traffic-aware system, where the probability of collision can be lowered from 0.47 to 0.36, increasing efficiency by 30%. An algorithm can be created that will take advantage of this decrease in probability of collision. The next chapter discusses this algorithm and presents the results of simulations that provide a proof-of-concept.
3.3 Summary

This chapter presented an analysis that describes the state of a communications link given an on/off traffic source transmitting data over that link. A problem that arises from such an analysis is that, when a system is observing another system’s traffic, there is no common starting point, leading to two systems that are not synchronized. The concept of using collisions as a synchronization method between two systems is introduced in this chapter. This system synchronization is analyzed, providing a closed-form expression describing the likelihood that a system observation will lead to a busy/idle observation of a link given an on/off traffic source. Given this synchronization/link prediction concept, the foundation is created for an algorithm that can be used to reduce the interference created by the collision of traffic between two different systems; this algorithm is presented in the next chapter.
Chapter 4

Coexistence Mechanism

Given the analysis showing a statistically optimum retransmission time, a coexistence mechanism that can be implemented as either a backoff algorithm or a traffic conditioning algorithm. When implemented in the MAC layer, such an algorithm is a backoff algorithm, while when it is implemented between the Network and MAC layers, it is considered a traffic conditioning algorithm. The following sections describe this algorithm and show simulation results using a traffic model similar to the Poisson traffic model.

4.1 Collision Avoidance

Evaluating $P[\Psi (k) | \Psi (0)]$ using the traffic model used as the example in this paper, the curve seen in Figure 3.6 has a minimum of 0.33 at $k = 4$, which converges to 0.50 as the time elapsed after the most recent collision approaches infinity. Using these results, a traffic-aware system can obtain an improvement in the probability of collision on the next transmission of close to 50% as compared to a traffic-unaware system.

$P[\Psi (k) | \Psi (0)]$ takes into account the statistics of the packet length and describes the probability that there will be a collision at any time $k$ after the most recent collision. Given
Figure 4.1: Layout of network negotiation structure

$P[\Psi(k) | \Psi(0)]$, a statistically-optimum time for retransmission can be generated. Using this assumption, a simple algorithm to exploit this behavior can be designed.

As a starting point for the algorithm, assume that the queue in Figure 4.1 contains a send timer that is set to zero ($\tau = 0$)

Table 4.1: Statistical-Backoff Algorithm

- Wait until $\tau = 0$
- Send packet at the head of the queue
- Wait for ACK/NACK from receiver
  - if ACK: Reduce $\tau$ by one (set to zero if negative)
  - if NACK: Set $\tau$ to optimum point in $P[\text{collision at } k]$ and restore sent packet to the head of the queue
- Go to first step

This algorithm is intended to minimize the probability of collision at retransmission time by using $P[\Psi(k) | \Psi(0)]$ as a guide. The value of $\tau$ is reduced by one after every successful transmission, since optimum values of $P[\Psi(k + x) | \Psi(0)]$ are less apparent as $x$ increases,
converging on a specific value as \( x \) approaches \( \infty \). By reducing the wait time \( \tau \), the throughput of the system is increased while at the same time avoiding the subsequent local maxima that are inherent to the next set of arrivals.

This algorithm was tested in simulation against two other algorithms:

- The persistent algorithm, which is used in some link formats specified in the Bluetooth standard.
- The exponential-back-off algorithm (variant of algorithm used by IEEE 802.11b).

Even though a single set of distributions for the traffic was used to test these systems, a different set of attributes was used for each set of simulations. In these simulations, network \( N_1 \) is assumed to be blind, where it does not see the behavior of \( N_2 \). The traffic generated by \( D_1 \) consists of packets whose length is \( l_1 \) or \( l_2 \) with equal probability; the idle time between packets is generated by an exponential random variable, where the control parameter is \( \lambda \). Note that this traffic model is close to the Poisson traffic model, the only difference being that, unlike a Poisson model, the model used in this set of simulations suspends the arrival random variable until the current packet is serviced on the channel. Also note that an exponential model for traffic is not an accurate model for traffic in real systems; it was selected for its ability to provide a common baseline.

In these simulations, throughput is defined as the number of successful transmissions per unit time. Efficiency is defined as power efficiency, or the expected amount of power required to successfully transmit a single packet.

### 4.1.1 Case 1 - Short Packets with Maximum Throughput

In this case, \( l_1 = 3 \), \( l_2 = 4 \), and \( \lambda \) spans the range \([4, 13]\) for \( D_1 \). \( D_4 \) is attempting to send as much data as possible using packets of length \( l = 1 \). The throughput of each system simulated, following a statistical back-off method, exponential back-off method, and persistent/zero back-off method respectively, is seen in Figure 4.2. The efficiency of each of
these systems is seen in Figure 4.3.

Figure 4.2: Simulated throughput for case 1

As seen in of Figure 4.3, the statistical back-off method shows an improvement in power efficiency of roughly five percent over either the persistent or the exponential-backoff mechanisms. Also, as seen in Figure 4.2, this increase in efficiency comes at little cost in maximum throughput, when the statistical-backoff method is compared with the persistent method. It should also be noted that the throughput of the exponential-backoff system is significantly lower than either the persistent or the statistical-backoff methods.

4.1.2 Case 2 - Long Packets with Maximum Throughput

In this case, \( l_1 = 8, l_2 = 9 \), and \( \lambda \) spans the range \([8, 17]\) for \( D_1 \). As in seen in Section 4.1.1, \( D_4 \) is attempting to send as much data as possible using packets with \( l = 1 \). Systems with configurations similar to Section 4.1.1 are seen in Section 4.1.2, where the throughput of each system is seen in Figure 4.4 and the efficiency of each of these systems is seen in Figure
As seen in Figure 4.5, the statistical back-off method shows an increase in efficiency of around five percent over the exponential back-off method and around ten percent over the persistent method. However, as seen in Figure 4.4, the statistical back-off supports a throughput that is significantly lower than the persistent method. This difference in throughput is due to the difference in scales between $N_1$ and $N_2$. Since the wait time that is statistically optimum is a function of, among other factors, the length of the packets in $N_1$, and since the packets in $N_1$ are significantly longer than in $N_2$, $N_2$ needs to wait a length of time to transmit that is significantly longer in case (b) when compared to case (a), where the difference between $N_1$ and $N_2$ in packet length is not as large. It should also be noted that the statistical back-off method supports a throughput that is significantly higher than the exponential back-off method.
Figure 4.4: Simulated throughput for case 2

Figure 4.5: Simulated efficiency for case 2
4.1.3 Case 3 - Long Packets with Variable Data Traffic Rate

In this case, just like in Section 4.1.2, $l_1 = 8$, $l_2 = 9$, and $\lambda$ spans the range $[8, 17]$ for $D_1$. Also like in Section 4.1.2, $D_4$ transmits packets $l = 1$. However, variable data traffic rate is used for $D_4$, with idle time between packets following the same pmf used in $D_1$, with $D_{4, \lambda} = D_{1, \lambda}$, such that all networks in the local area have the same idle time statistics. The throughput and efficiency of each simulated system are seen in Figures 4.6 and 4.7, respectively.

![Figure 4.6: Simulated throughput for case 3](image)

In this case, as in the case in Section 4.1.2, the statistical back-off method shows an increase in efficiency over the persistent system of around ten percent, as seen in Figure 4.7. The increase in efficiency of the statistical back-off method over the exponential back-off method is approximately five percent for the heavier-loaded systems approximating the exponential back-off efficiency as the load on the system decreases. Figure 4.6 shows that the exponential back-off method has a throughput that is significantly lower than either the persistent or statistical back-off methods. For the configurations with lighter loads, the persistent system
Figure 4.7: Simulated efficiency for case 3

performs better than the statistical back-off method, but this difference decreases significantly as the load on the system decreases, manifested as longer mean idle times between packets.

4.2 Discussion

The statistical back-off method was shown to have a packet efficiency that is systematically higher than either the persistent or exponential back-off methods. This increase in efficiency was shown to generally lie between five and ten percent. At the higher traffic loads, where the best-effort or exponential back-off methods perform at a level of efficiency of roughly forty percent, the statistical method showed efficiency levels closer to fifty percent, or roughly 25% better. Since battery consumption in most mobile wireless communications devices is dominated by the transmit power amplifier, this increase in efficiency can be compared to an increase in battery life of up to twenty-five percent.
The throughput of the statistical back-off method was shown to be systematically lower than the persistent system. However, under certain conditions this decrease in performance is very small, as seen in Section 4.1.1. Also, this decrease in throughput was shown to decrease significantly as the system load decreased, as seen in Section 4.1.3. It should be noted that the exponential back-off method performed poorly in all cases.

4.3 Summary

This chapter presented a simulation study of the coexistence of two different networks given three basic backoff schemes, a persistent algorithm, an exponential backoff algorithm, and the statistical backoff algorithm based on the analysis presented in the previous chapter. This study was performed on a generic network, providing some general insight into the problem. However, this insight does not necessarily imply that the algorithm presented in this dissertation will function properly when implemented over a real network protocol. The case study selected for this dissertation is the Bluetooth/IEEE 802.11b coexistence problem, so a strategy is necessary to implement this coexistence algorithm into existing protocols. Before the algorithm can be implemented over a system, a strategy for integrating the algorithm into an existing protocol. The next chapter presents such a strategy, which in this case is backwards-compatible, allowing the algorithm presented in this dissertation to be implemented in a Bluetooth device that is fully-compliant with the Bluetooth version 1.1 release.
Chapter 5

Integration

In the previous chapter, the coexistence mechanism was shown to function when applied to a generic system. This demonstration, while providing some insight, does not prove that the algorithm is practical. To show that the coexistence algorithm presented in this dissertation is practical, it needs to be applied to a set of protocols that exist. One of the principal problems in the integration of new algorithms into the framework of an existing standard is that the integration should not require the modification of the standard. In this chapter, I present a method for integrating some types of algorithms into Bluetooth without requiring the modification of the existing protocol.

Furthermore, when integrating performance enhancements into a standard, it is important to evaluate the impact that this integration will have on the overall system performance. When placed stand-alone, some enhancements will yield performance increases that are significant, yet when placed in the context of the system as a whole, these enhancements could decrease performance rather than increase it [58].
5.1 Transparency

In a layered system, transparency means that changes performed at a layer where the enhanced functionality exists is transparent to the layers both above and below the layer in question. Since systems rarely follow the strict layering structure, the layering example is an oversimplification, but it illustrates the point. In reality, transparency means that a specific module or set of modules can be isolated as a black box, such that the input and output sets to which the module has dependencies match the expected value, yet the method in which those values are achieved incorporates the desired performance improvements.

Bluetooth contains an advantage over IEEE 802.11b, the existence of the AUX1 packet, which is not subject to the ARQ algorithm that exists in ACL packets. The AUX1 packet has the same general structure as the DH1 packet, with the exception that it does not contain a CRC code, hence it cannot support an ARQ protocol. This lack of ARQ gives the application complete control over the timing of the packet transmission, as long as it abides by the Bluetooth slotting requirements.

The primary benefit of the AUX1 packet is that it is essentially a raw Bluetooth packet, with errors in reception propagating to the upper layers. Hence, these packets provide the application with significant levels of control over the link.

Bluetooth data packets, known as Asynchronouns Connection-Less (ACL), use ARQ to guarantee the accurate delivery of a packet. The ARQ scheme in Bluetooth is based on re-transmissions immediately after the receipt of the corrupted packet. The benefit of AUX1 packets is that they have no ARQ. If accurate data delivery is desired, the ARQ mechanism can be implemented at the application level, above the HCI. In this case, the timing of the retransmissions can be controlled by the application to whatever arbitrary position is desired, as seen in Figure 5.1. This capability allows the bypassing of aspects of the Bluetooth implementation such as FEC and timing while maintaining the more desirable Bluetooth attributes, such as its addressing and link management structure.
One sample implementation using these AUX1 packets is discussed in this chapter, a Hybrid ARQ implementation. The Hybrid ARQ implementation is presented as a demonstration of capability beyond the control of timing; in this case, the control of FEC. The use of AUX1 packets becomes directly relevant to the core of this dissertation in the next chapter, where this type of packet is used to implement the coexistence algorithm presented in this dissertation.

### 5.1.1 Hybrid ARQ

In traditional ARQ, the packet is re-transmitted when it received incorrectly. This approach is reasonable when a corrupted packet is expected to contain a very large number of errors or is expected to be lost altogether. In cases where packets are lost altogether, the only viable solution is to re-transmit the lost data. However, in cases where a certain number of errors is expected an FEC code can be added to the link. In such cases, the probability that the packet is unrecoverably corrupted is reduced. However, the introduction of such codes limits the environments in which the system can efficiently operate. If for any reason, such as a change in position, the average SNR increases in the link, the use of a FEC will hinder throughput, since it will incorporate added redundancy that will generally not be needed.
Therefore, traditional ARQ algorithms, even when incorporating FEC, can be bandwidth inefficient.

A hybrid ARQ algorithm, based on non-Bluetooth work presented in [59], is used as a demonstration algorithm in this chapter. This algorithm has the added advantage that it is blind. Blind adaptive algorithms allow a system to respond to changes in the environment without requiring the use of a channel estimation technique. The elimination of the channel estimation requirement eliminates a source of uncertainty or error from the algorithm implementation.

Algorithm Overview

A graphical version of the algorithm described above is seen in Figure 5.2. Note that added redundancy is sent only when it is needed, hence at no time is channel estimation performed, thus this algorithm can be considered to be a blind algorithm.

To implement the blind adaptive ARQ algorithm, a block size of 61 packets was selected. The reply packet selected is a (15,10) shortened Hamming code, since it is relatively easy to implement and since Bluetooth already implements this code in one of its packet types. Note that this FEC code is decoded by the application, so any FEC could have been selected for the reply section of the communications link. For the blind adaptive ARQ case, a (511,421) BCH code was chosen. Since this code was applied over Bluetooth packets whose payload is 240 bits, the code was truncated to (330,240). This code was chosen since its rate is approximate to Bluetooth’s 2/3 rate FEC used in DM1 packets, providing some similar basis for comparison. A 16-bit CRC sequence was selected to perform error detection on received packets. This CRC sequence was selected since this is the CRC used by Bluetooth systems.
Table 5.1: Bluetooth Hybrid ARQ

- Data is partitioned into blocks of packets, where each block contains N packets.
- A cyclic redundancy code is added to each packet such that errors can be detected when the destination node receives the packet.
- Using a systematic algebraic code, a redundant set of data is created for each of these packets and stored at the source node.
- The block of packets is then sent from the source node to the destination node (without the redundancy but with the CRC code).
- The destination node performs the CRC check on each of these packets and compiles a list of the packets that have been received incorrectly.
- The destination node then creates a single packet with this list of erroneous packets and sends this list to the source node. The delivery of this list is guaranteed through the use of the traditional ARQ algorithm, where the packet is retransmitted until the source node acknowledges its receipt.
- The source node then assembles the redundant data generated prior to transmission for each of the packets that was received erroneously by the destination node; the source node then sends this redundant data. Note that the redundant data will generally be considerably shorter than the data set for which it corresponds.
- After reception of the redundant data, the destination node applies the FEC code to each of the erroneously received packets, compiling a list of the packets cannot be corrected by the added redundant data.
- The destination node compiles a list of the packets that remain uncorrected and sends the list to the source node.
- The source node assembles the packets that could not be corrected on places them at the beginning of the next block of data to be sent to the destination node.
- The algorithm is started again by creating a new block of data.
Figure 5.2: Hybrid ARQ Algorithm
Simulation Validation

Given that a simulation of a Bluetooth system is used to test the Hybrid ARQ scheme, the simulation needs to be validated first. An analysis equivalent to the one presented in this section was first published in [60]. Assuming that no FEC is used on the packet and that bit errors are independent, the payload probability of error can be given by

\[ p_p = 1 - (1 - p_e)^l \]  

(5.1)

Where \( l \) is the length of the payload in bits, including a CRC block to detect errors. The next two parts of the packet that need to be analyzed are the packet header and the synchronization word. The packet synchronization is performed with a sliding correlator, which means that the peak threshold for detection needs to be set such that there is a minimized probability of false alarm. In the case of this set of simulations, a threshold of 7 bits out of the 72-bit synchronization word was selected. This selection was based on a probability of false alarm that is several orders of magnitude lower than the probability that there will be an error in the packet payload. This selection leads to the probability that there is a packet synchronization error of

\[ p_s = \sum_{k=0}^{7} \frac{72!}{k!(72-k)!} p_e^k (1 - p_e)^{72-k} \]  

(5.2)

The packet header is composed of 18 triplets of bits, since the Bluetooth packet uses a \( r = 1/3 \) repetition code. Therefore, the probability that a packet header error occurs is then

\[ p_h = \left( 3p_e (1 - p_e)^2 + (1 - p_e)^3 \right)^{1/8} \]  

(5.3)

The frame error rate of this link can then be described as

\[ FER_{\text{Bluetooth}} = 1 - \left( (1 - p_p) (1 - p_h)^2 (1 - p_s)^2 \right) \]  

(5.4)
The influence of both the header and synchronization sections need to be squared, since the return packet containing the acknowledgment will occupy the length required to transport just the packet header and synchronization word.

The probability mass function describing the probability that \( n \) retransmissions will be required in this link can then be calculated as

\[
pmf_n = FER_{\text{Bluetooth}}^{n-1} (1 - FER_{\text{Bluetooth}})
\]  

(5.5)

Given \( pmf_n \), the expected number of total packets transmitted over the link per unit time can be calculated as

\[
T = \sum_{i=0}^{\text{max}(n)} i \cdot pmf_n
\]  

(5.6)

However, the expected number of correct packets over the link per unit time can be calculated as

\[
C = \sum_{i=0}^{\text{max}(n)} pmf_n
\]  

(5.7)

Thus, the expected throughput for the system is

\[
\Theta|_{pe} = \frac{C \cdot l \cdot l_{CRC} \cdot l_{s+h}}{T \cdot \tau}
\]  

(5.8)

Where \( l_{CRC} \) is the CRC length in bits, \( l_{s+h} \) is the combined length of the header and the synchronization word in bits, and \( \tau \) is the duration of the packet.

This expected throughput was compared with the Bluetooth throughput as a function of distance when performed in simulation, resulting in the same value for the whole range, validating the Bluetooth simulation.
Figure 5.3: Blind Adaptive ARQ applied to Bluetooth system compared with DH1 and DM1 packets (AWGN)

**Performance Comparison**

The Hybrid ARQ algorithm was implemented using the AUX1 packets. The performance of this algorithm was compared to the basic Bluetooth system using DM1 and DH1 packets. Figure 5.3 shows a graph of this comparison.

As seen in Figure 5.3, the Hybrid ARQ algorithm provides performance that is comparable to the non-FEC version of Bluetooth (DH1) at the high SNR levels and performance comparable to the FEC version of Bluetooth (DM1) at the low SNR levels. This performance enhancement is implemented using a backwards-compatible implementation, making it fully compliant with the Bluetooth standard. This ability to implement a backwards-compatible enhancement to the Bluetooth link will be leveraged in the next chapter, Chapter 6, to implement the coexistence algorithm presented in this dissertation.
5.2 Summary

This chapter introduces a method that can be used to integrate some types of MAC algorithms into a Bluetooth device that allows the device, with the modifications included, to still comply with the Bluetooth version 1.1 specifications. This method is used in the next chapter to integrate the coexistence algorithm presented in this dissertation with a simulated Bluetooth device. The next chapter also discusses the rationale behind concentrating on the Bluetooth side of the coexistence problem for a solution and it also presents a thorough discussion of the coexistence algorithm presented in this dissertation as well as the implementation of an adaptive frequency hopping algorithm for Bluetooth.
Chapter 6

Practical Case Study

6.1 Introduction

The coexistence problem so far has been address as a general problem, where the type of networks that are required to coexist has not been an issue. The primary benefit of this approach has been that the concept of enabling the coexistence between two different systems can be proven in a generic fashion. One of the drawbacks of the general approach is that questions remain about the algorithm’s viability when applied to real, or at least realistic, systems.

To demonstrate the viability of a collision-based multiple access (CBMA) strategy, two protocols were selected, Bluetooth and IEEE 802.11b. The primary reason why these two systems were selected to demonstrate the algorithm’s viability is that these two systems will most likely represent the bulk of the network hardware that will be deployed in the near future where a coexistence question could arise. Both Bluetooth and IEEE 801.11b devices are capable of supporting data traffic making both suited for several home and office applications. While both standards have applications that are well suited for the same environment, those applications generally do not overlap. For example, a PDA is
most likely to use Bluetooth for short-range communication with another device such as a desktop computer or a printer, while general-purpose computers are most likely to use IEEE 802.11b for most data applications when a wireless connection is desired and Bluetooth for a limited suite of applications, such as synchronization with PDAs. While Bluetooth and IEEE 802.11b have the technical ability to support a variety of applications, their acceptance in the marketplace, especially for Bluetooth, is an issue to be addressed.

The widespread acceptance of IEEE 802.11b as an access solution has been proven, with the shipment of 6 million devices in 2002 growing to an expected 33 million devices shipped in 2006 [61]. Bluetooth has also shown significant growth, with 35 million units shipped in 2002 and an estimated 510 million units to be shipped in 2006 [62]. The number of device solutions available has consistently increased, with the announcement of products based on Bluetooth technology such as Microsoft’s Bluetooth keyboard and mouse [63]. Given Bluetooth’s relaxed specifications, the desired price point of approximately $5 per device is more than likely an achievable goal, allowing the projections likely to be achieved. For inexpensive, short-distance applications, Bluetooth will most likely become the solution of choice for many appliance manufacturers. Given IEEE 802.11b’s success in the marketplace and the continuous improvement in manufacturing efficiency of Bluetooth devices, these two systems will most likely be the first systems that will encounter significant coexistence issues.

6.2 Approach

It is generally relatively straightforward to achieve some desired behavior from a system when the standard is open for modification. For example, an adaptive frequency hopping architecture can be applied to Bluetooth systems for all packet types as long as the standard is open to modification. However, if the standard can be modified, there is no reason why the designer should be constrained to just adaptive frequency hopping, since there is nothing that makes the frequency hopping algorithm more susceptible to modification than another
aspect of the standard, such as the multiple access technique. If a designer is willing to accept
the modification of the hopping structure of a standard, there is no compelling reason why
that designer should not be able to modify the standard such that a carrier sense algorithm
is implemented.

In this dissertation, the goal is to minimize the amount of modification that is required from
a standard. By taking this approach, the main problem of coexistence can be addressed
directly, where the problem statement can be simplified to two current implementations.
The approach selected in this case study is to implement the algorithm on the Bluetooth
side of the system. There are three basic reasons why the Bluetooth side was selected: insertion interface, susceptibility to errors, and use paradigm.

6.2.1 Insertion Interface

One of the key advantages of Bluetooth is that it contains the ability to bypass much of
the MAC functionality that is inherent to most types of systems, including most of the
Bluetooth packet formats. Bluetooth DHn and DMn packets allow the user to transmit
asynchronously, but they contain an ARQ mechanism, thus limiting the application’s ability
to control the timing of the device. The SCO packets in Bluetooth allow errors to propagate
to the upper layers, but they leave the application with no control over the transmission
timing. The AUX1 packet allows the application to control the timing of the packets, within
the framework of the packet slots, yet errors are allowed to be propagated up the protocol
stack. The previous chapter provides a more thorough analysis of the use of the AUX1
packet for application-level implementations of MAC functionality. IEEE 802.11b, on the
other hand, contains no such packet. The application is given very little control over the MAC
of the system. While traffic conditioning is possible in IEEE 802.11b, the level of control
is too coarse for the purposes of this application. Thus, given AUX1 packets, Bluetooth
provides the most adequate insertion point for this algorithm.
6.2.2 Susceptibility to Errors

The IEEE 802.11b system is fairly fragile when placed in an interference-rich environment. The two main reasons for this susceptibility are the multiple access technique and the frame length.

As described in the background chapter, the IEEE 802.11b system uses a carrier-sensing algorithm to first listen to see if any system is currently transmitting. There are two carrier-sensing aspects of IEEE 802.11b, virtual carrier sense and physical carrier sense. Given that the coexistence algorithm is intended to be applied in a mixed Bluetooth/IEEE 802.11b environment, the virtual carrier sense plays no direct role in the coexistence between an IEEE 802.11b link and a Bluetooth link, since the virtual carrier sensing algorithm listens to the traffic being exchanged between other devices and can predict when the channel will be idle. For a review of IEEE 802.11b’s virtual carrier sense algorithm, see the discussion in the IEEE 802.11b RTS/CTS packet exchange description in Chapter 2. Obviously if Bluetooth is transmitting, IEEE 802.11b will be unable to decode the Bluetooth frames, so it has no information for the virtual sensing algorithm. In the case of two individual links, the physical carrier sense algorithm is the one that plays the key role, since it can detect a Bluetooth transmission, triggering the behavior outlined in the background chapter of this dissertation. Bluetooth, on the other hand, has no carrier sense algorithm, so when it arrives at the slot in which it is meant to transmit, it will transmit, regardless of the activity in the local environment.

The second reason why IEEE 802.11b is more susceptible is that it has significantly longer frames than Bluetooth. As discussed in the background chapter, IEEE 802.11b has frames that are composed of a physical layer convergence protocol (PLCP) initial section to each packet of length 192 bits at 1 Mbps, yielding a PLCP time length of 192 $\mu$s. The MPDU (medium access control - MAC - protocol data unit) of maximum length 18768 bits at 11 Mbps (for the high-speed variant), yielding a MPDU time length of 1706.2 $\mu$s, yielding a total packet time of 1898.2 $\mu$s. Of course, this packet will be followed by an acknowledgement,
which will also contain a 192 µs PLCP, but where the MPDU will occupy 112 bits at 11 Mbps, yielding a total packet time of 202.2 µs. The data packet and the acknowledgement will be separated by the short inter-frame spacing (SIFS), which is 10 µs. When combining the data packet, SIFS, and acknowledgement, the maximum total exchange length of an IEEE 802.11b packet is 2110.4 µs. This packet time can be compared with the Bluetooth packet exchange time of 366 µs for the packet, a 259 µs inter-packet space, and another 366 µs packet containing the acknowledgment to the original packet plus any data on the return path - note that this packet will always be 366 µs long, even when no data needs to be transmitted. The total exchange time between two Bluetooth devices is 991 µs. On a time scale, the IEEE 802.11b packet can be twice as long as the Bluetooth packet. However, the amount of data being transported by each of these packets is significantly different, yielding significant levels of performance.

Of interest in this case study is the AUX1 Bluetooth packet, so all calculations are performed with this packet in mind. For a Bluetooth packet to be correct, the access code, header, and payload need to be correct. This analysis was presented in [60], in which the dissertation author is listed as a co-author. The preamble, which contains the synchronization sequence, contains a sequence whose Hamming distance is such that it is tolerant to up to six bit errors over the 72 bits. The probability that the access code is received incorrectly is then

$$P[A] = \sum_{k=0}^{72-6} \binom{72}{k} (\epsilon(\gamma))^k (1 - \epsilon(\gamma))^{72-k}$$

(6.1)

where $\epsilon$ is the bit error probability at SNR $\gamma$. The Bluetooth header is made up of 18 data bits, each of which is protected using a $r=1/3$ repetition code. Given that decoding of these packets can be performed using a simple voting mechanism, the whole header will be in error when two or more errors occur in any one triplet of the header. Therefore, the probability that the header is incorrect can be described as

$$P[H] = \left(3\epsilon(\gamma)(1 - \epsilon(\gamma))^2 (1 - \epsilon(\gamma))^3\right)^{18}$$

(6.2)
Finally, the payload of the packet is the most vulnerable part of the whole packet, since, in the AUX1 case that is being examined any error in the payload will cause the packet to be decoded incorrectly. In this case, the payload contains 240 bits, thus the probability that the payload of the packet is incorrect can be described as

\[ P[H] = (1 - \epsilon(\gamma))^{240} \]  

(6.3)

In a Bluetooth AUX1 exchange, the probability that a transfer of 240 bits from one device to the next is then

\[ P[\text{Correct}] = 1 - P[A]^2 P[H]^2 P[B] \]  

(6.4)

Note that both the access code and header need to arrive correctly twice, since the return packet from the destination device will contain the acknowledgement, which has to arrive correctly. While the access code and header need to arrive correctly, the payload of the return path is not relevant to this analysis, since the AUX1 packet provides sufficient flexibility such that the application can decide to ignore the content of the payload of the return packet in the evaluation of the data sent to the destination device, since its correctness is independent of the correctness of the data that was sent from the source to the destination.

IEEE 802.11b packets are structured different from Bluetooth packets. As described in the background chapter, the IEEE 802.11b packet is composed two primary sections, the PLCP and the MPDU. The PLCP is not protected by any data structures, so the probability that the PLCP is incorrect is then

\[ P[PLCP_h] = (1 - \omega(\gamma_l))^{192} \]  

(6.5)

where \( PLCP_h \) is the PLCP header and preamble, \( \omega \) is the probability of bit error for an IEEE 802.11b bit and \( \gamma_l \) is the SNR for the low data rate part of the IEEE 802.11b frame.
The PPDU of the IEEE 802.11b frame generally has a far larger number of bits than the PLCP header, and it is also higher data rate, which for the same transmit power means that it has a lower SNR. Therefore, the probability that the payload is received correctly is then

\[ P[PSDU] = (1 - \omega(\gamma_k))^{18768} \]  \hspace{1cm} (6.6)

Given Equations 6.1-6.6, it is then possible for a given SNR to calculate the relative probability that a transaction is received incorrectly. As seen in Figure 6.1, the probability that a Bluetooth packet is incorrect is significantly lower than for IEEE 802.11b for any given SNR.
6.2.3 Use Paradigm

Bluetooth and IEEE 802.11b, while both packet-based systems that allow asynchronous operation, are used for fundamentally different tasks. IEEE 802.11b is used for applications such as web browsing, which can involve the transfer of large amounts of information, and email applications, which can not only be sizeable, especially when attachments are involved, but it can be used without the user’s knowledge, since most email clients are configured to check the email server at regular intervals. Bluetooth, on the other hand, will probably be used mostly for small applications, such as synchronization of smaller devices and connections with relatively low-bandwidth appliances, such as keyboards and mice [63]. Furthermore, most transfers with Bluetooth will be generally local, not to some remote server on the Internet.

The nature of the Bluetooth usage, coupled with Bluetooth’s natural low data rate, mean that Bluetooth will generally be used to transfer small amounts of data. Given the relatively low data rate expectations of Bluetooth, the penalty required in Bluetooth throughput to support the use of algorithm described in this dissertation will have little impact, since the transfer is small to begin with. Furthermore, since the transfers will be generally local, and given that the algorithm that is presented in this paper injects an artificial latency in the packet transfer, there is a lower likelihood that the packet latency will result in a TCP timeout.

TCP (transfer control protocol), is the Internet Protocol’s (IP) method to guarantee transport delivery of data. IP provides a routing mechanism between different points in the Internet. Using the address destination in a packet as a guide, IP is used to tell the routers where to send packets once they are received. IP provides no guarantees for delivery and, when packets are delivered, IP does not guarantee that the packets are in order. To solve this problem, packets can be sent using TCP as the transport protocol. TCP creates a window structure, seen in Figure 6.2, where packets that are received are counted. Once this window fills up, no new packets are sent until the destination device has acknowledged the missing
packet. The size of this window can be managed such that congestion in the network, the primary source of packet loss in a wired network, can be minimized. The problem with latency in a link over the packet path that is not related to congestion is that it can trigger a TCP window change. This window change, while designed to deal with network congestion, would have the undesired effect of unnecessarily reducing throughput.

If the final link in the connection injects a delay, the likelihood of a timeout grows significantly, since latency beyond that experienced at the final link will be incurred. On the other hand, if the connection is performed over a single link, there is only one source of delay in the whole transmission, and the likelihood of timeout is significantly reduced because there is a single, controlled, source of delay.

### 6.3 Case Study Scenario

Bluetooth and IEEE 802.11b are the protocols selected for this case study. The scenario that was selected for this case study is based on the expected usage of these devices. Bluetooth
devices are expected to operate over short ranges. The exact service that these devices are expected to support is difficult to determine, since Bluetooth devices are expected to support a wide variety of applications. However, the one common trait among Bluetooth applications is that they will generally be local, such as a keyboard, mouse, PDA synchronization, or some other form of local update. Given that the application is expected to run locally, where both the client and the server are located locally, then it is unreasonable to expect there to be much separation between packets. Therefore, the traffic for the Bluetooth piconet in this case study was modeled to be an infinite source, where a packet is always ready for transmission.

IEEE 802.11b devices, on the other hand, are designed to support networking applications such as web browsing, file transfers using applications such as ftp (file transfer protocol), and email applications. Note that in this case study, the IEEE 802.11b device was assumed to be operating in the 11 Mbps mode. The reason why this speed was selected is that it should provide the highest performance for the link, in terms of throughput in low-SNR cases. The IEEE 802.11 specifications do not specify the rate switching criteria. This open part of the standard adds significant ambiguity to the model, so it was deemed that at this stage, this extra dimension should be outside the scope of the dissertation. However, since a switch in data rate would effectively change the relative duration of data bursts, this study includes a scenario where longer-than-expected “on” bursts are seen in the channel. This scenario provides insight into how the algorithm presented in this dissertation may function when the system changes its traffic characteristics.

Given that the IEEE 802.11b link covers the last trunk of a data transfer spanning multiple hops over the Internet, a model that is based on measurements can be used. As described in the background chapter, self-similar traffic models have received considerable attention when modeling traffic supporting a large number of user sessions. The on-off source selected has off periods described by an exponential distribution whose probability distribution function is defined as:
The on periods of the traffic source are described by a Pareto distribution, which is defined as

\[ f_p(t) = \alpha \cdot \beta^\alpha t^{-(\alpha+1)} \]  

(6.8)

The selected \( \alpha \) for this distribution is 1.7, which, as mentioned in Chapter 2, can be used to build a web traffic model. The selected \( \beta \) is 1. Given selected traffic parameters, the mean size of each data burst is 5.5 kB. This mean burst size was chosen because the scale, discussed in Section 6.3.1, that is visible by a Bluetooth device is 1.25ms long, and it was desired for the Bluetooth observations to seem “bursty.” Such long bursts incur a significant probability of collision, and such a setup can be considered a “worst case” (or at least highly detrimental) for the system, and it was deemed important to test the robustness of the coexistence process presented in this dissertation against such a traffic condition. For the sake of completeness, the performance of the algorithm with a variety of burst lengths, both shorter and longer, is shown in Section 6.5.1.

It should be noted that one of the aspects of wireless communication system simulation that is significantly simpler than wired networks is in the assumption of traffic model. In general, the number of users sharing a specific trunk is low, generally just one, and the number of services that each of these users is limited in both number and complexity, generally because of a small or awkward user interface on the mobile device. Because of these general constraints, the assumptions used in this simulation towards the traffic supported by the systems do not represent a large deviation from the expected behavior of the device.
6.3.1 Scaling

One of the key aspects of the algorithm presented in this paper is that the synchronization between two systems is performed through the observation of errors that are the likely result of a collision between two packets. Because the algorithm is based on the collision between packets from two different sources, by definition the basic unit of time that is available is a single packet length, where the unit of time for each system is each system’s specific packet length. Given that the basic scale for the measurement of a collision is a single packet, the basic unit for the application of the collision avoidance algorithm is a single packet length of the system implementing the avoidance algorithm. Figure 6.3 shows an example of this scaling.

As seen in Figure 6.3, a single packet on one system can occupy several typical packet lengths for another system. As seen from the perspective of the algorithm implementing the collision avoidance algorithm, the traffic pattern can be characterized by binary pattern, either a packet can be detected or not. The collision seen in Figure 6.3, and subsequent packet overlap, can be characterized by the on/off pattern shown in Figure 6.4. Note that in Figure 6.4 there is a state called “undefined.” In this state, the system performing the
Figure 6.4: Overlap On/Off Pattern

detection of collisions is not transmitting a packet, so it cannot detect a collision. The non-detection is neither a positive or negative detection, but instead is a third state identified as “undefined.”

This difference in scale can create complications, namely in instances in which the granularity offered by the collision detection algorithm is not sufficient to show the detail needed to avoid traffic. For example, suppose that the collision avoidance algorithm is implemented on a system where the packets are significantly longer than those of the system whose traffic is the one that is meant to be avoided, as seen in Figure 6.5.

In this case, the off periods need to be excessively long in the system with short packets such that the system that performs the traffic ”measurement” is capable of detecting the off periods, seen in Figure 6.6. When the off periods are not sufficiently long, then the source of interference would be viewed by system 1 as always ”on,” as seen in Figure 6.7.

This difference in scale is another reason why Bluetooth was selected as the system on
Figure 6.5: Insufficient Collision Resolution

Figure 6.6: Long Off Periods for Sufficient Resolution
which to implement the collision avoidance algorithm. In the case of Bluetooth, the basic 
unit is a slot, which is 625 $\mu$s long. However, a transaction between two devices cannot 
be completed until an acknowledgment is received since the integrity of the data cannot 
be verified otherwise. Furthermore, taken into the context of the description of A, B, and 
G, an acknowledgement is no longer a random process, but instead can be considered a 
deterministic aspect, unless the destination was unable to received the packet completely. 
Thus, the length of A in the case of Bluetooth is not 625$\mu$s, but 1250$\mu$s. In this case, then 
the basic granularity of a Bluetooth observation is 1250$\mu$s.

Given that Bluetooth operates in 625$\mu$s packet pairs, the maximum resolution that is possible 
for a Bluetooth link is 1.25ms. IEEE 802.11b frames and waiting periods have features that 
are significantly smaller than 1.25ms. For example, the inter-frame spacing is measured in 
tens of microseconds and frames such as RTS/CTS/ACK are approximately 200$\mu$s long. 
Given this difference in scale, an approximation on the order of milliseconds should be 
sufficient for the purposes that this analysis will be used. Given the coarse requirements of 
the final application, several approximations are made in the following description. When
Given IEEE 802.11b’s variable packet length, the worst-case scenario can be considered to be the lowest granularity of the packet set for description of the random variable \( A \). In the case of IEEE 802.11b, the longest PSDU that is possible for a single IEEE 802.11b PPDU is 18768 bits. This is the length of the MPDU, which includes the CRC for error detection and all overhead incurred by the IEEE 802.11 MAC. Added to these bits is the PLCP, which is 192\( \mu s \) long in the case of the slow version of the IEEE 802.11b 11Mbps specifications. The short version of IEEE 802.11b at 11Mbps, which is not covered in this dissertation, uses 2Mbps DQPSK for part of the PLCP, rendering it slightly shorter than the slow 192\( \mu s \) version. The transaction between two IEEE 802.11b devices is not complete until the packet is acknowledged. The delays incurred by the acknowledgment include the SIFS delay, which is 10\( \mu s \), and the actual acknowledgement, which is a 112 bit MPDU coupled with a 192\( \mu s \) PLCP. Therefore, the total length of time that a single exchange between two devices occupies, signified by \( A \) in Figure 6.8, is

\[
A_{\text{exchange}} = 192 + \frac{18768}{11} + 10 + 192 + \frac{112}{11} = 2.11\text{ms}
\]  

(6.9)

While the maximum aggregate packet length of an IEEE 802.11b device is 2.11ms, the actual exchange may be considerably longer. In the case in which the MPDU that is to be
transported is longer than 18768 bits, a fragmentation operation would ensue, in which the packet is fragmented into a series of payloads that are 18768 bits long, with a trailing packet that carries the remainder of the data. If the original length of data to be transmitted is L bits long, the whole transaction, assuming that no contention mechanisms are active, is then

\[
A_{\text{main}} = A_{\text{exchange}} \left\lceil \frac{L}{18768} \right\rceil + \left( 192 + \frac{\text{mod}(L, 18768)}{11} + 10 + 192 + \frac{112}{11} \right) \quad (6.10)
\]

where \(\text{mod}(.,.)\) is the modulus operation. In the case in which the RTS/CTS threshold is exceeded (in this dissertation, the RTSThreshold was set to zero), an RTS/CTS exchange is required before any transfer of data. The RTS and CTS packets include 160 and 112 bits, respectively, plus the required PLCP. Each RTS/CTS exchange includes a SIFS separation of 10 \(\mu s\) between the RTS and CTS packets as well as the CTS packet and the subsequent data packet, yielding an aggregate length of

\[
A_{\text{RTS/CTS}} = 192 + \frac{160}{11} + 10 + 192 + \frac{112}{11} + 10 = 418.7\mu s \quad (6.11)
\]

Finally, the before the initial transmission is performed, the DIFS waiting period is required, lasting 50\(\mu s\), and the backoff window, which is anywhere between 0 and 20.62ms. The mechanism for managing the contention window as well as a specific breakdown of the different packet types and the handshaking algorithm are included in the background chapter of this dissertation.

In this case study, the RTS/CTS threshold was set to zero. This was done for two reasons. First, the packets are expected to be generally long, on the order of several thousand bits, since the traffic supported is fairly high bandwidth web traffic. Furthermore, the RTS/CTS exchange adds around 420\(\mu s\) to the data packet, which is around 1/3 the minimum granularity of the traffic Bluetooth can observe. Thus, the inclusion or exclusion of the RTS/CTS exchange will have little or no effect on the functionality of the algorithm. Given this constraint, the length of the A random variable in Figure 6.8 when applied to IEEE 802.11b is
then

\[ A_{802} = A_{\text{RTS/CTS}} + A_{\text{main}} \] (6.12)

An approximation of the random variable \( A \) can be made by scaling \( A_{802} \) to the data source’s “on” period, which is approximated by

\[ A_{\text{soln}} \approx A_{802} \times N \] (6.13)

where \( N \) is the mean burst length of the “on” source. The off period of the traffic source chosen for the IEEE 802.11b device is described by the random variable \( B_{\text{off}} \). In the case of this system, the idle time between packets is the data source’s “off” period minus the overhead incurred by the IEEE 802.11b MAC frames that are transmitted (RTS/CTS/ACK/MPDU overhead/MPDU PLCP),

\[ B_{802} = B_{\text{off}} - \text{overhead} \] (6.14)

Note that if the service rate of the channel, or the rate with which the data arriving at the IEEE 802.11b device is successfully sent over the channel, is less than the arrival rate of the incoming data, then the idle time between packets is \( \text{DIFS} + \text{CW} \). This framework provides an approximation of random variables \( A \) and \( B \).

\[
P[\Psi(k)|\Psi(0)] = \sum_{x=0}^{\max(A)} \sum_{i=0}^{k} P[\tilde{A} \geq k - i] P[\tilde{\Gamma}_n = i] + \sum_{n=1}^{\infty} \sum_{i=0}^{k} P[A \geq k - i] P[\tilde{\Gamma}_n = i] P[\Theta = x]
\] (6.15)
While $A$ and $B$ are useful in determining the overall interference solution, the interference solution in this system relies mostly on a related random variable, $\tilde{A}$ seen in Equation 3.37, reproduced as equation 6.15 for convenience. In Equation 6.15, the random variable $\tilde{A}$ describes the on-burst with which the collision occurred. Note that when IEEE 802.11b receives a frame which is incorrect, it will go through a recovery process. The key to the interference mitigation algorithm presented in this dissertation is predicting when this recovery process will end and thus Bluetooth can safely transmit again without damaging the MSDU on which a frame has been corrupted. The recovery process requires a $DIFS$ delay followed by a backoff followed by the exchange RTS/CTS/data/ACK, where the data transmitted is the frame that was previously lost. Any transmissions will be forced to go through a contention window before it is ready to transmit, since it is reasonable to assume that the last transmission required a retransmission, making the inter-burst space very small. Furthermore, it is reasonable to assume that the RTS/CTS exchange will fail at least once, since the exchange could have been corrupted by a Bluetooth transmission, therefore, it is generally reasonable to assume that a collision with a data frame will result in the implementation of the third backoff window size, or approximately $2.4\text{ms}$, then the longest possible recovery period is

$$rec \leq DIFS + CW + A_{RTS/CTS} + A_{main}$$

(6.16)

The integration of this modified $\tilde{A}$ value into Equation 6.15 would require a solution that employs a queuing model for this bursty traffic. Such a model is very difficult to describe, and has not been successfully accomplished since the creation of the self-similar traffic model in 1994. Therefore, it was deemed beyond the scope of this dissertation. Instead, an approximation was developed as a guide for the likely point of low-probability of collision retransmission. Equation 6.15 can be biased by Equation 6.16, leading to an upper bound on the collision probability.
6.3.2 Interference Solution

The goal of the algorithm is to create a seamless integration between Bluetooth and IEEE 802.11b. Thus, it should be possible for the Bluetooth system to be calibrated such that no significant impact is visible on the IEEE 802.11b system from the Bluetooth traffic.

Given the description in the previous section of random variables $A$ and $B$, and given the assumption that the CW is set to CWMin after a collision, given the traffic source described in Section 6.3, the probability that the channel is occupied at any time $t$ after a collision was calculated using Equation 3.39 in the Chapter 3 with the bias described in the previous section. The estimate of the probability that the channel is occupied at any time $t$ after a collision is seen in Figure 6.9.

As seen in Figure 6.9, the probability of collision drops quickly until it begins to settle between 8ms and 10ms. Therefore, a backoff of 7 Bluetooth slot pairs, or 8.75ms was selected for this case study. Given that the on periods for the source are heavy-tailed, a large amount of
Table 6.1: CBMA Implementation

1. If queue delay = 0 and packet ready for transmission, transmit packet
2. If queue delay ≠ 0, reduce delay by one and go to the beginning of the list
3. If packet incorrect (collision detected), then set delay to \( \tau \) and place packet at the top of the queue for retransmission
4. If packet correct, then reset queue delay to zero
5. Go to Step 1

uncertainty is injected into the visible channel pattern after the first packet is sent through the channel. As the packet being transmitted ends, it is no longer possible to predict when the next packet should arrive. This uncertainty is manifested by the fact that the probability that the channel is occupied drops significantly after the packet ends, yet quickly stabilizes to a fixed probability for all time after that. Note that the approximation seen in Figure 6.9 is based on the assumption that all packets are sent in overlapping channels. In reality, the probability of any one Bluetooth packet overlapping an IEEE 802.11b frame should be weighed by \( \frac{22}{79} \), the percentage of the Bluetooth operating band that the IEEE 802.11b device occupies.

The quickly converging probability of channel occupied leads to a modified implementation of the collision avoidance algorithm. Since the oscillations in the probability that the channel is occupied seen in the case where exponential arrival statistics do not appear in the case with heavy-tailed packet lengths and exponential “off” periods, the algorithm needs to be modified to account for this quick convergence. The modification required is, instead of reducing delay by one after every transmission, the delay is reset to zero after the first correct transmission of a packet. With this modification, the algorithm is modified to the set of steps shown in Table 6.1.

Fundamentally, the only change in the algorithm is that a successful transmission resets
the delay counter. It should be noted that the assumption that underlies this case study is that a system’s MAC has some structure which is inherent to the standard’s structure. In this case, the structure that is leveraged is IEEE 802.11b’s structure. It has been shown by Tickoo et. al [64] that the aggregated traffic characteristics of IEEE 802.11 start to show some definite structure instead of the expected self-similar structure. In this dissertation, the structure inherent to a single source is leveraged to achieve coexistence. It should be noted that while the source for the IEEE 802.11b link in this case is based on the building blocks for self-similar traffic, it is the shaping performed by the MAC on that one source that is leveraged to achieve coexistence.

6.4 Simulation

The performance of WLAN is generally very difficult to characterize analytically. The difficulties include issues such as arriving at an accurate description of the state of the queue, which becomes difficult or impossible for some of the recent traffic models. Added to this problem is the issue of several adaptive aspects of some protocols, such as the contention window of IEEE 802.11 devices. Given these difficulties, the study of these types of systems is generally done using simulation. In this case study, given the complexity of the systems under study, the method of choice is simulation.

6.4.1 Device Models

The device model that is used is based on the IEEE 802.15 Part 2 standard for coexistence. This standard provides the path loss, receiver behavior, interference rejection, and receiver statistical behavior for both IEEE 802.11b and Bluetooth devices. The details of the IEEE 802.15 Part 2 specifications are provided in Section 2.4. It should be noted that a simple channel model was selected for these devices. While another channel might be a more accurate description of the radio characteristics of the system, not only is this approximation
Figure 6.10: Scenario Layout

widely used, but it was also deemed by the IEEE as a sufficiently accurate method to determine the likely behavior of the system.

The basic configuration used for the environment follows the configuration seen in Figure 6.10. The IEEE 802.11b devices are set some distance \( d \) apart, while the Bluetooth devices are set some distance \( d_2 \) apart. As seen in Figure 10, the configuration is co-linear. Given that the channel assumption is that no fading exists, the channel is flat across all frequencies, and the path loss follows a smooth decay following two basic path loss exponents, it follows that the environment contains no obstructions and no reflectors. Therefore, a line-of-sight path exists between all devices in the local environment. While the distance between devices is scenario-specific, the device configuration described above is the one followed for all the simulations in this study.

**Noise Figure**

The receiver noise figure is required in each of these systems, providing a signal noise floor and hence a signal-to-noise ratio necessary to generate the error pattern that matches the received signal strength.

Neither the Bluetooth nor the IEEE 802.11b specifications provide a specific noise figure. However, the noise figure can be derived from the given data.

*Bluetooth Noise Figure*

The Bluetooth specifications say that that maximum acceptable FER that a Bluetooth link
may have is $10^{-3}$ when the signal is at the edge of the detection threshold of $-70\,dBm$.

A frame error occurs in Bluetooth when the synchronization word, the header, or the payload of the Bluetooth packet fails the detection process. This process can be described as

$$FER = 1 - (P[S] P[H] P[Y])$$

where $S$ is the synchronization event, $H$ is the header event, and $Y$ is the payload event. Given the event probabilities described in Equations 6.1 through 6.4, the frame error rate as a function of the probability of bit error can be described as

$$FER(p) = 1 - \left( \sum_{i=0}^{72-T} \left( \frac{72}{i} \right) p^i (1-p)^{72-i} \right) \left( 1 - (3p(1-p)^2 + (1-p)^3)^{18} \right) (1-p)^{240}$$

Given Equation 6.18, the probability of bit error can be found such that the FER is $10^{-3}$. It is interesting to note that the dominant vulnerability of the packet is the payload. Once the appropriate BER for the resulting FER is determined, the SNR can be found. This SNR was found to be $13.8\,dB$. Given an SNR of $13.8\,dB$, the noise floor can be such that when the received signal strength is $-70\,dBm$. Following this procedure, the noise power spectral density was calculated to be $-83.80\,dBm$.

**IEEE 802.11b Noise Figure**

The noise figure for IEEE 802.11b is calculated in a similar fashion as for the Bluetooth system. The specifications given in the IEEE 802.11b standard specify that when the received signal strength is $-76\,dBm$, the maximum allowable FER is $8 \cdot 10^{-2}$ given a PPDU length of 1024 bytes. The longest IEEE 802.11b packet, and hence the most vulnerable one, is the packet carrying the maximum data payload. Since the IEEE 802.11b packet is split into two parts, the PLCP and the MPDU, the equation describing the probability that the packet is corrupted can be split into two parts
\[ FER = 1 - \left( (1 - p_1)^{192} (1 - p_2)^{8192} \right) \]  

(6.19)

Using the IEEE 802.11b receiver characteristics described in Section 2.4.2, the SNR necessary to achieve a FER of \(8 \cdot 10^{-2}\) at the received signal of \(-76dBm\) is calculated to be \(-83.582dBm\).

It should be noted that the IEEE 802.11b and Bluetooth noise figures were included in this set of simulations to account for the viable range of the devices in this system. These noise figures are not part of the IEEE 802.15 Part 2 description. The main effect of this added noise is that the probability of frame error increases with a collision. In the case of the IEEE 802.11b device, this means that the presented throughput figures are lower than those that would otherwise result - therefore the results presented in this dissertation are a worst-case scenario. In the case of Bluetooth, the effect is that the likelihood that the Bluetooth device detects a collision increases. However, given the simulation’s maximum IEEE 802.11b device range of 33m, the minimum power received by a Bluetooth device from an IEEE 802.11b device with which a collision has occurred is \(-72.2dBm\). The additional noise from the Bluetooth receiver raises that received signal strength to \(-71.94dBm\), resulting in a difference of just 0.3dBm. It was judged that this difference in received signal strength is insignificant and should have no visible effect on the overall link results.

### 6.4.2 Metrics

System metrics are a sensitive issue, since they provide the groundwork necessary to evaluate the performance of a system. In WLAN, there are fundamentally three metrics that can be used to evaluate the raw performance of the network: delay, and jitter (delay variation), and throughput. Other metrics exist such as the amount of time needed to break the encryption, and link setup and termination time. However, these metrics are not particularly relevant to what in a classical communications framework is considered to be performance.

Delay and jitter are closely related, so they will be discussed as a single set of metrics. Delay
implies that there is a queue that holds packets until the underlying link can service them. In the case of the IEEE 802.11b link there are two basic conditions seen in the system: either the link can service the packets at a sufficiently high rate, in which case the queue will be empty long enough for the Bluetooth system to be able to use some of the non-transmission times, or the link will not be able to support the traffic source, in which case the link data rate will be lower than the traffic generation rate, and the queuing delay will consistently increase. Thus, for the case of IEEE 802.11b, queuing delay is not as appropriate a metric as other metrics would be. In the case of the Bluetooth link, the source is an infinite source, which means that a packet is always ready for transmission. If the packet was always ready for transmission, the time that the packet spent on the queue is meaningless. However, a related delay metric that is relevant is the time it takes a packet to be successfully delivered once its transmission has begun. Given that an artificial delay is injected in this algorithm every time that a collision is detected, the amount of latency that added delay injects on the link is important. The graphs that are presented in this dissertation describing delay show this mean delay value.

It should be noted that there is a minimum Bluetooth link delay. For a packet to be correctly delivered, it needs to be delivered, which takes 625µs and it needs to be acknowledged, which also takes 625µs, making the minimum delay for the link 1.25ms. The graphs representing Bluetooth delay will not shown any delay that is less than 1.25ms.

Throughput is another metric that provides the insight needed to determine the performance level of all the networks. Throughput is defined as the number of correct bits delivered per unit time. Given simulation time τ seconds, and total number of correct bits delivered n, throughput θ is described as

$$\theta = \frac{n}{\tau}$$  \hspace{1cm} (6.20)
6.4.3 Convergence

The simulations in this dissertation are based on a system clock rather than an event count. Therefore, to estimate the convergence time, the relative convergence of the traffic sources was used as a measure of the convergence time. Given that Bluetooth is modeled with an infinite traffic source, then the only traffic source whose convergence time is in question is the IEEE 802.11b’s heavy-tailed source. Given this source, experiments showed that approximately 400 trials yielded a mean value near the expected value of the random variable. It was found that approximately 400 events yielded relatively stable results. Therefore, the convergence period of the simulation was set to 5s. Since the 5s simulation time yielded a relatively stable result, each simulation point in this case study is the mean result of five trials lasting 5s each.

6.4.4 Validation

Before the simulation results can be presented, the simulation structure needs to be validated first. Since all the aspects of the systems presented in this study cannot be determined in a closed-form solution, not all the results presented in this study can be validated through a complete analytical description of every single result. However, some of the simulation results can be derived analytically, where an agreement between the analytical and simulation results can be used to guarantee that the simulation is working properly.

Bluetooth Validation

The Bluetooth simulation was validated in the previous chapter, where Bluetooth devices were used for the Hybrid ARQ and increased-strength FEC simulations.
IEEE 802.11b Validation

To validate the IEEE 802.11b devices, it is important to first determine the expected throughput of an IEEE 802.11b link. In the scenarios chosen in this simulation, there is a single IEEE 802.11b pair, thus a single link of IEEE 802.11b devices is validated in this section. This validation assumes that the packet is transmitted by the IEEE 802.11b device when the channel is not contested.

An IEEE 802.11b link is composed of several packets and several delays. The link is composed of first an RTS/CTS pair, therefore for each packet that needs to be transmitted, a single RTS/CTS exchange is necessary, where a SIFS period separates the RTS/CTS pair. Note that the packet to be transmitted may occupy more than the payload of a single packet, so it may need to be fragmented into several pieces for transmission. However, a single RTS/CTS exchange is required for all the fragments composing a single data packet. Before the RTS/CTS exchange occurs, the system observes the state of the channel for a period of time DIFS. Thus, before data can be transmitted, the device occupies

\[ T_{RTS/CTS} = DIFS + RTS_t + SIFS + CTS_t \]  

(6.21)

where the suffix \( t \) denotes the duration of a frame. After a successful RTS/CTS exchange, the device will wait SIFS time, followed by a number of packets (PPDU) and an equal number of acknowledgement packets, bringing the total time occupied by a single packet exchange to

\[ T = DIFS + RTS_t + SIFS + CTS_t + (SIFS + PPDU_t + SIFS + ACK_t) \cdot N \]  

(6.22)

where \( N \) is the number of packets occupied. Note that \( N \) need not be an integer. The time per packet needs to be modified to account for the inter-packet spacing resulting from the on/off source’s off time. Since the link has some added overhead due to hand-shaking
procedure this overhead needs to be discounted from the traffic source’s off time. The link off time can be described as

\[ t_{\text{off}} = \tau_{\text{off}} + \tau_{\text{overhead}} \]  \hspace{1cm} (6.23)

The total packet overhead can be defined as

\[ h_t = PPDU_{t,\text{overhead}} + MPDU_{t,\text{overhead}} \]  \hspace{1cm} (6.24)

\( \tau_{\text{overhead}} \) is then

\[ \tau_{\text{overhead}} = DIFS + RTS_t + SIFS + CTS_t + (SIFS + h_t + SIFS + ACK_t) \cdot N \]  \hspace{1cm} (6.25)

Using Equations 6.21 through 6.25, the data rate for the system can be calculated as

\[ \text{rate} = \frac{N \cdot b}{T + (\tau_{\text{off}} - \tau_{\text{Overhead}})} \]  \hspace{1cm} (6.26)

This equation is the simplified to

\[ \text{rate} = \frac{N \cdot b}{\frac{bN}{T} + \tau_{\text{off}}} \]  \hspace{1cm} (6.27)

where \( b \) is the number of data bits per frame. Note that if a new burst of data is ready for transmission as soon as the current burst transmission has been completed, the MAC is supposed to enter the contention process, making sure that the current device does not monopolize the use of the channel. In such a case, the above derivation needs to be modified to account for this extra delay. In the above derivation it was assumed that, on average, the amount of idle time between “on” bursts is sufficiently long to account for this contention process. Assuming a traffic pattern using a Pareto distribution for the packet length:
where \( \alpha = 1.7 \) and \( \beta = 1 \). The exponential distribution for the inter-packet spacing is:

\[
f_{off}(t) = \frac{1}{\lambda} e^{-\frac{t}{\lambda}}, t \geq 0
\]

where the mean off period is 7.5\( ms \).

Given that the system under study is IEEE 802.11b, \( b = 18496 \), the maximum length of the frame body of the MPDU. Since a Pareto distribution is used for the burst lengths, the mean burst length is calculated as

\[
N = \frac{\alpha}{1 - \alpha \beta}
\]

which, when \( \alpha = 1.7 \) and \( \beta = 1 \), then \( N = 2.428 \). Placing the resulting values into Equation 6.27, the system data rate can be calculated to be

\[
rate = 3.87Mbps
\]

Using these same values for the traffic model, the simulation yielded

\[
rate = 3.85Mbps
\]

Note that the simulation yielded a data rate that is similar to that derived analytically. This shows that the link manages to absorb the overhead incurred by both the MPDU header and trailer and the PLDU header. If a new MPDU is received by the radio interface while it is transmitting a packet, the incoming MPDU is queued and handled accordingly when it reaches the top of the queue. Since the time between bursts is, on average, longer than the amount of time incurred by the average overhead, on average the queue will remain empty,
and the data can, over the total length of the simulation, be serviced at the rate that it is received.

### 6.5 Algorithm Performance on IEEE 802.11b

The simulation was used to test the performance of the algorithm. Figure 6.10 shows the setup used for the simulation, which is reproduced here for ease of reading.

As seen in Figure 6.11, two IEEE 802.11b devices were set $d$ meters apart, where $d = 33$. A pair of Bluetooth devices, set $d_2 = 2$ meters, apart were moved between the next to the IEEE 802.11b source device to next to the IEEE 802.11b destination device. The decision to space the IEEE 802.11b devices by $33m$ is based on the propagation model used. A range was desired where the devices were sufficiently close that the likelihood of bit error is very low yet at the same time seemed like a reasonable range, so $33m$, yielding $SNR = 24.8dB$, was chosen. With such high SNR values, the principal source of errors will be interference from the Bluetooth devices. The reason for choosing a $2m$ separation between the Bluetooth devices was arbitrary, since bit errors are not used to detect collisions in the baseline scenario. A scenario is later presented where the distance between the Bluetooth devices is changed to show that the likelihood of detection. Two basic configurations were used for the Bluetooth devices. In one configuration, the Bluetooth devices are using a DH1 configuration, where packets are transmitted from one device to the next using no FEC and a 16-bit CRC code for error detection. In the other configuration, the devices use the algorithm presented in
this paper, which for the purpose of identification is called CBMA, collision-based multiple access.

In the previous section, the ideal waiting period of 8.75\,ms was determined for the Bluetooth device, and this time is used as the backoff value for the algorithm. Note that the IEEE 802.11b attributes \texttt{dot11ShortRetryLimit} and the \texttt{dot11LongRetryLimit}, which limit the number of times that a transmission attempt of a packet can be performed before the packet is discarded, were set to 7 and 4, respectively; these are the default values that are suggested by the IEEE 802.11 standard. Also, the threshold that was set for the physical carrier sense mechanism in the IEEE 802.11b device is an aggregate signal strength of $-76\,dBm$. This threshold was set because the device sensitivity is rated in context of a received signal of $-76\,dBm$, and the standard provides no other information concerning signal detection. The lifetime of the MSDU, or the maximum length that a burst can take, was set to infinity. This is a value that can be set by the user, where the default value is approximately 0.5s. Given that the default value is so long, and given that the a limit has the potential of setting a ceiling on the bursty source, it was deemed that this value should be set to infinity. It should be noted that this limit is set to reduce the likelihood that a single link monopolizes the channel. Given that a single IEEE 802.11b device link was implemented in this simulation, channel trapping behavior is not a concern. Furthermore, the BSS beacon, which is transmitted at regular intervals, was not implemented in this simulation. The beacon is a short burst that contains timing information for the devices in a BSS. This beacon is not acknowledged, is short, and occurs only rarely; the default setting for the beacon period is 100\,ms. Given these aspects of the beacon, its inclusion in the simulation of a link should have no appreciable effect on the results, so it was not implemented.

Note that in all simulations, IEEE 802.11b Channel 7, centered at 2.442\,MHz, was selected for the IEEE 802.11b link. This channel was selected to allow for adjacent channel interference in both the upper and lower parts of the occupied channel. Furthermore, it was assumed that the transfer of information in the Bluetooth devices is in only one direction; i.e. the payload section of the Bluetooth packet on the reverse channel is ignored and the only pertinent parts
of the reverse channel packet are the header and preamble sections. This is analogous to the way in which the IEEE 802.11b link is simulated, where data flows in only one direction. Since AUX1 packets are used on the link, the ARQ mechanism that is employed is driven by the application, and it is possible to ignore specific parts of packets. However, since the Bluetooth packet is fixed-length, a payload is still transported on the reverse channel. Errors in this payload, along with errors in any other part of the forward- and reverse-link packets, were used as a mechanism to detect collisions with an IEEE 802.11b (when perfect detection was not used).

Figure 6.12 shows the throughput of the IEEE 802.11b device as a function of distance between the destination IEEE 802.11b device and the nearest Bluetooth device. For this scenario, perfect detection of an IEEE 802.11b packet is assumed for the Bluetooth packet, regardless of whether or not the Bluetooth device is capable of detection the IEEE 802.11b packet. The scenario where the detection of a collision is taken into account is discussed in a later section.
As seen in Figure 6.12, in the unaltered version of IEEE 802.11b/Bluetooth coexistence drops the effective data rate of IEEE 802.11b by over 80%. As can also be seen in Figure 6.12, given a separation of 33m between the IEEE 802.11b devices, the range over which the Bluetooth device has the most impact on the IEEE 802.11b link is under 8m, or 26ft, and is quite dramatic at 5m, or 16.5ft. This range is around the same size as a typical office; this means that a Bluetooth device operating within an office space at the same time as an IEEE 802.11b device should have a significant impact on an IEEE 802.11b link. Note that the impact is visible in only one end of the graph, the area where the IEEE 802.11b destination device is. At the higher distance, the Bluetooth piconet is approaching the IEEE 802.11b data source. This effect is probably to be expected, since near the source device, the Bluetooth traffic can affect the IEEE 802.11b link only by triggering the physical carrier sense mechanism during either the DIFS or contention backoff (when under effect), or either of the IEEE 802.11b's transmitted packets, the CTS packet or the ACK packet, each of which is shorter than a single Bluetooth packet.

It should also be noted that the in [65], a maximum IEEE 802.11b coexistence throughput of 3Mbps is possible when the mean frame payload size is 750 bytes. In that study, the maximum frame size used is 1500 bytes, and it was shown that the expected throughput of the IEEE 802.11b link begins to drop from the maximum of 3Mbps when the frame size is 750 bytes. In that scenario, adjacent channel interference was not taken into consideration, and the frame size was below the RTSThreshold, so no RTS/CTS exchange occurred before the transmission of a frame. In this simulation, the average MSDU used is over 5600 bytes and the RTSThreshold is set to zero, so all frames are required to perform an RTS/CTS exchange before the transmission of any burst of information. Furthermore, in this case study adjacent channel interference is taken into account, further degrading the performance of the link. Given that this case study uses far larger data bursts combined with a channel model that accounts for adjacent channel interference, it is reasonable to expect that the maximum throughput of the IEEE 802.11b link should be significantly lower than the maximum experienced when frames are limited to 750 bytes.
As seen in Figure 6.12, the CBMA algorithm can completely eliminate the effect of the interference. Throughput as a function of distance is flat, where the impact of the Bluetooth device traffic on the IEEE 802.11b device is not visible, regardless of the distance between the Bluetooth device and the IEEE 802.11b device.

Figure 6.13 presents the simulated Bluetooth throughput. Note that a CBMA delay of 8.75ms was selected for the figure because 8.75ms gives the best results for the given traffic model. This concept is described in detail when the sensitivity of the algorithm to delay errors is shown in a later section. The non-CBMA implementation suffers a drop of approximately 50kbps, or 30% throughput, near the IEEE 802.11b source device. Near this device, the Bluetooth devices are likely to receive interference from the IEEE 802.11b device’s transmitted data frames, which can be fairly long and are transmitted at 100mW (compared to Bluetooth’s 1mW). The data frames force the Bluetooth link to re-transmit its data to recover from the error, resulting in a reduced throughput. Overall, a noticeable impact is apparent on the Bluetooth link when implementing the CBMA algorithm. The drop in throughput is between 50% and 60%, ranging between 60 and 80kbps. This penalty translates in a drop in throughput of between 60 and 110kbps, depending on the interference conditions. Note that a reduction in 60 to 110kbps in the Bluetooth link throughput translates to a gain that can be as high as 3Mbps on the IEEE 802.11b link throughput.

Figure 6.14 shows the delay of the Bluetooth link as a function of the distance between the IEEE 802.11b destination device and the nearest Bluetooth device in the piconet. As seen in Figure 6.14, the delay is below 3ms, which is roughly 2ms more than the minimum delay of 1.25ms. There is an increased delay near the IEEE 802.11b source device. This increased delay is due to error recovery from interference from the IEEE 802.11b device, seen also as a reduction in throughput in Figure 6.13.

One interesting behavior shown in Figures 6.12, 6.13, and 6.14 is that while the Bluetooth link suffers a considerable drop in throughput near the IEEE 802.11b data source, the drop in throughput of the IEEE 802.11b link remains relatively low. The drop in Bluetooth
Figure 6.13: Baseline Impact of Coexistence Algorithm on Bluetooth Throughput [bps]

Figure 6.14: Baseline Impact of Coexistence Algorithm on Bluetooth Delay [ms]
throughput is explained by the fact that long data packets are transmitted by the IEEE 802.11b source, and their transmission cannot be interrupted, it can only be delayed if the physical carrier sense mechanism is triggered by a Bluetooth packet. When near the data source, the only vulnerable aspects of the IEEE 802.11b link are the CTS and ACK packets returned by the IEEE 802.11b destination device. Given that Bluetooth has an infinite source, it is always transmitting. Therefore, a spectral overlap between the Bluetooth selected channel and the CTS or ACK packet is

\[
Overlap = \frac{22}{79} \quad (6.33)
\]

Both the CTS and ACK packets are 202 µs long. The Bluetooth packet used in this simulation is the AUX1 packet, which is 366 µs long; this packet is transmitted over a 625 µs slot. A collision between the CTS or ACK packet and a Bluetooth packet can then occur under two possible conditions. In the first condition, a Bluetooth packet is transmitted over a channel overlapping the IEEE 802.11b channel, while the subsequent packet is not. In this case, the CTS or ACK packet that begins anytime after the 366 µs transmission is completed will not result in a collision. In the second condition, a Bluetooth packet and the subsequent Bluetooth packet occupy channels that overlap the IEEE 802.11b channel. In this case, the 202 µs CTS or ACK has to begin after the 366 µs mark but before the next packet starts to avoid a collision. Each of these conditions is weighed by the relative likelihood of the condition occurring, which is a question of whether or not the second Bluetooth packet occupies an overlapping slot. Therefore, the probability of collision of a CTS or ACK packet is

\[
p_c = \frac{22}{79} \left( \frac{57}{79} \cdot \frac{366}{625} + \frac{22}{79} \cdot \frac{625 - (625 - 366 - 202)}{625} \right) = 0.19 \quad (6.34)
\]

This result was validated with simulation. A failed CTS packet results in a contention delay and subsequent RTS/CTS exchange, and a failed ACK packet results in a timeout, contention delay, and a subsequent retransmission. The simulation results can be explained
by the fact that, given the traffic source’s “off” period of 7.5ms, a mean burst length of 2.43 packets, resulting in 3 ACKs per “on” burst, and the low probability of collision with an ACK or CTS, a retransmission resulting from a collision with a CTS or ACK packet will most likely be slightly longer than the source’s off period.

One of the key aspects of the simulation shown in Figures 6.12 and 6.13 is that there is one set of variables in the testing, the traffic and the associated backoff period. In later sections, more complexities will be added to the simulation structure, adding new dimensions. However, for the sake of simplicity, the baseline for the simulations presented in this dissertation are the simulation results seen in Figures 6.12, 6.13, and 6.14, with the system set to perfect detection.

6.5.1 Parameter Sensitivity

As shown in the previous section, the algorithm is effective in making Bluetooth transparent when operating in an IEEE 802.11b environment. However, the example shown is based on an assumption of perfect knowledge of the traffic in the IEEE 802.11b system. This section includes a study of the sensitivity of the algorithm to errors in the traffic estimation of the IEEE 802.11b traffic.

Delay Sensitivity

In Section 6.3.2, the optimal delay for the system was calculated. However, not always is it possible to have complete information concerning a traffic source, and it becomes important to determine how sensitive the algorithm is to errors in the selection of a delay. In this section, simulation results are presented that cover a range of delay values.

Figure 6.15 shows the throughput of an IEEE 802.11b link when in the presence of a Bluetooth piconet as a function of the distance between the IEEE 802.11b data destination node and the closest Bluetooth device. The graph shows a variety of CBMA delays,
Figure 6.15: Sensitivity of IEEE 802.11b throughput to different delay settings in CBMA algorithm
As seen in Figure 6.15, the delay injected into the Bluetooth queue has a strong impact on the performance of the algorithm. One point to note is that when delays that are less than 8.75\(\text{ms}\) generally yield a throughput in the IEEE 802.11b link that is noticeable. However, configurations with the delay of 8.75\(\text{ms}\) or more yield a relatively flat throughput curve for the IEEE 802.11b system. To explain why this effect occurs, it is important to look at the visible traffic result shown in Figure 6.9. To improve the readability of this document, Figure 6.9 was reproduced in Figure 6.16.

As seen in Figure 6.16, the probability that the channel is occupied as a function of time quickly convergences to a single probability value. This means that, after a certain point, the level of uncertainty is such that it is no longer possible to predict the state of the channel, and no added gain on the IEEE 802.11 throughput can be achieved through extra delays. Note that Figure 6.16 was used to approximate the state of the channel, leading to a delay time of 8.75\(\text{ms}\), and that any delays less than 8.75\(\text{ms}\) yield a decrease in performance, while delays greater than 8.75\(\text{ms}\) yield no improvements in IEEE 802.11b performance. This means that
Figure 6.17: Sensitivity of Bluetooth Throughput to different delay settings in CBMA algorithm

the approximation seen in Figure 6.16 is accurate. The only effect that any delay beyond the optimal will incur is an extra reduction in the throughput of the Bluetooth system. This is evident from Figure 6.17, where the Bluetooth throughput is shown for the different configurations. As seen in Figure 6.17, the impact on the Bluetooth throughput can be significant.

Figure 6.18 shows the delay of the Bluetooth link as a function of the different wait settings. As seen in Figure 6.18, the delay of the Bluetooth device varies between 2ms and 3ms, depending on the wait parameter that is selected.

On Sensitivity

Traffic can change as a function of time. These changes can be due to the change in the service that is supported by the system. These changes can also be due to changes in
Figure 6.18: Sensitivity of Bluetooth Delay to different delay settings in CBMA algorithm

the characteristics of the application. Changes in service include supporting web services versus supporting email services. Application characteristic changes include aspects such as a change in the length of the files or messages passed between two devices, where these changes are the result of switching from a web site high in text content to one supporting streaming video. In some cases, the length of the data bursts will change, since an image can be considerably longer than a text stream.

Given that the expected packet length supported by the system can change as a function of time, it is important to evaluate the impact of changes in the on period of a traffic source on the CBMA algorithm.

Figure 6.19 shows the IEEE 802.11b traffic for a series of configurations where the traffic model’s on period has an average of 4.4kB, 5.5kB, 11kB, 16.5kB, respectively. As the data rate increases, a floor appears on the throughput curves at approximately 4Mbps. This is reasonable, since the Bluetooth backoff period was optimized for an IEEE 802.11b
throughput of approximately 4Mbps with this specific type of on/off source. When the traffic load is higher than that supported by the system parameters, the system is no longer capable of quickly servicing the arriving data packets. These mechanics are described visually in Figure 6.20.

As seen in Figure 6.20, the basic problem is that, when the Bluetooth backoff is insufficient, the packet has a difficult time being fully serviced. When a collision occurs, the packet will need to be re-transmitted. The Bluetooth delay is based on the precept that the IEEE 802.11b packet with which a collision occurred will end, the contention period will expire, and the IEEE 802.11b packet will be re-transmitted before the Bluetooth backoff delay expires. However, if the delay is shorter than the expected IEEE 802.11b packet length, then Bluetooth will transmit a packet during the IEEE 802.11b transmission, possibly causing another collision. This behavior is nearly circular, and may result in a series of collisions to occur with a single frame as it is constantly re-transmitted. It may occur that the MSDU is serviced at reduced pace, where most frames in it have to be re-transmitted at once or more.
Figure 6.20: Increased Queuing Delay Due to Extended On Period
Figure 6.21: Impact of Different On Periods in IEEE 802.11b Source on Bluetooth Link

Given that this behavior is not deterministic, but a random process instead, eventually a circular pattern will eventually be broken when Bluetooth selects a non-overlapping set of channels, or some other random event, like IEEE 802.11b not detecting a collision, which will break the cycle. More importantly, the transmissions will significantly reduce the IEEE 802.11b device’s service rate, reducing the overall throughput of the link. This inability to keep up with the incoming traffic is manifested as the non-flat throughput curve in the IEEE 802.11b seen in Figure 6.19 for the longer on-period traffic sources.

Figure 6.21 shows the Bluetooth throughput for each of the different IEEE 802.11b traffic settings.

As seen in Figure 6.21, the throughput of the Bluetooth link can be affected by collisions with IEEE 802.11b, to different degrees, depending on the traffic transported over the IEEE 802.11b link and depending on where the Bluetooth piconet is located. When the Bluetooth piconet is near the IEEE 802.11b source device (high distance on the graph), the Bluetooth
throughput can decreases as a function of the length of the source’s “on” burst length. This is because the longer “on” burst has the potential to cause more errors on the Bluetooth link, while the Bluetooth link has a limited potential in causing harm to the IEEE 802.11b link. However, when the Bluetooth piconet is near the IEEE 802.11b destination device, the throughput stabilizes at around 50kbps. This floor is due to the fact that the only significant cause for errors in the Bluetooth link, the data frames, cannot cause errors on the link because the signal is too weak, and the relative likelihood of a collision with an IEEE 802.11b frame has stabilized to a fixed value, as evidenced by the ceiling in IEEE 802.11b throughput at this distance.

Figure 6.22 shows the delay of the Bluetooth link as a function of the “on” period of the IEEE 802.11b traffic source. As the number of collisions increase due to a mismatch in the traffic pattern, the delay of the Bluetooth link can increase to a maximum of approximately 3.75ms. This delay is triple the minimum Bluetooth link delay of 1.25ms, but can still be
considered fairly low for most applications.

**Off Sensitivity**

Not only can the “on” period of a traffic source change, but so can the “off” period. The off period of a traffic source can be affected by a variety of factors, such as the overall network load. Even though the simulation set presented in this dissertation covers only one specific link of a system, some of the links are sometimes part of larger networks. The IEEE 802.11b link is assumed to be supporting a service like web browsing. This type of service is generally supported over a multi-hop link between the client device and a server somewhere on the Internet. Given that the overall Internet plays a role on the link traffic, delays beyond those incurred at the last link are possible. These delays will generally be manifested as increased separation between arriving packets.

Furthermore, mechanisms such as TCP are generally used for IP traffic. Added delays on the link may trigger the TCP congestion control algorithm, changing the amount of data transported. This change in data transported is not manifested as changes in the source on time, since no packet fragments are changed. Instead, these changes are manifested on the “off” time, changing either the mean “off” time or the distribution of the “off” times.

In this scenario, the resilience of the algorithm is tested against “off” times that are not ideally matched to the delay implemented by the Bluetooth device. Figure 6.23 shows the throughput supported by the IEEE 802.11b link in the presence of a Bluetooth link with a fixed backoff of 8.75ms when the IEEE 802.11b traffic source off time is changed.

As seen in Figure 6.23, the IEEE 802.11b performance is relatively flat when the “off” time is greater than or equal to 7.5ms. When the mean off time is less than 7.5ms, the traffic supported over the IEEE 802.11b link drops to a ceiling of around 4.5Mbps. The reason for this behavior is seen in Figure 6.24.

As seen in Figure 6.24, the reason for the ceiling in the performance of the IEEE 802.11b
Figure 6.23: IEEE 802.11b Throghput for Traffic Source with Different Off Period

Figure 6.24: Increased Queuing Delay Due to Shorter Off Period
Figure 6.25: Effect of different IEEE 802.11 source Off times on Bluetooth throughput

link when the Bluetooth delay is not properly matched to the IEEE 802.11b traffic source “off” time is that the allotted “off” time is not sufficient to completely service the queue before the Bluetooth packet is transmitted.

To minimize this problem, given that the packet after the collision is ready for transmission very soon after the current packet has gone through retransmission, the Bluetooth delay after a collision should be set to a higher value, unless a ceiling in throughput of approximately 4.5Mbps, is acceptable. In other words, if it is known that the traffic source “off” time is very short, and if it is known that the packet that arrives after the current packet will be queued immediately after the current packet is re-transmitted, then the Bluetooth delay should be set to the point where the current packet is allowed to be re-transmitted and the next packet is allowed to be transmitted. Of course, this configuration will incur added delays on the Bluetooth link. The acceptable points for this tradeoff are issues that need to be resolved by anyone deploying such a set of systems. The tradeoff in the selection of a Bluetooth delay is clearly seen when viewing the Bluetooth throughput, seen in Figure 6.25.
Figure 6.26: Effect of different IEEE 802.11 source Off times on Bluetooth delay

As seen in Figure 6.25, some significant penalties on the Bluetooth link can be incurred by the coexistence between Bluetooth and IEEE 802.11b, especially when the match is not proper. When coexisting with the IEEE 802.11b system, the Bluetooth link can be as slow as 25kbps. Note that when the Bluetooth piconet is near the IEEE 802.11b destination device, the Bluetooth throughput does not fall below 45kbps, or roughly the same minimum data rate under the same conditions in Figure 6.21. Given that bit errors are unlikely to occur at this end of the link, the single source of throughput reduction is the CBMA delays, which happen to be the same in the Off Sensitivity case as in the On Sensitivity case.

Figure 6.26 shows the delay incurred over the Bluetooth link as a function of different IEEE 802.11 source mean “off” time. As seen in Figure 6.26, as the “off” time decreases, the Bluetooth link delay increases. Since the “off” time of the IEEE 802.11 source determines how likely the channel will be open when the current delay expires. As the mismatch increases, the likelihood of collision increases significantly, leading to a considerable increase in expected number of retransmissions (with their associated delays). Figure 6.25 shows that
mean delay can be as high as $6 ms$, which is significantly higher than the minimum delay of $1.25 ms$.

However, when considering the IEEE 802.11b low-traffic case, Figures 6.25 and 6.23 show that the Bluetooth link supports a rate that is inversely proportional to the IEEE 802.11b rate. As the IEEE 802.11b rate decreases, the Bluetooth data rate increases. This means that a mis-estimation of the IEEE 802.11b source “off” time has little effect on the Bluetooth throughput, since, as expected, the data rate of the Bluetooth link will follow a generally good sharing format. The biggest throughput price paid when a mismatch is performed is that the IEEE 802.11b link data rate may pay a significant penalty in throughput, as seen in Figure 6.23. However, there is an additional penalty that the Bluetooth device may pay, but instead of throughput, this price is manifested in overall link latency. The additional link latency, seen in Figure 6.25, can be as high as $6 ms$, or 5 times the minimum link delay.

### 6.5.2 Multiple Devices

Given that the algorithm presented in this dissertation is based on the concept of implementing a fixed link backoff, it is reasonable to question what happens when two or more links using the same coexistence algorithm are operating in the same area at the same time. Generally, if after a collision two systems are backed off with exactly the same delay, then it is expected that the two systems will constantly collide.

To evaluate the effect of the CBMA algorithm on two Bluetooth links operating at the same time, a simulation was run with two co-sited piconets to test what happens when two Bluetooth devices are placed in close proximity while using the CBMA algorithm. Both of these systems were set such that, when a collision occurred, both piconets detected it and applied an added delay to the system queue. The delay was set to $8.75 ms$, using the delay that was used for the IEEE 802.11b system. Since the delay was set to the same level for both systems, then no optimal avoidance level can be found.
Table 6.2: Throughput Performance with Overlapping Piconets

<table>
<thead>
<tr>
<th>Piconet</th>
<th>One</th>
<th>Two</th>
</tr>
</thead>
<tbody>
<tr>
<td>Throughput (bps)</td>
<td>135 kbps</td>
<td>135 kbps</td>
</tr>
</tbody>
</table>

Since the two systems have exactly the same behavior, and since both systems are set to have a perfect detection configuration, then both systems should have exactly the same performance. Table 6.2 shows the performance of both of these systems at a distance between master and slave in each piconet of 2m, where the distance between the masters of each piconet and the slaves of each system are set to zero.

As seen in Table 6.2, both systems have the same data rate. This is to be expected, since both systems have exactly the same environment, where the SNR is sufficiently high for an error to occur almost never, and where a collision between these systems will always be detected by each of these systems and where the effect of that collision is the same for both systems.

The impact of two Bluetooth piconets on an IEEE 802.11b link yields interesting results. The throughput of the Bluetooth piconets is seen in Figure 6.27.

As seen in Figure 6.27, the throughput for each of the Bluetooth systems is fundamentally the same. Figure 6.28 shows the delay for the Bluetooth systems in the simulation - this delay is also fundamentally the same for both Bluetooth devices. The fact that the performance of the Bluetooth devices is the same is understandable, since both systems are affected in effectively the same manner by the IEEE 802.11b system.

When looking at multiple devices, the most interesting effect is the effect of the dual Bluetooth devices on the IEEE 802.11b link. Figure 6.29 shows the throughput of the IEEE 802.11b link as a function of the distance between the master device of each piconet, which is the same for both, and the destination device of the IEEE 802.11b link.
As seen in Figure 6.29, the throughput of the IEEE 802.11b link is affected by the dual Bluetooth links to the point where transparency is not achieved given the same parameter as that used when a single Bluetooth piconet is used. The dual Bluetooth piconets reduce the overall IEEE 802.11b link to around 2.85 Mbps. The reason why the combined Bluetooth piconets have this effect can best be described in conjunction with a visual representation.

The effect of a collision between the IEEE 802.11b link and one Bluetooth piconet is that a single delay is incurred on the Bluetooth piconet with which the collision occurred and the IEEE 802.11b system proceeds with the retransmission process. Unfortunately, since a collision is required for synchronization, one of the Bluetooth piconets is synchronized with the IEEE 802.11b link, but the other is not. The second, unsynchronized, Bluetooth piconet is then capable of colliding with the IEEE 802.11b packet during its retransmission. This second collision will require the IEEE 802.11b link to go through a second retransmission process, during which it may collide with the Bluetooth piconet with which the original
Figure 6.28: Delay of multiple Bluetooth devices when implementing CBMA Algorithm in the Presence of IEEE 802.11b Device

Figure 6.29: IEEE 802.11b Throughput in the Presence of Multiple Bluetooth Devices
collision occurred, since the first Bluetooth system has assumed that the IEEE 802.11b system has finished its original retransmission. Since the Bluetooth systems are frequency-hopping systems, and since the IEEE 802.11b packet length as well as the location over which the collision occurred are both random variables, then there is some chance that the IEEE 802.11b link will ultimately be able to finish its retransmission. This chance that the IEEE 802.11b link will be able to finish is retransmission is the reason why a data rate of around 2.85 Mbps out of a maximum possible 3.9 Mbps for the given traffic settings is seen.

To provide a frame of reference for the 2.85 Mbps, a simulation was run where the Bluetooth devices do not implement the additional delay from the CBMA algorithm. Figure 6.30 shows the throughput of the IEEE 802.11b system in the simulation.

As seen in Figure 6.30, the throughput of the IEEE 802.11b link when two Bluetooth piconets are deployed nearby can be significantly lower than 2.85 Mbps. The results seen in Figure
Figure 6.31: Bluetooth Throughput for Multiple Bluetooth Devices in the Presence of IEEE 802.11b Link when Bluetooth Devices are Not Implementing CBMA

6.30 show that, even though the algorithm is not necessarily completely effective in making the Bluetooth piconets transparent to the IEEE 802.11b link, a significant improvement in performance is still possible, raising the likely throughput of the IEEE 802.11b link. Figure 6.31 shows the Bluetooth throughput for the simulation.

As seen in Figures 6.31 and 6.32, the Bluetooth impact is minimal for both Bluetooth devices. It should be noted that the Bluetooth throughput changes as a function of distance with the IEEE 802.11b device. However, the Bluetooth throughput is not much different from the throughput expected from a single-Bluetooth environment. The primary reason for this behavior is that Bluetooth devices are designed to co-exist with each other, and the CBMA algorithm should be called into use on the rare occasion that a collision occurs. In other words, the Bluetooth air interface is designed to handle interference from Bluetooth links efficiently, so the addition of the CBMA algorithm over the existing Bluetooth link management protocol incurs no real additional penalty on the Bluetooth link.
Figure 6.32: Bluetooth Delay for Multiple Bluetooth Devices in the Presence of IEEE 802.11b Link when Bluetooth Devices are Not Implementing CBMA
6.6 Realistic Implementation

In a real implementation, it is not possible to have perfect collision detection. The reason why such detection levels cannot be guaranteed is that collision detection is based on the existence of bit errors on the packet. These bit errors are based on the level of interference relative to the received signal strength. Not only is it a factor of the distance between the Bluetooth devices and the IEEE 802.11b devices, but it is also a factor of the distance between the different Bluetooth devices in the piconet. This section presents an analysis of collision detection and a simulation showing the different detection thresholds.

6.6.1 Error Detection

The probability of detection is based on the distance between the IEEE 802.11b devices and the Bluetooth devices and the distance between different Bluetooth devices in the piconet. Using the IEEE 802.15 Part 2 channel and system models, the following analysis is possible. The probability of bit error for the Bluetooth receiver is

\[ p_e \left( \frac{E_b}{N_0} \right) = e^{-\frac{E_b}{2N_0}} \]  \hspace{1cm} (6.35)

The received signal level is a function of the transmitted signal level and the distance between the Bluetooth devices, where the distance between the devices affects the path loss.

The Bluetooth received signal strength can be defined as \( R \). The Bluetooth noise floor was calculated to be \(-83.8dBm\), or \(4.17 \cdot 10^{-9}\) in Section 6.4.1. The interference signal transmitted by the IEEE 802.11b signal when received by the Bluetooth device can be defined as \( I \). This signal is spread over \( 22MHz \), where the Bluetooth receiver bandwidth is described by the window \( W(s) \) but can be approximated to \( 1MHz \). The overall SNR is then
$$SNR = \frac{R}{4.17 \cdot 10^{-9} + W(I)} \approx \frac{R}{4.17 \cdot 10^{-9} + \frac{I}{22}} \quad (6.36)$$

The probability of bit error for the Bluetooth device is then

$$p_e(R, I) = e^{-\frac{R}{2\left(4.17 \cdot 10^{-9} + W(I)\right)}} \approx e^{-\frac{R}{2\left(4.17 \cdot 10^{-9} + \frac{I}{22}\right)}} \quad (6.37)$$

Given its length and lack of FEC (in the case presented in this dissertation), the part of the Bluetooth packet that is most susceptible to bit errors is the payload. Therefore, the probability that the packet contains an error can be approximated by

$$p_B(p_e) \approx 1 - (1 - p_e)^{240} \quad (6.38)$$

When taken into context of the interference from IEEE 802.11b, the probability of packet error is then approximated by

$$p_B(R, I) \approx 1 - \left(1 - e^{-\frac{R}{2\left(4.17 \cdot 10^{-9} + \frac{I}{22}\right)}}\right)^{240} \quad (6.39)$$

As seen in Equation 6.39, the probability that a packet collision is detected is a function of the received signal strength and the received level of interference. As discussed in Section 6.4.1, the impact of the receiver’s AWGN is minimal.

In the case of IEEE 802.11b, given that the PPDU is generally significantly longer than the PLCP overhead, the dominant part of the frame will be the PPDU, so the probability of bit error for the IEEE 802.11b frame can be approximated as

$$p_{eQ}\left(\frac{E_b}{N_0}\right) \approx \frac{128}{255} S\left(\frac{E_b}{N_0}\right) \quad (6.40)$$

The receiver noise floor for the IEEE 802.11b devices was calculated as $\approx -83.582 dBm$, which is $4.38 \cdot 10^{-9}$. The DSSS process used by IEEE 802.11b provides a suppression factor of $8 dB$, ...
which is applied to the received Bluetooth signal $I_B$. When the received signal is $R_I$, the resulting SNR is then

$$SNR = \frac{R_I}{4.38 \cdot 10^{-9} + 0.1585I_B}$$

(6.41)

Given the above SNR, the probability of bit error for the IEEE 802.11b frame is then

$$p_{eQ}(R_I, I_B) = \frac{128}{255} S \left( \frac{R_I}{4.38 \cdot 10^{-9} + 0.1585I_B} \right)$$

(6.42)

Assuming a maximum frame size, the probability that the IEEE 802.11b frame contains an error is

$$p_I(R_I, I_B) = 1 - \left( 1 - \frac{128}{255} S \left( \frac{R_I}{4.38 \cdot 10^{-9} + 0.1585I_B} \right) \right)^{18768}$$

(6.43)

As seen in Equation 6.43, the probability that the IEEE 802.11b packet is affected by a collision with a packet from a Bluetooth system is a function of both the IEEE 802.11b packet signal strength and the relative strength of the interference signal. It should be noted that there is a small probability that a collision between Bluetooth and IEEE 802.11b packets will not result on a packet error in the IEEE 802.11b link.

### 6.6.2 Moving Scenario

Given the different signal strengths that are possible in difference configurations, a set of simulations was performed to test the ability of the Bluetooth device to detect a collision with the IEEE 802.11b network. Figure 6.33 shows the throughput of an IEEE 802.11b link as a function of the distance between the IEEE 802.11b destination device and the nearest Bluetooth device of the Bluetooth piconet for a series of Bluetooth devices separated by 1m, 2m, 5m, or 10m.
Figure 6.33: Throughput of IEEE 802.11b device in the presence of Bluetooth link when comparing different Bluetooth device separations
As seen in Figure 6.33, the Bluetooth devices’ ability to detect a collision with the IEEE 802.11b link is strongly influenced by the separation between the Bluetooth devices. At a distance of around 1m, the Bluetooth devices are limited in their ability to detect a collision near the destination device. Even though the link is limited in its ability of detecting a collision, it still provides a minimum IEEE 802.11b throughput of approximately 1.2Mbps, or 500kbps faster than the no-CBMA implementation. However, as the separation between the devices increases, the likelihood of detection increases accordingly. At a separation of 2m, the likelihood of detection is low near the destination device, but the separation is sufficient to mitigate the effects of interference at approximately 5m, raising the IEEE 802.11b throughput to approximately 2.5Mbps. When the separation between the Bluetooth devices is 5m, the likelihood of detection grows considerably leading to a minimum IEEE 802.11b throughput of 2Mbps. When the separation is 10m, the Bluetooth piconet detects collisions sufficiently often to lead to a flat IEEE 802.11b throughput curve. Note that these ranges are based on a separation of 33m between the IEEE 802.11b devices.

In Figure 6.33 the throughput of the IEEE 802.11b link is seen. The striking difference between in throughput under different Bluetooth device separations is just as evident when the throughput of Bluetooth devices is seen, as shown in Figure 6.34.

As seen in Figure 6.34, when the separation between the Bluetooth devices is 1m, the Bluetooth throughput is generally close to 172.8kbps, except when the Bluetooth piconet approaches either the IEEE 802.11b source or destination device. When the separation between the Bluetooth devices is 2m, an interesting effect occurs. When the Bluetooth piconet is roughly 15m from the IEEE 802.11b source and destination devices, the throughput of the link is its maximum; the link never detects a collision. However, as the Bluetooth piconet approaches either device, it has a fairly strong likelihood of collision detection. Near the IEEE 802.11b destination device, the CTS/ACK traffic is strong enough to affect the Bluetooth probability of detection, but the Bluetooth link interference is not strong enough to affect the IEEE 802.11b traffic, decreasing the Bluetooth throughput until this separation grows beyond the detection range. When the Bluetooth device separation is 5m, there is
Figure 6.34: Throughput of Bluetooth device in the presence of IEEE 802.11b link when comparing different Bluetooth device separations
Figure 6.35: Latency of Bluetooth device in the presence of IEEE 802.11b link when comparing different Bluetooth device separations

A gradual gradient in detection that spans the whole 33m. Note that the 2m case is a mix between the 1m case and the 5m case. Unlike the 5m case, the 2m case has a sharp transition between detection/non-detection that because of the range over which the devices were placed. The 1m case shows a clear transition at either end of the link and the 5m case is too far to show such a transition. This observation is further reinforced by the range over which a Bluetooth piconet impacts an IEEE 802.11b link, as seen in Figure 6.33. When the range between the Bluetooth devices is 10m, the Bluetooth piconet detects a collision sufficiently often to remain flat over the whole range, as seen in Figures 6.34 and 6.33.

Figure 6.35 shows the delay incurred by the algorithm implementation for the case in which the separation between Bluetooth devices is 1m, 2m, 5m, and 10m.

As seen in Figure 6.35, the delay incurred per packet is between 1.5ms and 2ms. The delay curves match the expected detection threshold. The delay for the piconet with a 1m separation begins to increase when the distance to the IEEE 802.11b source or destination device is short, the delay for the piconet with a 2m separation shows a sharp transition near
the center of the simulation span, the delay for the piconet with a 5m separation shows a gradual increase over the whole span, and the delay for the piconet with a 10m separation is roughly flat for the whole span. Note that the delay incurred by all these cases is lower than the delay seen in the perfect detection scenario. This difference is due to the fact that not all collisions are detected in the error-detection scenario, and even though the IEEE 802.11b link is capable of supporting a throughput level that is flat over all distances with imperfect detection, the additional delay incurred by the perfect detection scenario is not manifested as additional throughput in the IEEE 802.11b link.

Detection Penalty

The algorithm presented in this dissertation is triggered when an error occurs in the Bluetooth packet. Unfortunately, in a wireless system an error will occasionally occur that was not triggered by interference. As the signal-to-noise ratio of a link decreases, the likelihood that a packet error occurs increases. In the case in which a bit error occurs in a packet where the main source of noise is not interference is called a false positive. The probability of a false positive can then be described as

\[
p_B (R, I) \approx 1 - \left( 1 - e^{-\frac{R}{2(5.495 \cdot 10^{-9})}} \right)^{240}
\]  

(6.44)

A simulation was run to determine the effect of the CBMA algorithm on the range of a Bluetooth link, where Figure 6.36 shows the results from this simulation. As seen in Figure 6.36, the range of the effective range of the Bluetooth devices is approximately 20m. In the case of the system implementing CBMA, the reduction in effective range of the device is approximately 2m. In effect, the CBMA implementation reduces the device range by approximately 10%.

The reason why the reduction in range is so short is that the drop in throughput is very sharp. This means that the Bluetooth link transitions very quickly from essentially error-
Figure 6.36: Comparison of range of Bluetooth link when CBMA algorithm is implemented with when it is not implemented

free to a very high error rate, where the throughput is essentially zero. Since the transition in throughput is so quick, then the distance range over which the additional delay may be triggered by an even other than increased interference floor is short. Therefore, the reduction in the effective range due to the implementation of the CBMA algorithm is limited.

6.7 Algorithm Enhancements

As seen in the previous chapter, AUX1 packets are very flexible, allowing a variety of algorithm implementations. One of the principal problems with the algorithm presented in this dissertation is that the Bluetooth systems over which the CBMA algorithm is implemented suffer a penalty in throughput. While this penalty is generally limited, and this penalty is tradeoff that is necessary to achieve far larger increases in IEEE 802.11b link performance, any configuration that provides increases in Bluetooth performance with no degradation in IEEE 802.11b performance is welcome. In this section, an adaptive-frequency
hopping/CBMA implementation is presented that is meant to improve Bluetooth performance with no reduction in IEEE 802.11b performance.

### 6.7.1 AFHH - Closed form

One of the surprising aspects of the AUX1 Bluetooth packet is that it can be used to implement an adaptive frequency-hopping (AFH) algorithm. The reasoning behind the implementation is seen in Figure 6.37.

As seen in Figure 6.37, the Master/Slave packets, which are sent in pairs, can be matched to the appropriate frequency. In the Bluetooth standard, the frequency-hopping sequence is determined by a pn-sequence whose structure is known a-priori. The master device controls the piconet clock; with the piconet clock, the master device knows the frequency that will be used by the next packet that the master device will transmit as well as the frequency for the packet that will be transmitted by the slave’s response. Therefore, as seen in Figure 6.37, the master device can predict which slot pair will result in packets sent over frequencies not used by the IEEE 802.11b system with which the Bluetooth system is coexisting.

The expected system performance can be calculated analytically. This analysis assumes that there is no adjacent channel interference in the system. The total bandwidth over which the
Bluetooth system can hop is 79 MHz. At 1 MHz per channel, an expression can be set to

\[ BW_{Total} = 79 \]  

(6.45)

Each IEEE 802.11b channel occupies 22 MHz, so the occupied bandwidth can be set to be

\[ BW_{Occupied} = 22 \]  

(6.46)

The proportion of the band that is not occupied by the IEEE 802.11b system is then

\[ p_{Available} = \frac{BW_{Total} - BW_{Occupied}}{BW_{Total}} = 0.7215 \]  

(6.47)

Therefore, the probability that any one packet to be sent by Bluetooth is \( p_{Available} \). The probability that two consecutive packets will be transmitted over unoccupied channels is

\[ p_{Tx} = p_{Available} p_{Available} = 0.5206 \]  

(6.48)

Given the probability that the two consecutive packets will be transmitted over unoccupied channels, the maximum Bluetooth throughput that can calculated. Given that AUX1 packets using a 16-bit CRC code for error detection can support 172.8 kbps, the AFH maximum throughput can be calculated as

\[ Throughput = 172.8 \cdot 10^3 \cdot 0.5206 = 89.96 \cdot 10^3 \]  

(6.49)

To calculate the link delay, a only a simple observation is needed. Given the way that delay is measured in this dissertation, a delay on a Bluetooth packet is measured from the first instance in time that a packet delivery is attempted to the point in time at which the packet has been successfully delivered. Given this definition, then the delay per packet is 1.25 ms,
since no collisions are possible using this scheme, and hence no delay except for transmission delay is possible on the Bluetooth link. These results have been validated through simulation. Therefore, it is possible for the AFH configuration to be completely transparent to the IEEE 802.11b system yet maintain a Bluetooth throughput of around 90 kbps, which is approximately 50% higher than the worst-case throughput shown in the traffic conditions presented in this dissertation using the CBMA algorithm, and a delay of 1.25 ms.

It should be noted that the above implementation is a backwards-compatible form of adaptive frequency hopping for Bluetooth systems. It should also be noted that what the solution provided in this section gives is a maximum throughput for a system where the frequencies to be avoided are known. In a truly AFH system, the system would need to sample different channels and determine which ones are occupied and which are not. The investigation of this aspect of the implementation of AFH in Bluetooth is beyond the scope of this dissertation.

6.7.2 AFH/CBMA Combination

As seen in the previous section, the AFH implementation shown can provide performance equivalent to the CBMA system. In this section, a method to combine the CBMA algorithm presented in this dissertation with the AFH algorithm presented in the previous section is presented.

Structure

The AFH system is ideally suited for situations in which there is a high risk that certain channels will be occupied. If the channel cannot be assumed to be occupied, or at least that there is a high probability that the channel will be occupied, it makes no sense that there should be any restrictions on the transmission of packets. Therefore, the combination implementation is presented in Table 6.3. The algorithm transmits with no AFH until a collision is detected. When a collision is detected, the system switches to AFH until a timer,
Table 6.3: AFH/CBMA implementation

1. If packet ready for transmission, transmit packet
2. If no collision occurs, go to step 1
3. If a collision occurs, start wait state (set timer to \( t \))
4. If packet ready for transmission, transmit packet using AFH algorithm
5. Reduce timer by 1
6. If timer = 0, go to step 1
7. If timer \( \neq 0 \), go to step 3

set to match the CBMA algorithm’s best guess as to when the channel is most likely to be empty, reaches zero. When the timer reaches zero, the Bluetooth transmission no longer uses AFH. This algorithm is re-triggered whenever a new collision occurs. In short, the AFH algorithm is in effect only during the periods during which there is a strong likelihood that there will be a collision. If there is no indication of a strong likelihood that there will be collision, no restrictions should be placed on the transmission of data.

Moving Scenario

To evaluate the performance of this algorithm, a simulation was run where the combined AFH/CBMA algorithms were implemented on a Bluetooth device where perfect collision detection is implemented. Figure 6.38 shows the throughput of the IEEE 802.11b device in this scenario.

As seen in Figure 6.38, the throughput of the IEEE 802.11b device is flat for all distances simulated between the Bluetooth piconet and the IEEE 802.11b devices. Hence, it is shown that the AFH/CBMA combination has the expected transparent coexistence with the IEEE 802.11b link.

The goal of the combined algorithm is to provide an increase in the Bluetooth link through-
Figure 6.38: IEEE 802.11b throughput when Bluetooth implements combined AFH/CBMA put while maintaining transparency with the IEEE 802.11b link. Figure 6.39 shows the throughput of the Bluetooth devices. As seen in Figure 6.39, the AFH/CBMA combination provides an increase between 40 and 50 kbps over the CBMA configuration. The data rate for the AFH/CBMA combination is somewhere between 105 kbps and 120 kbps. Table 6.4 summarizes the comparison of CBMA, AFH, and AFH/CBMA. As seen in Table 6.4, the minimum throughput of the AFH/CBMA combination provides increases in performance of 75% and 17% versus the CBMA and AFH algorithms, respectively. The maximum throughput of the AFH/CBMA combination provides increases in performance of 50% and 33%, respectively. The AFH/CBMA combination should provide a decrease in mean link delay. As seen in Figure 6.40, the decrease in delay is as large as 1.5 ms. It should be noted that such a delay would have little or no impact on most applications. The AFH/CBMA combination clearly
Figure 6.39: Bluetooth throughput when implementing combined AFH/CBMA in the presence of an IEEE 802.11b link

Table 6.4: Comparison of CBMA, AFH, and AFH/CBMA over all simulated distances

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Minimum Throughput (kbps)</th>
<th>Maximum Throughput (kbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBMA</td>
<td>60</td>
<td>80</td>
</tr>
<tr>
<td>AFH</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>AFH/CBMA</td>
<td>105</td>
<td>120</td>
</tr>
</tbody>
</table>
Figure 6.40: Bluetooth delay when implementing combined AFH/CBMA in the presence of an IEEE 802.11b link

provides a consistent improvement over either the CBMA or AFH algorithm implementations individually.

**Realistic Implementation Scenario**

As in Section 6.6, the ability of the Bluetooth device to detect a collision with an IEEE 802.11b can significantly affect the performance of the CBMA algorithm. In the case where the AFH/CBMA algorithm is implemented, the same issues arise. In Figure 6.41, the throughput of the IEEE 802.11b device simulated in this scenario is shown.

As seen in Figure 6.41, the performance of the AFH/CBMA combination is consistently similar to the performance using the CBMA algorithm only. This result is reasonable, since the detection algorithm is the same in both systems, and the AFH algorithm is invoked only when a collision has been detected by the Bluetooth system.
In Figure 6.42, the throughput of the Bluetooth device is seen for a combination of Bluetooth piconet ranges using the CBMA algorithm and the AFH/CBMA combination. As seen in Figure 6.42, considerable gains are possible when implementing the AFH/CBMA combination in the throughput of a Bluetooth device. All gains need to be constrained by the maximum link throughput of \(172.8\text{kbps}\). This ceiling is evident in the case where the devices are separated by 1\(m\) and the Bluetooth piconet is over 10\(m\) away from either IEEE 802.11b device. In these cases, the throughput of the Bluetooth link is already at or near 172.8\(\text{kbps}\), so an additional increase in throughput is not possible. However, as the throughput of the Bluetooth link drops further from the 172.8\(\text{kbps}\) maximum, the gain achieved by the implementation of the AFH/CBMA combination becomes substantial and consistent.

Figure 6.43 shows the delay simulated for the Bluetooth devices. As seen in Figure 6.43, the delay can be significantly lower than that incurred by the CBMA-only case. Given that the system has a lower bound of 1.25\(ms\) for delay, then no delay decreases below 1.25\(ms\) are
Figure 6.42: Bluetooth throughput when implementing combined AFH/CBMA using errors as a trigger

Figure 6.43: Bluetooth latency when implementing combined AFH/CBMA using errors as a trigger
possible. However, some cases, such as the case where the Bluetooth devices are separated by 10m, the delay is over 2.25ms. In this case, AFH/CBMA combination manages to reduce the delay to around 1.5ms. In this case, the reduction in delay is 0.75ms.

As seen in Figures 6.41 through 6.43, not only does the AFH/CBMA combination perform consistently better than the CBMA-only version for all cases shown, but the ability for the algorithm to detect collisions is not impaired by the combination.

6.8 Summary

This chapter presented a case study of the Bluetooth/IEEE 802.11b coexistence problem as well as the coexistence mechanism presented in this dissertation, CBMA, a form of adaptive frequency hopping (AFH) also presented in this dissertation, and a combined CBMA/AFH strategy, also presented in this dissertation. The CBMA algorithm was shown be able to significantly reduce the impact of a Bluetooth link on an IEEE 802.11b link. Since the CBMA algorithm is based on traffic estimation, the sensitivity of this algorithm to poor estimation; it was shown that while the impact of poor estimation can be significant, the CBMA algorithm never leads to impact on IEEE 802.11b links that is greater than the case in which no coexistence algorithm is added to the Bluetooth system. Given the flexibility of the algorithm introduction method into the Bluetooth protocol, an AFH algorithm is also presented. This algorithm is shown to have comparable performance to the CBMA algorithm. One of the drawbacks of both the CBMA and AFH algorithms is that they have a negative impact on the performance of the Bluetooth link implementing the algorithm. A combined CBMA/AFH algorithm is presented, which is shown to not only have an impact on the IEEE 802.11b link that is not greater than the CBMA-only implementation, but the Bluetooth link throughput is shown to be significantly greater than either the CBMA or AFH implementation alone.
Chapter 7

Conclusion

This dissertation presented a backwards-compatible method to establish a coexistence mechanism between two different types of networks operating at the same time over the same band. The algorithm presented in this dissertation, coined Collision Based Multiple Access (CBMA), is based on the characterization of the traffic of a “weak” system by a “strong” system. The more powerful system, which is capable of disrupting the weak system while sustaining relatively little damage from the weaker system, then implements the CBMA algorithm. By detecting collisions with the weaker system, the stronger system can steer its traffic around the weaker system, performing some form of spectrum sharing between the two systems.

The test systems selected for CBMA are Bluetooth and IEEE 802.11b. The weaker system was deemed to be IEEE 802.11b, since it contains relatively long packets and it uses CSMA/CA, which has a tendency to defer its transmissions to other stronger devices. The Bluetooth system was determined to be the stronger system, since it contains shorter packets coupled with a deterministic, or non-carrier sensing, mechanism. The combination of these two attributes allows Bluetooth to cause significant degradation in IEEE 802.11b performance. The degradation was shown to be capable of reaching approximately 75% while at the same time sustaining relatively little impact in performance.
The CBMA algorithm, when properly tuned to the IEEE 802.11b traffic, was shown to make Bluetooth essentially transparent to IEEE 802.11b, leaving no discernible impact on the IEEE 802.11b link. It should be noted that because AUX1 packets were selected for the CBMA implementation, the algorithm implementation is fully compliant with the Bluetooth standard, which is a claim that none of the suggestions to the IEEE 802.15 Part 2 Working Group, the IEEE group tasked with solving the IEEE 802.11b/Bluetooth coexistence problem, are not capable of claiming. This transparency was achieved at a cost in throughput and delay for the Bluetooth device. While the added delay was negligible, under 2ms, the penalty in throughput was significant, on the order of 40%. However, while the reduction in Bluetooth throughput would best be measured in kbps, the increase in throughput of the IEEE 802.11b device could best be measured in Mbps. From a system-wide standpoint, the relatively low cost in the reduction of the Bluetooth throughput led to a significant increase in the aggregate throughput of all the networks. This rationale can be misleading, since an increased aggregate throughput can be achieved through the termination of the Bluetooth link. However, in the case of the CBMA implementation, the reduction in throughput of the Bluetooth device increases the IEEE 802.11b throughput while maintaining a generally acceptable Bluetooth throughput.

One of the penalties that is paid by the implementation of the CBMA algorithm is that it can reduce the effective range of a device. Since a delay is injected when bit errors are detected, an increase in BER due to a reduction in SNR will lead to a disproportionately high decrease in throughput. However, the penalty in range for the Bluetooth implementation of the CBMA algorithm was shown to be approximately 10%. While the reduction in range can be noticeable, it is not sufficiently significant to negate the benefits of the implementation of the CBMA algorithm. Furthermore, in this dissertation, it was shown that two links using the CBMA algorithm are not likely to affect each other much. The reason for this is that the CBMA algorithm is implemented over the already existing coexistence mechanism. This coexistence mechanism is designed to allow the same type of devices to operate in the same environment at the same time. Injecting CBMA over this existing mechanism does
not reduce the existing mechanism’s capability of avoiding collisions, but it enhances the device’s ability to coexist with other standards.

The detection threshold of the algorithm became a key issue of the simulation. Given that the algorithm can function only if a collision is detected, the likelihood of a detection is important. This detection likelihood is a function of the relative range of the Bluetooth and IEEE 802.11b devices. It was proven that a Bluetooth separation of 5 meters, approximately 15 feet, is sufficient for the algorithm to have a strong likelihood of detection and collision avoidance.

Traffic estimation is a key aspect of this dissertation. Estimating the traffic of a network is, at best, a difficult task. However, it was proven through simulation that the CBMA algorithm, when implemented on Bluetooth devices to avoid IEEE 802.11b device traffic, can be fairly robust and still perform the coexistence functionality even when the estimation is off by a large margin. The primary reason for this robustness is that, while traffic is difficult to estimate, there are several aspects of a link protocol that are not random, and this deterministic behavior is the behavior that allows the coexistence algorithm to still function. Even though the algorithm functions when the traffic estimate is inaccurate, it should be noted that the performance is best when the algorithm is tuned closely to the actual traffic.

As a part of the research focus of the dissertation, a mechanism was developed using AUX1 packets to implement a backwards compatible form of adaptive frequency hopping on Bluetooth devices. The performance of this algorithm was evaluated, showing that it has a comparable performance to the CBMA implementation. An enhancement of the CBMA algorithm was developed, where the CBMA algorithm was combined with the adaptive frequency hopping algorithm. This combined implementation was proven to be superior to either the CBMA or adaptive frequency hopping implementations separately.

A summary of the original contributions of this dissertation is presented in Table 7.1.
Table 7.1: Dissertation Original Contributions

- Stochastic analysis of on-off traffic source in terms of channel occupation
- Demonstration that stochastic traffic solution can lead to an optimal transmission point in a link with known traffic patterns
- Demonstration that an optimal transmission point in a link with known traffic patterns can significantly reduce the probability of collision between two networks with different packet/channel structures
- Method to implement algorithm that is Bluetooth backwards-compatible yet flexible enough to allow timing and error control over a link
- Analysis of IEEE 802.11b traffic when using heavy-tailed on-off traffic sources
- Demonstration of functional Hybrid ARQ on Bluetooth device
- Effective CBMA implementation on Bluetooth/IEEE 802.11b environment
- Backwards-compatible implementation of adaptive frequency hopping version of Bluetooth
- Combined CBMA/Adaptive frequency hopping algorithm for coexistence
7.1 Future Work

Several aspects of this dissertation are open for future research. The future research topics are broken down into: IEEE 802.11 implementation, parameter estimation, protocol estimation, multi-source behavior, and channel estimation.

7.1.1 IEEE 802.11 Implementation

One of the recent developments relating to IEEE 802.11 is the development of IEEE 802.11e, which is a QoS-aware version of the IEEE 802.11 MAC. IEEE 802.11e would be a MAC variant that is applied to IEEE 802.11b/a/g, so versions of this standard would operate in the 2.4GHz ISM band and 5GHz UNII band. IEEE 802.11e manages QoS guarantees by managing the relative priority of each different type of traffic that the link is managing. The management of priorities is performed by managing aspects like the minimum and maximum contention window sizes for the different types of traffic [66]. If it is possible to manage the timing in different traffic sources, it may be possible to implement the CBMA algorithm presented in this paper over IEEE 802.11e. It is suggested that the CBMA algorithm be applied to IEEE 802.11e to manage interference from Bluetooth devices. An interesting point to note is that since IEEE 802.11 is based on the concept of giving other devices the opportunity to transmit, it may be possible to create a more aggressive version of IEEE 802.11 such that it is capable of better dealing with Bluetooth interference.

7.1.2 Rate-varying IEEE 802.11

In this dissertation, the device model used for IEEE 802.11b is the 11 Mbps variant of the standard. It is unclear if, when in a high-SNR environment, the IEEE 802.11b device is likely to change its data rate when faced with Bluetooth interference. It is suggested that, as further work, algorithms that perform rate switching be investigated in the presence of Bluetooth interference.
interference to determine not only the likelihood of switching, but also the result of such switching. For example, rate switching based on received signal strength, such as the RSSI, may not prompt a switch in rate, since an increased in the received aggregate signal (from the Bluetooth device) would be misinterpreted as a good-signal condition. Furthermore, enhanced algorithms that can perform this switching so that it is optimized to environments prone to interference may be possible, reducing the impact of interference on the IEEE 802.11b link.

### 7.1.3 Parameter Estimation

The core of this dissertation is based on traffic estimation; this traffic estimation can be broken down into two separate pieces: source estimation and standard estimation. Source estimation is just an estimate of the traffic source that is feeding the system that is transmitting data. In this dissertation, the source is reduced to an on/off traffic source. The CBMA algorithm is implemented using perfect knowledge of the traffic source. While the robustness of the CBMA algorithm’s performance to changes in the traffic source was shown, the performance of the algorithm is best when the traffic source is accurately estimated.

Given that the collision avoidance mechanism is intrusive, i.e. through the detection of collisions, it follows that if the same method is used to establish the traffic of devices in the local neighborhood, the measurements themselves will affect the traffic that is intended to be measured. A reliable method to determine the traffic of a network where the measurement method itself changes the traffic is required. This problem may sound difficult to surpass, since the original traffic would never be known if the measurements themselves affected the system traffic. However, since the only behavior that matters is the one that occurs after a collision, it follows that a sampling methodology that is disruptive may not cause insurmountable problems.

Therefore, it is suggested as future work that a method be devised to determine the traffic characteristics of a different network through collision detection. Variables that should be
determined are the expected lifetime of the measurements, the level of uncertainty inherent to the measurement system, and a more descriptive measure of how precise the measurements need to be for the algorithm to function properly.

### 7.1.4 Protocol Estimation

In Section 7.1.3, it is suggested that future research investigate how to determine the statistical characteristics of on/off traffic sources. An associated suggested research topic is to determine the protocol that the other network is using. The CBMA algorithm’s robustness is largely based on the non-stochastic aspects of the link. Therefore, more than traffic estimation, protocol estimation is important. Two aspects of protocol estimation need further research. First, the common trait between different sampling instances of a protocol need to be researched. This means that the behavior of a link that is not a function of the traffic pattern, the aspects that are common to the device regardless of the traffic source, need to be established. There are dual benefits to this estimation. First, the longevity of traffic measurements can be extended. Since these aspects of the source are non-varying, as long as the link protocol is the same, the lifetime of the traffic estimates can increase. This increase in longevity is probably best manifested as a resilience to on/off source estimation, as seen in Chapter 6.5.1 of this dissertation.

It should be noted that the IEEE 802.11b variant that is under study in this dissertation is the 11 Mbps variant, which is just one of the four possible IEEE 802.11b transmit speeds. Each of these variants has different behavior characteristics, such as the maximum or expected length of different parts of the packet. This issue leads into the other related suggested research topic: the sensitivity to protocol estimation. For the near future, it is a reasonable assumption to expect the traffic that is to be avoided to be IEEE 802.11b. However, IEEE 802.11g, a faster version of IEEE 802.11 also operating over the ISM band, will probably see deployment soon. As the success of protocols increases, the likelihood of more protocol variants for specific applications and market niches to be developed and deployed in the un-
licensed bands increases. In such cases, the likelihood that the same protocol is encountered on a consistent basis decreases.

7.1.5 Multi-source Behavior

The behavior of the CBMA algorithm when multiple Bluetooth devices are implementing the algorithm is shown. One aspect that has been left open for future research is what happens when multiple devices sharing the spectrum using the same protocol are the devices that are to be avoided. In the context of this case study, it was shown what happens when multiple Bluetooth devices are implementing the CBMA algorithm, but what was not shown, and what is suggested for future research, is what happens when multiple IEEE 802.11b links are the sources that the Bluetooth-based CBMA algorithm should avoid.

The goal of estimating the condition of the channel when multiple devices using the same protocol are populating the channel may sound chaotic, but there are reasons why it is expected that a solution is possible. The key in the search for a solution lies in the fact that all protocols are bound by some deterministic behavior. For example, in protocols like IEEE 802.11b, all devices that can hear the local traffic are aware of any errors that may occur in a link and respond accordingly, as seen in Section 2.2.2. When an error is detected by an IEEE 802.11b device not directly involved in the data exchange, the device invokes an extended wait period followed by a contention window process. This behavior is meant to make sure that the device that detected the error allows the recovery process to begin between the devices over whose link an error occurred. This behavior is a form of synchronization between the devices over whose link an error occurred and all the other devices in the local environment. This synchronization means that if the reference source for synchronization of the CBMA algorithm is the link that just had an error, every other device in the network will be synchronized, allowing the CBMA algorithm to synchronize every single device that might share the channel.

While it is not unfeasible for the multi-source behavior of a network to remain compatible
with a CBMA implementation, the relative ease of this capability needs to be established, as well as the likely tradeoffs that are to be expected. These research questions are left open as suggestion for future research.

### 7.1.6 Channel Estimation

In this dissertation, an adaptive frequency hopping algorithm was presented that is backwards-compatible. Furthermore, a method to combine the adaptive frequency hopping algorithm with the CBMA algorithm was also presented. One assumption in both the adaptive frequency hopping variations presented in this dissertation is that the channels that are to be avoided are known by the Bluetooth device. It is suggested that future research be performed to determine methods to establish which channels are available and which channels are not available. Parameters that should be investigated include the lifetime of the estimate and the expected convergence time of the estimate, i.e. how long it takes for an estimate of the available channels to be created.

### 7.2 Suggested Publications

Much of the research presented in this paper has not been published yet. The journal papers that are suggested for future publication are,

- Backwards-compatible coexistence strategies for Bluetooth/IEEE 802.11b environments
- Accelerated simulation strategies for wireless networks
Bibliography


Max Robert was born on April 24, 1973, in Buenos Aires, Argentina. In 1991, he received the Albert W. Smith Scholarship at Case Western Reserve University in Cleveland, Ohio, where he received his Bachelor of Science Degree in Electrical Engineering and Applied Physics in May of 1996. During his undergraduate program, he participated in internships at Westinghouse Electric in Chuchill, Pennsylvania, and co-ops at Sensis Corporation in DeWitt, New York and Keithley Instruments in Cleveland, Ohio. In the fall of 1996, he joined the Master of Science Program at Virginia Tech, culminating in earning a M.S. degree in Electrical Engineering in 1998. Since 1997, he has been a member of the Mobile and Portable Radio Research Group at Virginia Tech. In 1999, he entered the Ph.D. program, also at Virginia Tech. In 1999 he received the Bradley Graduate Fellowship, and in 2001 he received the America Online Wireless Home Networking Technologies Fellowship. His research interests include system design, system modeling and simulation, cross-layer optimization, software-defined radio, wireless local and personal area networks, and software design. Max Robert’s publications include:


- F.Bendeck, B.Kotting, M.Schaaf, F.Maurer, M.Valenti, and M.Robert, "Engineering of e-Business Applications and Infrastructure and Applications for the Mobile Internet," *Pro-

