CATER:
AN OPPORTUNISTIC MEDIUM ACCESS CONTROL PROTOCOL FOR WIRELESS LOCAL AREA NETWORKS

Barry E. Mullins

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

Dr. N. J. Davis IV, Chairman
Dr. C. W. Bostian
Dr. S. F. Midkiff
Dr. F. G. Gray
Dr. M. Abrams

23 June 1997
Blacksburg, Virginia

Keywords: IEEE 802.11, Wireless LAN, Spread Spectrum, Adaptive MAC Protocol

CATER:
AN OPPORTUNISTIC MEDIUM ACCESS CONTROL PROTOCOL FOR WIRELESS LOCAL AREA NETWORKS

Barry E. Mullins

Dr. Nathaniel J. Davis, IV, Chairman
Electrical Engineering
(ABSTRACT)

The past decade has seen an explosive growth in wireless products such as cellular phones and pagers. The zeal with which consumers are using these products and services is a testament to our society's desire to remain in touch while on the move. Computing systems are now reaping the benefits of this quest for mobility. Since their inception in the late 1970's [FrE80, GfB79], wireless local area networks have been thrust to the forefront of networking alternatives especially when confronted with difficult or impossible obstacles sometimes associated with wired networks such as routing cables and spontaneous *ad hoc* LANs.

Before a wireless LAN is considered, system capabilities, cost, and limitations must be given due consideration. To gain user acceptance, a wireless LAN must sustain data rates comparable with wired networks, gracefully control the inherently larger number of bit errors, and allow inter-operability with heterogeneous vendor products. Pivotal to network performance, the medium access control (MAC) is the protocol used by all stations within the network to control when each station is allowed to transmit and ultimately determines the degree to which the aforementioned factors satisfy the user. Current wireless LANs are artificially limiting their performance by using a wired LAN MAC model. This "worst case" model assumes that once the medium suffers a performance degradation, it cannot be improved.

An adaptive MAC protocol is developed and analyzed that offers a "best case" scenario by allowing the MAC to control medium parameters thereby fully exploiting the channel of an *ad hoc* wireless LAN. This new, opportunistic medium access control protocol is called CATER (Code Adapts To Enhance Reliability) and is based on the proposed MAC standard for wireless local area networks (WLAN)—IEEE 802.11 [IEE96]. As currently proposed, IEEE 802.11 uses a fixed pseudo-noise (PN) code for spreading the information signal, implying a fixed process gain at the receiver. When the channel degrades, IEEE 802.11 offers only retransmissions at the MAC layer to combat a corrupt medium.

However, CATER allows communicating stations to reconfigure their transceivers to use a longer PN code after a prescribed number of failed retransmissions. This longer code increases the process gain of the receiver and reduces the error rate. After the two stations are reconfigured, the source station sends the frame in question. Immediately after that frame is acknowledged, the source station may send additional frames during the reconfigured period.

Simulation and emulation are used to demonstrate and validate the adaptive protocol's capabilities. Results show that this new protocol offers substantial improvement in system throughput when the channel degrades to a point that reliable transmission of frames is not feasible in a standard IEEE 802.11 WLAN. Specifically, CATER continues to function, permitting up to 14 percent normalized aggregate throughput at times when IEEE 802.11 permits *no* frames to pass through the WLAN. In addition, throughput experiences only a small decrease due to protocol overhead during periods when stations experience a good channel with few bit errors. Moreover, CATER does not adversely affect the predominate transport layer protocol (i.e., TCP), and provides equitable service to all stations within the network.

Acknowledgments

As in any major effort, no one person can accept full responsibility for success. This dissertation has profited from the guidance and support of several individuals over the years, and I wish to thank those responsible.

I thank my advisor, Dr. Nathaniel J. Davis, IV for accepting me as one of his Ph.D. students. His guidance and support are greatly appreciated. I thank him for always being available to answer my questions. His efforts were instrumental in the success of this research.

I also thank my committee members, Dr. Scott F. Midkiff, Dr. Charles W. Bostian, Dr. F. Gail Gray, and Dr. Marc Abrams for their guidance and assistance. Each shaped the research into the success it is today. I am particularly indebted to Dr. Midkiff for his exceptional guidance and support. He offered valuable suggestions on ways to improve the research. I thank Dr. Bostian and his Center for Wireless Telecommunications staff for providing an excellent working environment within one of his labs.

I thank my office mates for their support and assistance. Todd Fleming was an invaluable asset to this research in that he answered numerous questions regarding the C programming language and the Microsoft Visual C++ compiler and for this I am thankful. Dr. Boris Davidson provided insight into the characteristics of wireless channels. Others promoted a relaxed environment wherein help was never too far away; these people are Rusty Baldwin, Aaron Hawes, Raza Shah, Jeanette Mulligan, Theodoros David, Donald Cuffee, and John Stanhope.

I thank my parents for understanding the importance of a college education and providing support throughout my undergraduate studies. This understanding and support have, in a large part, made me who I am today. Thanks mom and dad.

Thanking my family would simply be too shallow—my deepest thanks go to my loving family. Besides being a pillar of support, my wife gave me another beautiful daughter during our stay in Virginia. Her love and understanding was a major part of my success. I thank my daughters, Caitlyn and Rebecca, for trying to understand that daddy had to go to work and couldn't play with them as often as I wanted. I am looking forward to normal weekends with all three of you!!

Table of Contents

List of Figures

List of Tables

Chapter 1. Introduction

Men are only as good as their technical development allows them to be.
—George Orwell (1903–50), British author.

Learning without thought is labor lost.
—Confucius

1.1 Background

Communication is a fundamental need of humans which has manifested itself in our computer systems. The evolution of computers from the mainframe monolith to the slick, lightweight laptops has included numerous enabling technologies. Computer communication is on the brink of a new and fascinating revolution thanks in part to the commercialization of spread spectrum technology. This revolution is leading computing in a completely new direction away from the confines of a cramped laboratory or office. This revolution promises to take computers to new locales while maintaining connectivity with other computers (and other people). This revolution is breaking the traditional mindset of a machine tethered to a costly wired network. This revolution is called mobile computing and is destined to transform the way people use computers.

The past decade has seen an explosive growth in wireless products such as cellular phones and pagers. The zeal with which consumers are using these products and services is a testament to our society's desire to remain in touch while on the move. Computing systems are now reaping the benefits of this quest for mobility. Since their inception in the late 1970's [FrE80, GfB79], wireless local area

networks have become better known and, consequently, are seriously considered as options for interconnecting computers within small areas.

Traditional, terrestrial (wired) local area networks (LANs) have enjoyed many successes including widespread adoption, decreasing system component costs, and standardization by the IEEE. In fact, LANs have progressed to the point that home nets (LAN within a residence) supporting a 10 Mbps data rate are no longer an oddity.

Wired networks are not, however, without disadvantages. Moving a networked computer typically involves an expensive rerouting of the LAN cable. In some cases rerouting is not feasible (e.g., presence of asbestos in a building could prohibit remodeling). Establishing a spontaneous LAN for a meeting is virtually unknown. These limitations and recent advances in wireless LAN products have spurred many network administrators to consider wireless LANs. However, before a wireless local area network is casually purchased, system capabilities, cost, and limitations must be considered. To gain user acceptance, a wireless LAN must support data rates comparable with wired networks, gracefully control the inherently larger number of bit errors, and allow inter-operability with heterogeneous vendor products.

Designing a wireless LAN involves many decisions affecting system performance. The inter-node distance dictates the transmission medium and the power requirements of each node. These requirements can be satisfied via three media categories: infrared light, unlicensed spread spectrum radio, and licensed radio. Infrared cannot penetrate opaque objects, limiting its usefulness to a single room. Unlicensed spread spectrum radio performs admirably in the face of interference but its transmitting power is limited to less than one watt thus limiting range. Licensed radio overcomes the aforementioned limitations but requires a license, which is becoming more difficult to obtain given spectrum crowding.

The medium access scheme is another consideration when developing a wireless LAN. Since each node must gain control of the medium to successfully transmit data, the medium access scheme is pivotal to network performance. Within the wireless LAN realm, three access schemes have received the most attention. These are time division multiple access (TDMA) in which each computer is allotted a time slot to transmit; the ALOHA access family which allows a computer to transmit data as soon as it wants; and carrier sense multiple access (CSMA) which forces a computer to first determine the status of the medium before transmitting.

Current wireless LANs are artificially limiting their performance by using a wired LAN model at the link layer which is the LAN layer responsible for the error-free link between two directly connected stations. This “worst case” model assumes that once the medium suffers a performance degradation, it

cannot be improved. The adaptive protocol research presented in this dissertation offers a "best case" scenario by allowing the link layer to control medium parameters thereby fully exploiting the channel.

1.2 Research Goals

The primary goals of the research effort defined in this document are three-fold. First and foremost is to extend the body of knowledge relative to performance enhancements of wireless LANs via the adaptive, opportunistic control of the link layer interfaces. Specifically, a new medium access control protocol is developed to allow communicating wireless stations to reconfigure the connection between the two at times when the link is degraded, thus effectively increasing the reliability of the link. The protocol dynamically varies the data rate via the pseudo-noise (PN) code when the channel becomes corrupted. This research investigates the effects of adaptively varying the PN code length, thus providing a trade-off between data rate and error rate, to counteract a poor link between communicating stations. The second goal is to optimize the protocol given a certain network configuration. The final goal is to emulate the protocol on a wireless LAN testbed.

These goals are achieved in two ways. First, a simulation model of the proposed system is developed and validated against analytical models of similar systems. This simulation provides valuable insight into system performance under various operating conditions. Thus, network performance can be gauged with and without adaptive control. The relative performance improvements and under what circumstances these occur are learned during this research which facilitates protocol optimization.

The second means of achieving the goals is though emulation of the new protocol on a wireless LAN testbed. The testbed consists of two personal computers networked via Ethernet. Simulation results are validated using this emulation.

To date, adaptively varying the PN code to mediate a failing or corrupt channel between two wireless stations has not been investigated. This research investigates this unexplored area and is a first step in developing *wireless networks* instead of networks running over wireless media where *wireless networks* are networks designed specifically for wireless media.

It is shown that an adaptive protocol can be used to counter the effects of an unreliable medium. A simulation of this protocol is discussed as well as results of simulating the protocol model. The manner in which factors affect system performance is discussed and used to optimize the protocol for the wireless system modeled.

1.3 Document Overview

This chapter provides an introduction to the problem under investigation. Specifically, the importance of providing a reliable link between communicating wireless LAN stations is discussed. A brief overview of the merits and pitfalls of wireless LANs is presented. Design considerations are presented as well as why wireless LANs are becoming viable options.

Background information is the subject of Chapter 2. Information pertinent to the development of wireless LANs is presented. A complete understanding of wireless LANs first requires insight into the operation and organization of wired LANs; this chapter provides this insight by presenting the OSI layered model of LANs, LAN topologies, and medium access control techniques. Performance of the medium access control techniques is presented and compared. It is seen that using a carrier sense mechanism significantly improves network throughput. Moreover, adding a collision detection capability further increases throughput. However, collision detection is not always feasible; implementing a network using radios renders the collision detection capability impractical. Therefore, another technique known as collision avoidance is proposed by IEEE 802.11 to enhance throughput. Chapter 2 discusses this technique as implemented by IEEE 802.11 as well as other nuances of its implementation. An enabling technology for wireless LANs is spread spectrum techniques. Chapter 2 presents a brief overview of the specific spread spectrum techniques used in this research effort. Specifically, characteristics of spread spectrum systems that enable them to perform in adverse conditions is discussed. Chapter 2 concludes with a discussion of related research efforts.

The research objectives and methodologies are the topic of Chapter 3. Beginning with a problem definition that includes bounding assumptions for the network, this chapter delineates the scope of the research. Research methodology is presented as well.

Chapter 4 provides the first look at the adaptive protocol called CATER (Code Adapts To Enhance Reliability). Its operation is described in detail in addition to examples of the protocol in action. It is seen that CATER attempts to transmit frames using native 802.11 until the connection between two stations becomes suspect. After a prescribed number of failed transmission attempts CATER reconfigures the link between the two stations and retransmits using a longer PN code.

The simulation environment is presented in Chapter 5. Starting with a description of the simulation tool, this chapter completely describes the simulation. Simulation parameters and factors are defined and assigned values. Aggregate throughput is defined as the response variable of interest for the simulations. The chapter continues with a presentation of the computing platform used during simulations as well as verification and validation efforts.

Chapter 6 presents the results of the simulation phase of this research effort. It is shown that CATER performs better than standard IEEE 802.11 when considering the dynamic nature of wireless links. Aggregate throughput for a wireless LAN is improved when using the CATER protocol. This is demonstrated using both static BERs and dynamic BERs. This chapter demonstrates that CATER does not allow asymmetric use of the channel; all stations are allowed a fair share of the medium. The behavior of TCP was shown to be acceptable when using CATER; TCP retransmissions are kept to a minimum (less than 3 percent of the time for worst case). Simulation results indicate CATER offers a substantial improvement in aggregate throughput during periods of high BER while suffering only a slight decrease in throughput during periods of low BER. The CATER protocol offers an alternative to the rigid IEEE 802.11 standard that promises a more robust protocol and, thereby, wireless connection, which is the ultimate goal of any wireless LAN.

The emulation is presented in Chapter 7. The emulation environment is discussed in detail along with the experimental design and results. This chapter culminates with the validation of the simulation using the emulation data.

Chapter 8 concludes the dissertation with a summary of the research including the significance of research results. Finally, the chapter offers recommendations for future work within this research area.

Chapter 2. Background and Literature Survey

Those who cannot remember the past are condemned to repeat it.
—George Santayana (1863–1952), U.S. philosopher, poet.

Every time history repeats itself the price goes up.
—Anonymous.

Wireless local area networks must contend with several issues to perform at a level satisfactory to end users. This chapter examines these issues. Section 2.1 presents a review of local area networks (LANs), how they operate at the different layers of the OSI model, and how they provide connectivity. Various medium access schemes including ALOHA and carrier sense multiple access (CSMA) are discussed and compared; throughput performance of each scheme is addressed. Spread spectrum technology is presented in Section 2.2; a review of its history and capabilities is presented with emphasis on direct sequence spread spectrum. Section 2.3 is dedicated to wireless LANs. A summary of wireless LAN standardization efforts is presented in Section 2.3.1, then the proposed IEEE standard for wireless LANs, IEEE 802.11, is explained in Section 2.3.2. The medium access control mechanism and physical layer specifications are discussed. A review of relevant research in this field is presented in Section 2.4 followed by a chapter summary in Section 2.5.

2.1 Computer Communication Networks

The successful marriage of telecommunications and computer technology over the past two decades has brought a phenomenal and exciting capability to our society—computer communication networking. Mass production of computer communication equipment (e.g., modems and network

adapters) and the associated improvements in manufacturing techniques have driven their price down to a point of making them standard options on most computers sold today. In addition, the recent popularity of the World Wide Web (WWW), the growing number of Internet service providers, the improving cost/performance ratio of computer communication equipment, and the ease with which the general population can assess these has heightened interest in computer networks. The ultimate goal of these networks, in their quest to satisfy user's unprecedented demand for information, is the instantaneous transmission of error-free data from one location to another. To this end, a network must be able to efficiently accommodate the dynamic, bursty nature of data transmissions and bit errors over links of varying reliability.

Computer networks can be cataloged into either a Local Area Network (LAN), a Metropolitan Area Network (MAN), or a Wide Area Network (WAN) [BeG92, Kei89, SaI95]. Since all networks are formed by interconnected media in one form or another, a good differentiating characteristic of these networks is distance covered. LANs typically provide service to a relatively small (i.e., less than 10km) area such as an office building or a campus. MANs extend the service area to a few hundred kilometers, while WANs extend coverage to the entire world [Kei89, SaI95, Tan88]. Beyond distance, LANs support higher data rates (i.e., on the order of 1 to 100 Mbps), provide lower error rates, and rely on a multi-access channel [Kei89, Tan88]. Since this research effort focuses on LANs, it is the only network discussed henceforth.

2.1.1 Local Area Networks

LANs are exploding in popularity due, in part, to the same reasons mentioned above as well as the proliferation of available information sources on the Internet [Kei89].

LANs are used to interconnect computers and peripherals to form systems that make the individual components (e.g., personal computers, workstations, printers) more useful—to form a gestalt. The successful integration of these devices (often called nodes or stations) to form a LAN is strongly tied to the adoption and implementation of a governing body and resulting standards. Proprietary technologies will not survive in this "open" market.

2.1.2 Layered Network Architecture

Nodes must agree on how and when they are allowed to transmit and listen. The framework for accomplishing this is spelled out in a well-defined model developed by the International Standards Organization (ISO) in 1978 [DaZ83]. The model is called the Open System Interconnection (OSI) Model

and establishes a set of standards all manufacturers of communication hardware and software must abide by in order to interconnect with other computers reliably.

Figure 2-1 shows how the model is divided into seven layers. When developing the OSI model, the ISO realized the complexity of specifying an all-encompassing standard for all present and future communication systems. Therefore, using sound engineering practices, the ISO subdivided the task into manageable pieces—the seven layers [Kei89]. This layered approach has stood the test of time by specifying a modular framework within which industry can work. Each layer is considered a black box that provides services to the box above and requests services from the box below. The modular characteristic of the model allows layers (black boxes) to be continually updated as technology and demand warrant. Naturally, any new implementation of a layer must still satisfy the OSI model.

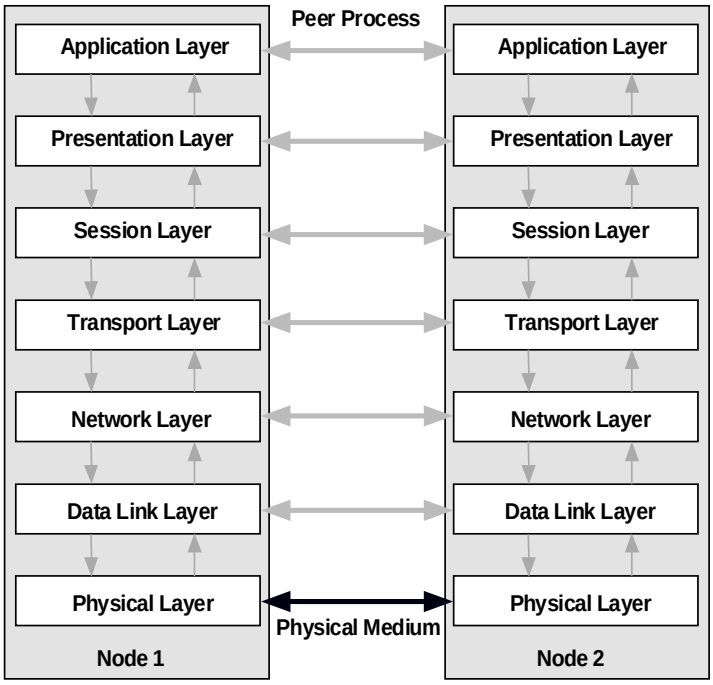

Figure 2-1. OSI Seven Layer Model

In order for communication to occur, each OSI layer must have an associated peer process on another node. This simply means that the transport layer of node 1 must use the same protocol as the transport layer of node 2 [BeG92]. These peer processes are illustrated in Figure 2-1.

2.1.2.1 The Physical Layer (Layer 1)

Often called a virtual bit pipe [BeG92], the physical layer is concerned with the actual medium connecting the nodes. This is the only layer specifically concerned with hardware almost to the exclusion of software-implemented functionality [Col90]. As such, it regulates the electrical, mechanical, functional and procedural characteristics of the link between nodes [Col90, DiL92]. Its fundamental task is to accept a stream of bits from the data link control layer and generate the corresponding signals on the medium. In some instances, the physical layer provides collision detection for the upper layers; this aspect of the physical layer is addressed later in the chapter.

2.1.2.2 The Data Link Control Layer (Layer 2)

The function of the data link control (DLC) layer is to transform the unreliable bit pipe at layer 1 into an error-free link [Kei89] by establishing, maintaining, and releasing connections between two directly connected nodes within a network [Con83]. Since all media experience some degradation in performance in the form of bit errors, the data link layer must be able to handle these errors gracefully. To this end, this layer handles the initialization and termination of the links as well as handling some error detection and correction. It transforms a long sequence of bits into smaller frames of data via fragmentation and then brackets the data with pre-established beginning and ending flags [DiL92]. These flags are used by the receiving node to delineate the bounds of the frame and provide synchronization. The data link layer uses flow control on a link-by-link basis to ensure a node is not overwhelmed by a faster transmitting node.

2.1.2.3 The Network Layer (Layer 3)

The function of the network layer is to provide routing and flow control primarily for wide area networks. This layer is responsible for providing a path for the end-to-end transfer of data [DiL92] by either accepting a packet from the network or forwarding it on to another node as governed by a routing table or other routing information. This layer also provides flow control in the form of congestion control. It is responsible for the introduction of packets into the network in such a fashion that all nodes are provided equitable service [BeG92].

As mentioned, the network layer provides services essential for wide area networks. If a LAN is used in which all nodes share the same medium, routing is a trivial matter. All nodes hear any transmissions on the medium; therefore, routing is not required. Flow control is also handled at a lower layer (data link) in a LAN. Consequently, the network layer is of little consequence in a LAN [BeG92].

2.1.2.4 The Transport Layer (Layer 4)

The transport layer is responsible for the reliable end-to-end transport of data, error recovery and flow control by providing an idealized, full-duplex connection between two end systems [DiL92]. Whereas the three lower OSI layers are responsible for providing data transfer between nodes (physical machines some of which may be intermediate points on the way to the final end system), the transport layer provides data transfer between user processes (end systems) [Kei89].

The transport layer accepts messages from an upper layer. If the message is longer than a prescribed length, the transport layer breaks it into smaller packets such that the newly formed packets stand a better chance of traversing the network. Even with smaller packets, the chance of a corrupted packet exists. Since the service provided by the network layer is often unreliable, the transport layer handles packets arriving in error or out of order.

The bottom four layers of the OSI model form a foundation for all data communications by providing a high-integrity data transfer capability between end systems [Col90, DiL92]. These layers, in some form or another, must be present in order to reliably send and receive data over a network. The upper layers (layers 5, 6, and 7) are not always required. Their inclusion in a particular network depends on the applications being run and the configuration of the network. However, simply sending binary data over a virtual bit pipe is of no use if the receiving system does not know how to interpret the data or what to do with it. The upper three layers address these questions and more.

2.1.2.4.1 Transmission Control Protocol/Internet Protocol

The Transmission Control Protocol/Internet Protocol (TCP/IP) is an example of a transport and network layer. Officially called the TCP/IP Internet Protocol Suite, it encompasses a family of network and transport layer protocols. Introduced in the 1960s as a packet switching research project for the government, TCP/IP has evolved as the most widely used type of networking [Ste94]. In fact, it forms the foundation for the Internet, which provides internetworking to heterogeneous networking systems around the world [Com95]. As a transport layer protocol, TCP provides for the reliable flow of data between two hosts by using packet-level acknowledgments, fragmentation, and timeouts. IP is the companion network layer for TCP, which routes packets throughout the network.

2.1.2.5 The Session Layer (Layer 5)

The session layer is tasked with establishing, maintaining, and terminating a connection between two application processes (end systems) [Col90, Kei89]. This may entail checking access rights of users requesting permission to use a resource. Although the session layer seems to duplicate network layer

functions, there are differences. The session layer operates on end-to-end connections whereas the network layer works on link-to-link connections.

2.1.2.6 The Presentation Layer (Layer 6)

Data encryption, data compression, and code conversion are handled at the presentation layer [Kei89]. While not essential for all data transfers, these services may be useful in a specific situation. Data encryption is the transformation of data into unintelligible gibberish to avoid possible compromise of sensitive information. Data compression reduces the necessary bandwidth of the channel by reducing the total number of bits required to convey the same information. Not all devices use the same format or code to represent data which necessitates the need to convert code formats to fit the needs of the equipment in use. The presentation layer handles this code conversion.

2.1.2.7 The Application Layer (Layer 7)

The application layer is the interface between the network and the application processes. This is to say that all application programs and processes access the network layers via the application layer. As such, the specific protocols used at this layer are dictated by the user. Network users typically select the application layer protocol to be used, whereas all lower layer protocols are selected by network designers [Kei89].

2.1.3 IEEE LAN Layers

The openness of the OSI model offers countless layer implementations. This in itself made the model a powerful tool. However, with power came complexity and inconsistent implementations. To alleviate potential inconsistencies, the IEEE created the IEEE 802 standard family to address how protocols at the bottom two layers operate in a LAN environment [Kei89, DiL92]. Specifically, implementation-specific protocols were constructed for different mediums and topologies at the data link and physical layers. The IEEE 802 standards [Kei89] are listed in Table 2-1.

The relationship between some of these standards and the lower layers of the OSI model is illustrated in Figure 2-2. The IEEE committee subdivided the data link layer into two sublayers to facilitate LAN implementations. The sublayers are the logical link control and the medium access control. As a whole, these two sublayers are logically equivalent to the data link control. Consequently, instead of two lower layers, the DLC and physical layer, the IEEE LAN model consists of three layers.

Standard 802.1 specifies how the other 802 standards interrelate and how these standards relate to the higher layers in the OSI model and is primarily concerned with general and architectural issues [Col90]. Standard 802.2 resides at the top of the three-layer hierarchy and is responsible for addressing

and logical link control [Sta91]. 802.2 is a generic standard completely independent of the topology, medium, and medium access control scheme of the LAN [DiL92]. At the next layer below 802.2 are specific implementations of topologies and media. Standards 802.3, 802.4, 802.5, and 802.6 all dictate the requirements of a medium access control and physical media. As Figure 2-2 shows, these standards must still abide by 802.2 requirements.

Table 2-1. IEEE 802 Standards

Standard #	Description
802.1	Overview, internetworking, and systems managemen
802.2	Logical Link Control (LLC)
802.3	Carrier Sense Multiple Access with Collision Detection (CSMA/CD)
802.4	Token Bus
802.5	Token Ring
802.6	Distributed Queue Dual Bus (DQDB) subnetwork of a Metropolitan Area Network
802.7	Advisory group for broadband cable medium (not a standard)
802.8	Advisory group for fiber optics cable medium (not a standard)
802.9	Integrated Services LAN (ISLAN). Allows integrated voice and data LANs
802.10	Interoperable LAN Security
802.11	Wireless LANs (proposed)
802.12	Demand Priority LAN (DPLAN)

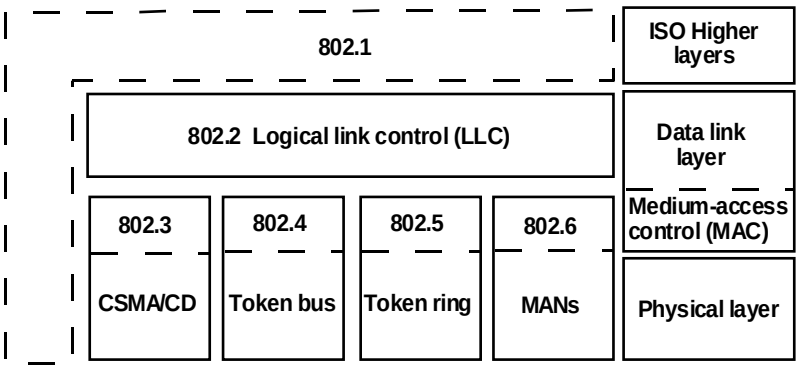

Figure 2-2. IEEE Family of Protocols [Kei89]

2.1.3.1 LAN Topologies

The way in which stations are interconnected is called the network topology and often controls the capabilities and limitations of the overall network [ClP78]. Although several topologies exist, they all stem from three fundamental types—the ring, the bus, and the star.

2.1.3.1.1 Ring Topology

The ring topology is characterized by a circle, or ring, of nodes all sharing a medium. As shown in Figure 2-3, all nodes attach to the ring at a repeater. The repeaters serve a simple, yet critical, function; they accept bits on one link and transmit them on the other without altering the data [Sta91]. Since this unidirectional ring has no beginning or ending, the source node serves as both; it generates the packet and absorbs it after it has traversed the entire ring. All nodes cannot transmit a packet onto the medium simultaneously. To do so would cause a collision corrupting all packets. A medium access control mechanism is required. A ring network will fail if any node goes down or a cable is cut or disconnected. In addition, connecting all nodes in a ring network involves complicated wire routing [Kei89].

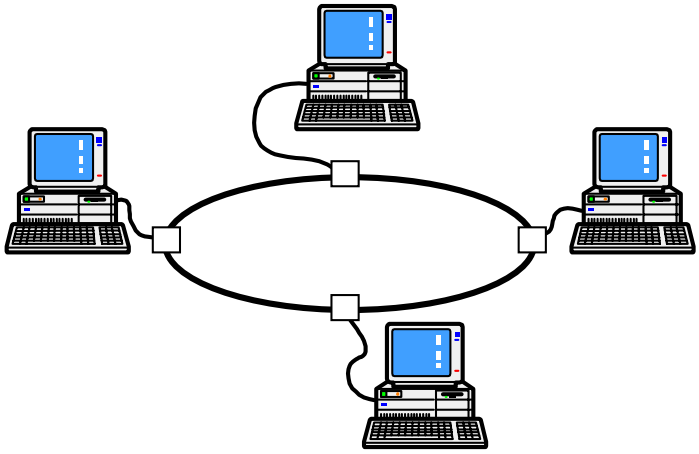

Figure 2-3. Token Ring Topology

2.1.3.1.2 Bus Topology

As shown in Figure 2-4, the bus topology allows all nodes to connect to a single medium. Since the medium is shared, all transmissions are heard by all nodes virtually instantaneously, but only one node can successfully transmit at one time. Again, all nodes must abide by a medium access control scheme. Bus topologies do not suffer from a performance degradation when a node fails and use coaxial cable that is widely available. Some drawbacks include limitations on the length of the bus and signal balancing

between nodes [Kei89]. Since the advantages far outweigh the disadvantages, the bus topology is the most common topology used in the United States [Kei89, SaI95].

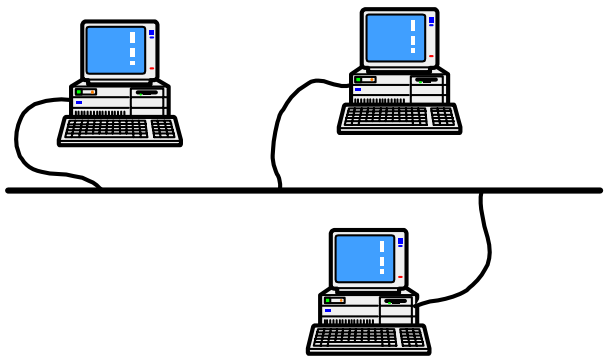

Figure 2-4. Bus Topology

2.1.3.1.3 Star Topology

A central controlling switch is the heart of the star topology. As shown in Figure 2-5, all transactions must traverse this switch. A transmission from a node enters the switch and is broadcast on the remaining links [Sta91]. This behavior is logically equivalent to a bus topology in that all nodes hear the transmissions of all others. One difference is that a star facilitates the use of fiber optic cable thereby allowing higher data rates associated with fiber. A significant drawback to the star topology is the entire network will fail if the central switch fails.

2.1.3.2 Medium Access Control

Regardless of the topology, a LAN must utilize a medium access control (MAC) mechanism to handle when and how a station gains access of the medium and to decide what to do in case of colliding frames. The purpose of the MAC is to allocate the medium in an equitable manner to all stations. The more notable MAC techniques are categorized as asynchronous time-division multiplexing. Further breakdown yields two mechanisms—contention-based (random-access) and deterministic-based (controlled) [Kei89, Sta91].

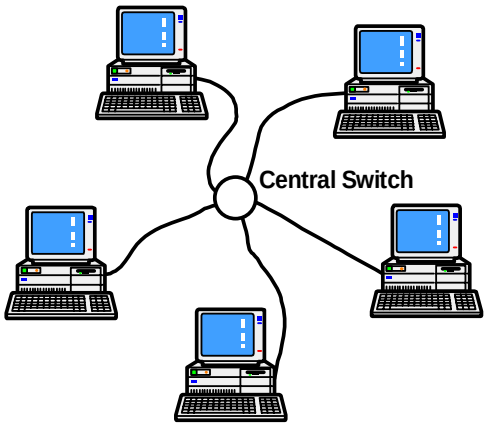

Figure 2-5. Star Topology

As the name implies, contention techniques are based on the premise that all stations must contend for access to the channel. Also, there is no a priori knowledge of which station will gain control. Deterministic techniques, on the other hand, offer complete predictability of which station will control the channel in the near future.

This section discusses some contention-based mechanisms and how their implementations affect network performance. Specifically, ALOHA, carrier-sense multiple-access (CSMA), and CSMA with collision detection (CSMA/CD) are presented.

2.1.3.2.1 ALOHA Protocols

Developed by Abramson [Abr73] in the early 1970s to allow geographically distant computer terminals access to the central computer system at the University of Hawaii, ALOHA is generally regarded as the first radio multi-access technique [Sta91]. The vast distances and obstacles (200 mile radius between the Hawaiian islands) between the operating units and the central computer necessitated the use of radio to transmit data [Abr73].

2.1.3.2.1.1 Pure ALOHA

A truly free-for-all protocol, ALOHA allows a station to send immediately when it has a data packet to send. Then, the station waits a prescribed time (typically twice the maximum round-trip propagation delay of the network). If an acknowledgment is received within this time, the packet is assumed to be successful. If, however, an acknowledgment is not received, the packet is retransmitted, and the process repeats until the packet is either successful or the station gives up because of too many transmissions [Abr73, BeG92, Kei89, Sta91]. When receiving a packet, the station checks its validity. If

the packet is in error, the station simply rejects it and does not send an acknowledgment back to the sender. A corrupted packet may result from bit errors due to noise in the channel or two stations transmitting simultaneously.

An analytical model for system throughput was developed using the following assumptions [Abr73, Sta91]:

1. There are an infinite number of stations generating an infinitely small number of packets thereby creating a finite arrival rate to the network.
2. Packets are of fixed length.
3. The channel is ideal and faithfully transports all bits without error.
4. A station can only accept one packet at any one time from the input source.
5. The offered load is Poisson distributed.

System throughput for this model is given by [Abr73, BeG92, Kei89, KoY77]

$$S = Ge^{-2G}, \tag{2-1}$$

where S is the normalized throughput of the network and G is the normalized offered load. The maximum throughput of S = 0.184 occurs at an offered load of G = 0.5 implying an ALOHA network can at best use only 18 percent of the channel capacity.

2.1.3.2.1.2 Slotted ALOHA

In an effort to improve on ALOHA's dismal channel utilization, Roberts [Rob75] devised a variation of ALOHA called slotted ALOHA. Slotted ALOHA is ostensibly the same as pure ALOHA with one difference—packets are only transmitted at the beginning of a slot. A slot is a fixed period of time defined to be the time required to transmit a packet. Synchronization among stations is achieved by a central mechanism. The end result is a throughput of [Abr73, KoY77, Rob75]

$$S = Ge^{-G}. \tag{2-2}$$

Throughput is maximized at S = 0.368 when G = 1. By requiring all stations to delay transmission until a slot begins, throughput is doubled over pure ALOHA [Rob75, Tan88].

Metzner [Met76] devised a technique to improve the utilization of slotted ALOHA by exploiting the capturing effect. (Capturing allowed a sufficiently stronger signal to be correctly received by the destination despite the presence of interfering signals.) He found that if users were divided into two groups such that one transmitted using high power and the other using low power, the maximum channel utilization increased to approximately 53 percent.

2.1.3.2.2 Carrier Sense Multiple Access

Although slotted ALOHA improves on pure ALOHA, a 37 percent channel utilization is still far from ideal. Kleinrock and Tobagi [KlT75] introduced a family of protocols called carrier sense multiple access (CSMA) to take advantage of the physical attributes of LANs. Since stations on a LAN share a medium and can hear each other's transmissions with small propagation delays relative to packet transmission time, CSMA can exploit these phenomena by requiring a station to listen to the medium before transmitting a packet. The fundamental difference between CSMA and ALOHA is that ALOHA transmits a packet regardless of the state of the channel. With CSMA, if a station senses the channel is in use, it simply does not transmit; doing so would cause a collision. Instead, the station waits some period of time before attempting transmission. If the channel is idle, the station may transmit. Assuming a station successfully gains control of the channel and sends a packet, it must wait for an acknowledgment using the same procedure used in ALOHA.

A collision can still occur however. If two or more stations wait (back off) the exact same time before transmitting, a collision will occur. Also, if two or more stations transmit within a time window equal to the propagation delay between stations, a collision will occur. If a packet traverses the entire network (heard by all stations) before all other stations attempt to transmit, the transmitting station has gained control of the channel until the entire packet is transmitted [KlT75].

Kleinrock and Tobagi made assumptions in their analysis which made the problem tractable. A station could either transmit or receive, but not do both at the same time. The carrier detect time and the time to switch from transmit to receive were negligible. All packets had fixed length. Packet errors only resulted from colliding packets; random errors induced by noise were negligible. Overlapping packets were totally destroyed; the system assumed noncapture. Packets arrived from an infinite number of sources and followed a Poisson distribution. Finally, the propagation delay was identical for all source-destination pairs.

Two fundamental schemes are used to determine when to transmit and how to handle collisions. These are referred to as nonpersistent and 1-persistent CSMA. The distinguishing characteristics of each are presented below.

2.1.3.2.2.1 Nonpersistent CSMA

In nonpersistent CSMA, if a station senses the medium to be idle, it transmits its packet. If the medium is busy, the station waits a random amount of time and then listens to the medium again. This process continues until the packet is successfully transmitted or the station gives up [Kei89, Kle76, Sta91].

Random wait times help avoid collisions from multiple transmitting stations. This is a nonslotted scheme since slots are not implemented.

Channel throughput for this nonslotted variation is [Kei89, Kle76, KlT75, Sta91]

$$S = \frac{Ge^{-aG}}{G(1+2a)+e^{-aG}}, \tag{2-3}$$

where S is the normalized throughput of the network, G is the normalized offered load, and *a* represents the fraction of a packet in which a collision can occur. Specifically,

$$a = \frac{\text{propagation delay}}{\text{packet transmission time}}. \tag{2-4}$$

A slotted version of nonpersistent CSMA also exists. This variation operates similar to slotted ALOHA in that all stations are synchronized and stations are allowed to send packets only during the beginning of a slot. The throughput for a slotted nonpersistent is [Kei89, Kle76, KlT75, Sta91]

$$S = \frac{aGe^{-aG}}{1-e^{-aG}+a}. \tag{2-5}$$

Sinha and Gupta [SiG85] provide further insight into how the nonslotted, nonpersistent CSMA behaves in the presence of packet errors. They analyze channel throughput under the assumption that a digitally modulated signal is corrupted by Rayleigh fading and additive white Gaussian noise. Their findings are

$$S = \frac{Ge^{-aG-\frac{22.5621}{\bar{\rho}}}}{G(1+2a)+e^{-aG}}, \tag{2-6}$$

where $\bar{\rho}$ is the average signal-to-noise ratio (SNR). The analysis finds that at an average SNR of 50 dB the channel throughput achieves the same performance as a nonfading channel.

2.1.3.2.2.2 1-persistent CSMA

The 1-persistent scheme was developed to improve throughput over nonpersistent CSMA for systems experiencing lighter loads [KlT75]. Since nonpersistent CSMA uses random wait times, time may be wasted while waiting to transmit. The 1-persistent scheme follows the same rules as nonpersistent CSMA if the medium is idle. However, if the medium is busy, the station will continue to listen to the channel and transmit immediately with probability 1 (hence the name 1-persistent) when the channel is free. If a collision occurs, the transmitter waits a random amount of time and starts listening to the medium again after the wait time expires.

The 1-persistent scheme is a special case of the *p*-persistent CSMA protocol in that the *p*-persistent protocol does not necessarily transmit immediately when the channel becomes free. Instead, the *p*-persistent protocol transmits with probability *p*. The p-persistent scheme has not been readily adopted, and in general, 1-persistent is preferred over *p*-persistent.

Channel throughput for unslotted 1-persistent CSMA is [Kle76, KlT75, SoM87, TaK85]

$$S = \frac{G\left(1+G+aG\left(1+G+\frac{aG}{2}\right)\right)e^{-G(1+2a)}}{G(1+2a)-\left(1-e^{-aG}\right)+(1+aG)e^{-G(1+a)}}. \qquad (2\text{-}7)$$

Channel throughput for slotted 1-persistent CSMA is [Kle76, KlT75, TaK85]

$$S = \frac{Ge^{-G(1+a)}\left(1+a-e^{-aG}\right)}{(1+a)\left(1-e^{-aG}\right)+ae^{-G(1+a)}}. \qquad (2\text{-}8)$$

Figure 2-6 illustrates the behavior of slotted and pure ALOHA as well as slotted variations of 1-persistent CSMA and nonpersistent CSMA. The value for *a* is set to 0.01 for both CSMA schemes. All techniques eventually become unstable under heavy loads (larger values of G). Even stations using random backoff times ultimately become overwhelmed. To maximize system performance, the network should operate in the region of the plot that maximizes throughput.

2.1.3.2.2.3 Carrier Sense Multiple Access with Collision Detection (CSMA/CD)

Metcalfe and Boggs [MeB76] devised a scheme to improve CSMA channel utilization by enhancing it with a collision detection mechanism. CSMA/CD is essentially the same as CSMA but behaves differently when a collision occurs. If the medium allows a station to listen to the channel as it transmits, CSMA/CD can improve channel utilization by providing an early collision indication to all stations. Since a station can hear itself transmitting, it is able to determine immediately when a collision has occurred by detecting a garbled signal on the medium instead of its signal alone. When this occurs, each colliding station continues to transmit bits for a predetermined amount of time to ensure all stations hear the collision. In other words, each station purposely jams the channel with gibberish. This is where CSMA/CD departs from the CSMA scheme. CSMA allows colliding stations to continue transmitting until the entire packet is sent; CSMA/CD does not. After jamming the channel, each transmitting station waits a random time before attempting to transmit again. This random backoff helps avoid subsequent collisions.

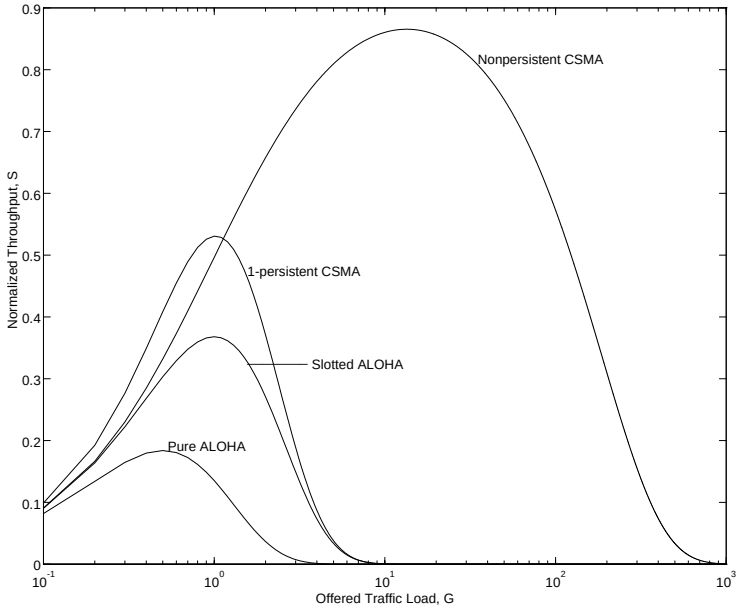

Figure 2-6. Performance of MAC Schemes [Kei89]

CSMA/CD comes in the same varieties as CSMA—slotted and unslotted versions of nonpersistent, 1-persistent, and *p*-persistent. Although each protocol has been analyzed by various researchers [ChK91, ChR85, Gon85, MeB76, MeL83, SoM87, TaK85, TaK87, ToH80], only one is addressed here due to its popularity—unslotted 1-persistent CSMA/CD [CoL83, ShH82, Sta91, Tan88, Tas86].

Channel throughput for unslotted 1-persistent CSMA/CD is [SoM87]

$$S = \frac{\left(P_{20} + P_{21}\right)e^{-aG}}{\frac{\left(1-P_{11}\right)P_{20} + P_{10}P_{21}}{G} + \left(\left(1-e^{-aG}\right)\left(2a+b+\frac{1}{G}\right) + e^{-aG}\right)\left[P_{20}+P_{21}\right] + (2a+b)\left[1-P_{10}-P_{11}\right]}, \quad (2\text{-}9)$$

where

$$P_{10} = e^{-G(a+1)} + \frac{1}{2}e^{-G(a+b)}\left[1-e^{-2Ga}\right],$$

$$P_{11} = Ge^{-G(1+a)} + \frac{1}{4}e^{-G(a+b)}\left[1 - e^{-2Ga}\right]\left(1 + 2G(a+b)\right),$$

$$P_{20} = e^{-G(a+b)},$$

$$P_{21} = G(a+b)e^{-G(a+b)},$$

and b is the time required to transmit the jamming signal. The protocol's behavior is shown in Figure 2-7. It is interesting to note how the throughput now has a broad maximum.

Due to its popularity, this scheme was developed into a standard called Ethernet [MeB76]. To improve stability under heavy loads, Ethernet added a random backoff time drawn from a truncated binary exponential backoff mechanism (each collision doubles the potential time a station must wait before attempting retransmission). Ethernet was so successful that the IEEE used it as the basis for the IEEE 802.3 standard [IEE85].

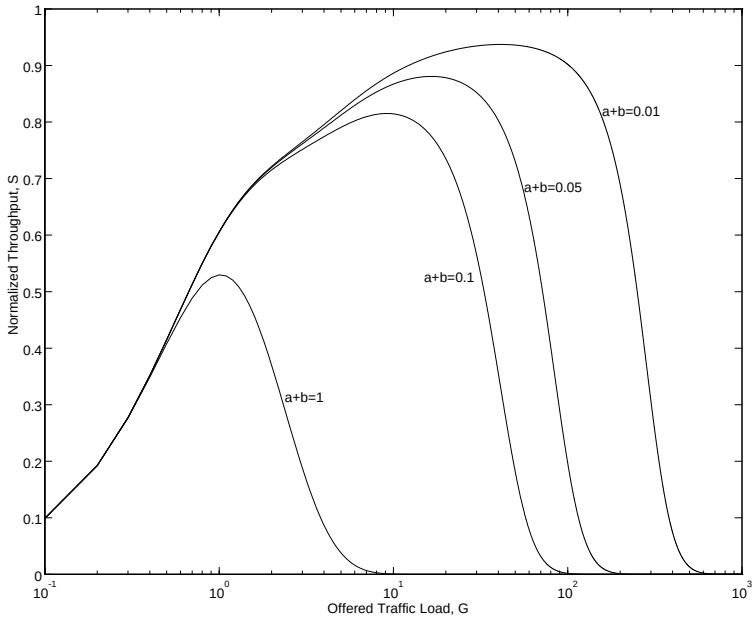

Figure 2-7. Throughput of CSMA/CD as a Function of G [Kei89]

2.2 Spread Spectrum Techniques

Spread spectrum is coming into vogue as a popular transmission form for commercial applications. Spread spectrum techniques have been used extensively by the military for over 50 years with very good results [Sch82]. In addition, recent events have enabled its commercialization to the point that handheld Global Positioning System (GPS) receivers and even residential cordless telephones are implementing this technology to improve performance [Dix94]. The U.S. Federal Communications Commission (FCC) opened three license-free frequency bands in 1985 to spread spectrum users [BaB94, Dix94]. Also, recent cutbacks in military spending has coaxed several spread spectrum companies as well as small startup companies to diversify into consumer products [PaP95]. Advances in chip manufacturing such as digital signal processing and application-specific integrated circuits have spurred this emerging technology by permitting implementations of complex spread spectrum functions in cost-effective packaging [BaB94, Fre91]. Consequently, markets for wireless communication products and services are growing to satisfy the demand for autonomous and untethered communications.

In its most basic form, spread spectrum can be defined as "a technique in which an auxiliary modulation waveform, independent of the information data, is employed to spread the signal energy over a bandwidth much greater than the signal information bandwidth" [MaN94]. The receiver despreads the signal using a synchronized replica of the original spreading (auxiliary) signal to recover the information signal [MaN94].

Spread spectrum techniques offer many unique capabilities. Spreading the signal allows a spread spectrum system to operate with little interference to conventional radio systems. The signal is easier to hide from unauthorized or unfriendly systems. The possibility of jamming is reduced, and signal detection is more difficult leading to increased security. Spread spectrum systems offer selective addressing and code division multiplexing [Dix94]. They can reduce the effects of multipath thereby increasing node mobility within an enclosed area such as an office [Mit91, Pah85, TsC95, WeC90]. (Multipath can be described as multiple copies of the same signal arriving at the receiver via different paths at slightly different times, potentially causing mutual interference [Mit91, TsC95].) Although spread spectrum has many qualities, not all capabilities can be provided by one system configuration at one time.

Spread spectrum is not without disadvantages. It requires increased bandwidth to accommodate the spread signal. Although spread spectrum offers a performance improvement in the presence of other signals, it does not negate the corrupting effects of Gaussian noise. Spread spectrum systems are typically more expensive than traditional radio systems. Finally, FCC regulations dictate operating parameters that may limit system performance.

FCC Part 15.247 allocated the unlicensed use of spread spectrum systems in the Industrial, Scientific, and Medical (ISM) frequency bands (Table 2-2) [BaB94, Dix94].

Table 2-2. Industrial, Scientific, and Medical (ISM) Frequency Bands

ISM Frequency Band (MHz)	Available Bandwidth (MHz)
902 - 928	26
2400 - 2483	83.5
5725 - 5870	125

As the term ISM implies, other systems use these same bands; therefore, spread spectrum systems operating in these bands must contend with interference from these "other" systems. In addition, the spread spectrum systems are considered the secondary users and must not interfere with primary users. To this end, the FCC regulations state that secondary users, which includes wireless LANs, must abide by certain limitations [Dix94]. (Although the FCC regulations address both frequency hopping and direct sequence systems, only a few regulations germane to direct sequence are discussed here.) The transmitter's maximum peak output power into the antenna (6-dBi maximum antenna gain) is limited to less than one watt and the processing gain must be at least 10 dB. Processing gain is explained later in this chapter.

A better appreciation of the potential interference is gained by realizing that primary users in the ISM bands are authorized to transmit on the order of 100 to 10000 times more power than a secondary user. Therein lies the possibility of significant intervention. Primary sources are scarcely used, so these drawbacks do not preclude the successful use of secondary sources.

Spread spectrum capabilities are based on a direct implementation of Shannon's channel capacity formula which is [Dix94]

$$C = W \log_2\left(1 + \frac{S}{N}\right), \tag{2-10}$$

where C is the channel capacity in bits per second, W is the bandwidth in hertz, N is the noise power, and S is the signal power. Switching bases and assuming S/N is small (this removes the log function), the equation becomes [Dix94]

$$\frac{N}{S} = 1.44\frac{W}{C} \approx \frac{W}{C}, \tag{2-11}$$

which simplifies to

$$W = \frac{NC}{S}. \tag{2-12}$$

Equation 2-12 offers an intuitive understanding of the principles of spread spectrum and the tradeoffs required. Bandwidth requirements increase if channel capacity increases while channel noise and signal power remain unchanged. Channel bandwidth must increase if the capacity remains the same but the channel becomes noisier (the noise-to-signal power ratio increases). The tradeoff is bandwidth for noise and capacity.

There are two spreading techniques used by radio spread spectrum systems known as direct sequence and frequency hopping. Direct sequence is the only spread spectrum technique used in this research; therefore, frequency hopping is not discussed.

2.2.1 Direct Sequence Spread Spectrum

Direct sequence spread spectrum (DSSS) is accomplished by modulating the information signal with a digital code sequence, called the pseudo-noise (PN) code, that resembles (the sequence is not truly random since it has a period) a random series of "ones" and "zeroes" called chips [Dix94, Cof95]. As shown in Figure 2-8, the heart of DSSS is the modulo-2 addition of the binary information signal with the PN code. This PN modulated signal is then sent to a modulator, typically a biphase phase-shift keying (BPSK) variation. It is this simplicity that makes DSSS the best known and most used technique [Dix94]. The modulated signal, by virtue of its "increased data rate," is characterized by having a much wider bandwidth than the original information signal.

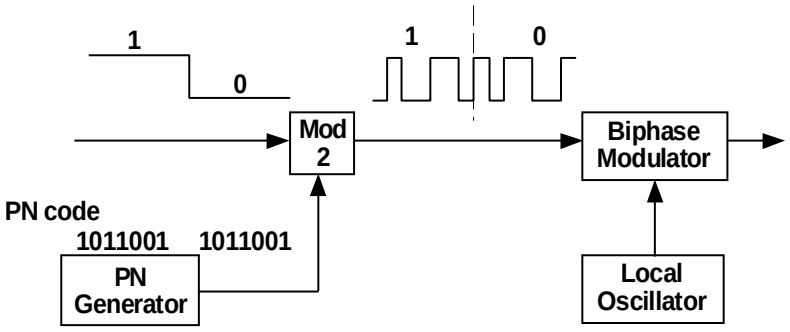

Figure 2-8. Direct Sequence Spread Spectrum Transmitter

2.2.1.1 Maximal Linear Code

As illustrated in Figure 2-8, a PN code is used to spread the information signal. Selecting the proper code is not a trivial task. Code length, type, and rate must all be considered when designing a DSSS system since these parameters dictate system performance. In fact, an incorrect code selection may invalidate advertised spread spectrum capabilities and actually decrease system performance [MaN94].

Once chosen, a code can be easily implemented using a sequence generator similar to the modular multiple-tap sequence generator shown in Figure 2-9. (Other generator configurations are available and offer the same performance; however, the modular generator is addressed here due to its versatility.) A series of delay elements act as a shift register where feedback from various stages is provided via logic gates (exclusive OR gates in this case). Actual feedback implementation is a function of the required code and has been conveniently generated for shift registers with up to 100 stages [Dix94].

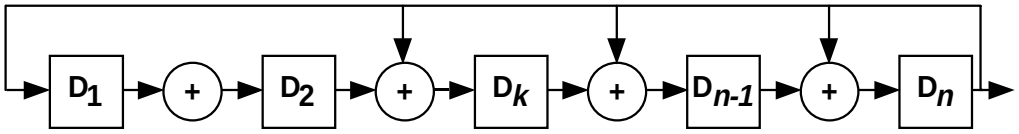

Figure 2-9. Modular Multiple-Tap Sequence Generator [Dix94]

To fully exploit DSSS systems capabilities, a maximal linear code sequence should be used. This class of codes has been shown to be the most efficient for general communication needs and must satisfy a host of requirements which are described here [Dix94]. A maximal linear code sequence must have a length of $2^n - 1$ chips where *n* represents the number of shift register stages (number of delay elements). The number of ones can only exceed the number of zeros by at most one, and the distribution of the ones and zeros must follow rigid guidelines [Dix94]. Autocorrelation of phase-shifted versions of a code with itself results in a value of -1 with one exception—autocorrelation of a code with a one chip shift in either direction results in a linear relationship between -1 and $2^n - 1$. Figure 2-10 illustrates this concept which is critical for recovering the signal in the presence of jamming. Modulo-2 addition of a code with a phase-

shifted version of itself yields yet another version with a unique phase shift. The sequence generator must cycle through all possible states, with one exception which is explained later, while creating a maximal linear code. If all of these prerequisites are met, the resultant sequence is a maximal linear code.

Once a maximal linear code is chosen and implemented via appropriate feedback logic, the sequence generator is loaded with any initial vector except the all-zeros vector. (An all-zeros vector will lock the generator in that state; therefore, the generator must never transition to this state.) The generator then produces the code by outputting one chip per generator clock cycle. As shown in Figure 2-8, the PN code is exclusive-ORed (Modulo 2 addition) with the information-bearing binary signal, modulated and sent to the channel. Upon detecting this signal, a spread spectrum receiver attempts to demodulate it. If properly coded and implemented, the resulting signal appears like noise to all receivers without an exact PN code match [CoM83].

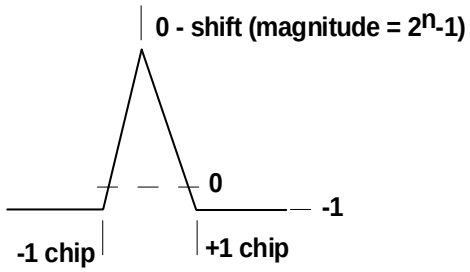

Figure 2-10. Autocorrelation Function [Dix94]

2.2.1.2 Demodulation

The underlying power of spread spectrum technology lies squarely in the demodulation process. It is here that process gain is realized.

A spread spectrum receiver attempts to despread the incoming signal by multiplying it with an exact synchronized replica of the PN code used during modulation. This technique is implemented via one of two types of despreaders—a serial correlator or a matched filter [MaN94]. Both perform the same basic operation of calculating the cross-correlation of the incoming signal with the reference PN sequence. The fundamental difference between the two lies in how the reference sequence is generated and compared to the incoming signal. As the name implies, the serial correlator performs correlation in a serial fashion;

the matched filter operates in parallel. The serial correlator must generate the PN code serially for each correlation check. Matched filters use a parallel register to store the sequence implying faster acquisitions [CoM83, MaN94]. Since internal registers are used in matched filters, the sequence length is typically limited to 64 chips; serial correlators do not suffer from code length limitations [MaN94]. Selecting one implementation over the other is a design decision based on system requirements. Regardless of the implementation, both yield the same results assuming the receiver is properly synchronized with the received signal [CoM83].

Figure 2-11 [Dix94] is a model of a direct sequence correlation receiver along with representative signal spectra at various stages of despreading. The input to the correlator is a mixture of the desired direct sequence signal and an interfering signal. The correlator despreads the desired signal such that its narrow-band representation is output to the bandpass filter. Incidentally, any interference is also subjected to this multiplication. Since the interference is not correlated to the PN code, the result is to spread the interference power over a wide bandwidth. Passing these two signals through the filter allows the desired signal to pass unabated while the diffused interference signal is vastly attenuated. Thus, the signal-to-noise ratio is [Dix94]

$$\frac{SG_p + I}{I}, \tag{2-13}$$

where S is the signal power, I is the interference power, and G_p is the process gain.

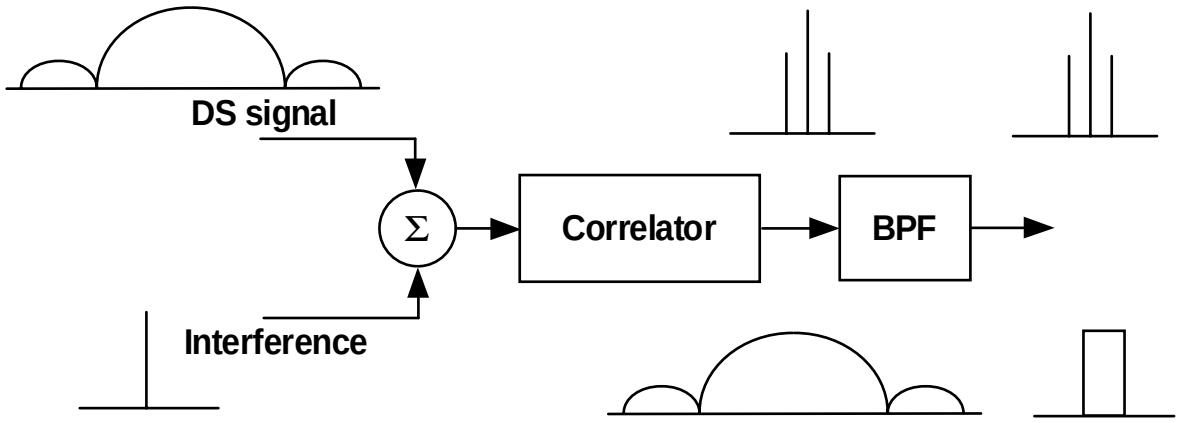

Figure 2-11. Direct Sequence Receiver Model [Dix94]

2.2.1.3 Process Gain

Process gain is a measure of merit for spread spectrum systems and is defined as

$$G_p = \frac{BW_{RF}}{R_{\inf o}}, \tag{2-14}$$

where BW_{RF} is the RF bandwidth and R_{info} is the data rate of the information. For DSSS systems, the process gain is usually taken as [Dix94, CoM83, MaN94]

$$G_p = \frac{\text{Chipping Rate}}{\text{Data Rate}}. \tag{2-15}$$

Assuming the system is symbol synchronous (one PN code sequence per data bit), the process gain is simply the number of chips used per data bit [CoM83, MaN94]. The process gain is in the form of a signal-to-noise improvement which comes at the expense of channel capacity. Therein lies a critical feature of spread spectrum. If capacity can be sacrificed, an improvement can be achieved in the signal-to-noise ratio. This characteristic is exploited in this research effort.

2.3 Wireless LANs

After the FCC opened the ISM bands to spread spectrum users in 1985 [BaB94, Dix94], wireless networking gained popularity and several wireless computer communication products emerged. These products can be divided into three broad categories: wireless LANs, Personal Communication Services (PCS), and home wireless networks. Wireless LANs are typically used for computer-to-computer communications within a relatively small area (e.g., within a building) and support data rates of around 1 Mbps. PCS is targeted towards voice and data applications that support data rates less than 56 Kbps. Finally, home wireless networks are an attempt to service homes with integrated voice and data service via cable into the home, and distribute the information throughout the home via wireless techniques with data rates less than 1 Mbps [DaE93]. This section addresses wireless LANs.

Wireless LANs provide a unique capability to users in the form of dynamic network topologies. More specifically, two fundamental configurations are provided—*ad hoc* and infrastructure. First, wireless LANs allow highly-mobile portable (typically laptop) computers to connect with each other in an *ad hoc* peer-to-peer configuration (Figure 2-12) with virtually no setup or disconnect overhead. That is, *ad hoc* networks do not depend on a backbone structure; each computer implements a distributed MAC. Although *ad hoc* computers are in close proximity to each other, a fully connected network cannot always be guaranteed [Che94]. Depending on the transmission band, signal strength may vary by 30 dB within one meter [AnR95, Gol94, MaN94, Mit91, Pah85]. Second, wireless LANs may be used to extend an

existing network by allowing wireless computers to access a wired network by using an access point or similar device in an infrastructure configuration (Figure 2-13). Regardless of the intended application, a wireless LAN offers the ability to move networked computers at will.

The first truly wireless LAN products arrived in 1990 and were labeled the "first" because of their high data rates and compliance with existing networking software [PaP95]. The wireless LAN market has continued to grow at an explosive rate to the point that numerous companies are marketing wireless peripherals for personal computers. AT&T's WaveLAN technology is the most prolific with Digital, Solectek, and Persoft all licensing the technology for their own products [Bal95]. Other products include Motorola's Altair system, Proxim's RangeLAN2, Aironet's ArLAN, Xircom's Netwave, and Windata's Freeport [Boy95, DaE93, NeT95]. These are but a few of the more well-known systems on the market; others exist.

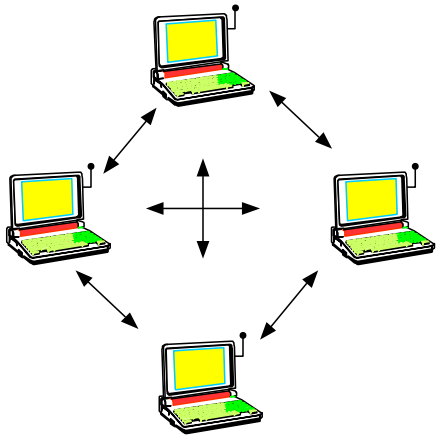

Figure 2-12. *Ad hoc* Network Configuration

2.3.1 Standardization Efforts

This proliferation of wireless LAN products and services has prompted standardization efforts for wireless technologies. Parallel efforts are currently underway in Europe and the United States. The European Telecommunications Standards Institute (ETSI) Committee RES10 was founded in 1991 to develop a standard called HIgh-PErformance Radio LAN (HIPERLAN). HIPERLAN is designed to provide high speed (up to 20 Mbps), short distance (50 meters) radio links between computers using dedicated frequency bands at 5.2 and 17.1 GHz [Fla94, Kru92, PaP95].

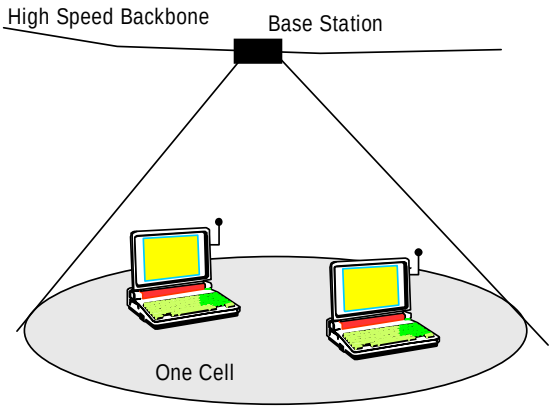

Figure 2-13. Infrastructure Configuration

Another LAN standards committee, called the IEEE Standards Project 802, is also formulating a standard for wireless LANs. The wireless local area networks standards working group, designated IEEE P802.11 (often referred to as simply 802.11), was established in July 1990 with the purpose of specifying a medium access control protocol and physical layer implementations [Fla94, Hay91, LiD94]. Interestingly enough, this working group was establish after the Executive Committee of IEEE Project 802 decided that a token bus protocol would lead to ineffective use of the radio frequency spectrum [Hay91, Ryp92]. The basis for this decision was that paths between stations are unreliable [Ryp92].

HIPERLAN and IEEE P802.11 are both setting the standards for wireless LANs. Although both are reaching for the same fundamental goals, variations may exist between the two due to spectrum allocation differences [BaB94]. To mediate differences, liaison has been established between these two governing bodies [Hay91].

2.3.2 IEEE 802.11

The IEEE Project 802.11 Committee was tasked "to develop a medium access control (MAC) and Physical Layer (PHY) specification for wireless connectivity for fixed, portable and moving stations within a local area" [IEE96]. The end product of this committee is the IEEE 802.11 standard, which is currently in draft form. The IEEE 802.11 standard (hereafter referred to as simply 802.11) addresses connectivity to fixed and moving devices including those mounted on moving vehicles. It also describes requirements for *ad hoc* and infrastructure networks and how mobile stations move within and across these networks. Asynchronous and time-bounded MAC procedures to support delivery of MAC service data units (MSDU) are described as well as security, privacy, and authentication issues [IEE96].

2.3.2.1 Distributed Foundation Wireless Medium Access Control

The Distributed Foundation Wireless Medium Access Control (DFWMAC) was chosen as the MAC for 802.11 devices. Designed for versatility, the DFWMAC services both *ad hoc* and infrastructure networks via a layered approach. This hybrid nature of this MAC is illustrated in Figure 2-14. The Distributed Coordination Function (DCF) is the lowest protocol level in the DFWMAC upon which all other MAC functions rely. DCF provides contention-based, asynchronous access by implementing a mechanism called Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) with a positive MAC-level acknowledgment. All stations, whether in an *ad hoc* or infrastructure network configuration, must implement the DCF. An optional access method, called the Point Coordination Function (PCF), may be used within an infrastructure network only. The PCF relies on the DCF while providing contention-free services. This access technique uses a point coordinator to determine which station has the right to gain access to the medium. The point coordinator serves as the polling master while querying all stations within the infrastructure. Both DCF and PCF must coexist such that both access methods can gain control of the medium.

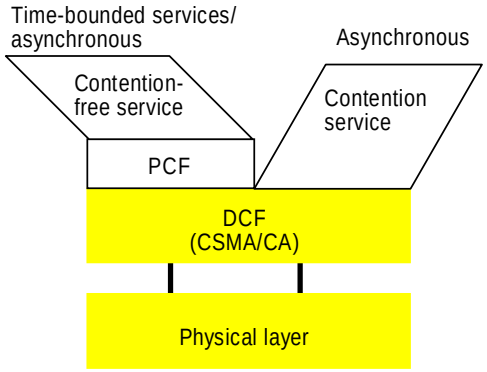

Figure 2-14. Distributed Foundation Wireless Medium Access Control (DFWMAC) [IEE96]

2.3.2.1.1 Distributed Coordination Function

As the foundation of all MAC functions, the distributed coordination function (DCF) using CSMA/CA must provide a flexible means of controlling the medium. This access mechanism is shown in Figure 2-15.

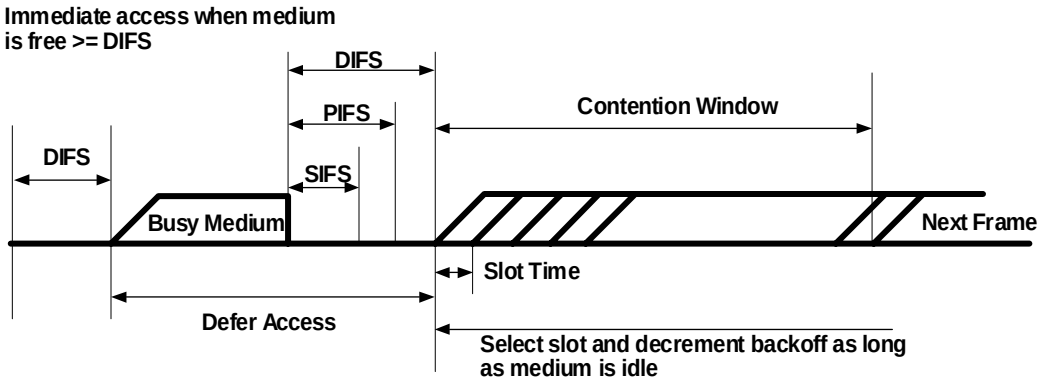

Figure 2-15. Basic Access Method [IEE96]

2.3.2.1.1.1 Carrier Sense Mechanisms

CSMA/CA relies on the ability of the station to detect the presence of a carrier. Two carrier sense mechanisms are used. One is the mechanism provided by the physical layer upon detecting the presence of a certain amount of signal energy in the frequency band. The other is a virtual mechanism provided by the MAC itself called the Net Allocation Vector (NAV). The NAV is explained later in this chapter.

2.3.2.1.1.2 Inter-frame Space

Inter-frame space (IFS) is the time between frames. The spaces in Figure 2-15 provide a priority scheme. When a station senses the medium to be free for the prescribed IFS, it may gain control of the channel. Once a station starts transmitting (gained control) all other stations will hear this transmission and wait. Naturally, the station using the shorter IFS will win access; hence, a shorter IFS implies a higher priority. The highest priority IFS is the Short-IFS (SIFS), followed by the PCF-IFS (PIFS), and, finally, the lowest priority space called the DCF-IFS (DIFS).

The SIFS is used for all immediate responses such as acknowledgment frames, clear-to-send frames, polling responses, and fragmented data frames. PIFS is used to initiate a contention-free transmission, and DIFS is used to transmit asynchronous, contention-based data and management frames.

2.3.2.1.1.3 Access Mechanism

When a station wants to transmit a frame, it first senses the medium. If the medium has been free for at least DIFS, the station may transmit immediately. Assuming the frame is correctly received at the destination, the destination waits for SIFS to elapse and then sends a short acknowledgment frame back to the source indicating a successful transmission. The SIFS priority is used to ensure other stations do not attempt to transmit before this transaction is completed. If a correct (the acknowledgment frame may actually be in error) acknowledgment is not received within an allotted time, the source station assumes the frame was received in error and retransmits. This retransmission must abide by the backoff procedures discussed below.

If the medium is sensed busy, the station will defer until it detects a DIFS at which time the station generates a random backoff time. The backoff mechanism is implemented to alleviate possible collisions when they are most likely to occur—immediately after a busy period. If a backoff is required, the station selects the appropriate contention window (CW) based on the number of frame transmissions attempted. For example, the initial frame transmission has a CW of 31; the next transmission of the same frame has a CW of 63 and so on. This truncated binary exponential backoff mechanism is illustrated in Figure 2-16 and is used to improve network stability under heavy loads. Once the appropriate CW is selected, the backoff time is calculated by multiplying the CW with a uniformly-distributed random number between 0 and 1. This result is then multiplied by the medium's slot time yielding the number of slots to backoff.

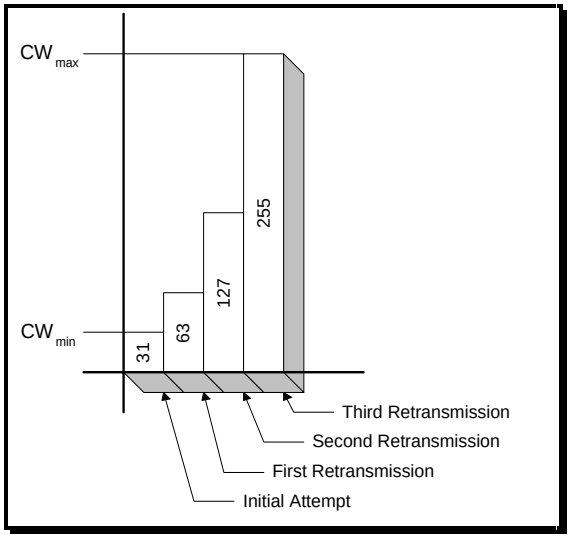

Figure 2-16. Exponential Increase of Contention Window [IEE96]

After the station selects the backoff time and DIFS has elapsed, the station's backoff timer is allowed to decrement while the medium remains free. Other stations may have also performed backoff in hopes of gaining access to the medium. The station selecting the smallest backoff time will win. Its timer will expire first, and it will transmit immediately. Upon hearing this transmission, all other deferred stations will halt their backoff timers until DIFS is sensed once again. At this point, all backoff timers are allowed to countdown again. This process continues until all stations are serviced. Figure 2-17 illustrates multiple stations in various stages of backoff. Station A gains access to the channel immediately. Stations B, C, and D all wish to transmit at the same time as A, so they all defer until DIFS is sensed. Each chooses a backoff time using a CW equal to 31 (initial transmission). Station C chooses the shortest backoff time (it chooses the smallest random number between 0 and 1) and is allowed to transmit when its backoff timer expires. Stations B and D freeze their timers while the medium is busy. Eventually all stations are allowed access to the channel. In fact, stations that loose the contention and defer are more likely to be serviced during the next contention than a new station entering backoff for the first time.

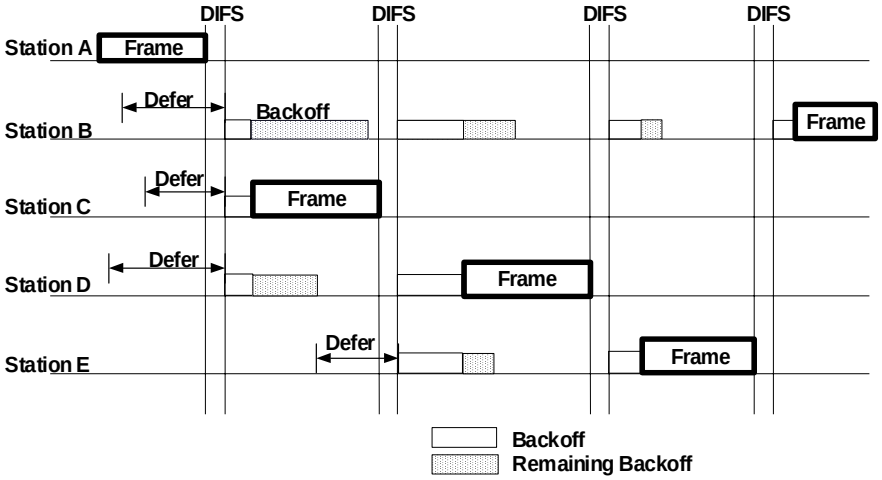

Figure 2-17. Backoff Procedure [IEE96]

2.3.2.1.1.4 Request-to-Send and Clear-to-Send

Within the DFWMAC, a mechanism exists to essentially reserve the medium for a specified period of time. This mechanism is implemented via a four-way handshake involving the exchange of special frames called request-to-send (RTS) and clear-to-send (CTS) as depicted in Figure 2-18. These

frames contain a duration field that defines how long the station wishes to control the channel. When the source transmits the RTS to the destination, all stations hearing the RTS copy the duration field to their NAV, which is simply a countdown timer. All stations avoid using the channel until the NAV has expired. Since the CTS frame also contains a duration field, the destination's CTS response back to the source causes all stations hearing this frame to update their NAV. This technique of distributing medium busy reservation information via RTS/CTS frames is called the virtual carrier sense mechanism. Any station requesting access to the channel must first check both the physical layer and virtual carrier sense mechanisms.

RTS/CTS brings fortunate capabilities to 802.11. This technique can be thought of as a collision detection mechanism since the data frame is transmitted only after a successful RTS/CTS exchange [IEE96]. A collision is most likely to occur during the transmission of the relatively small RTS frame resulting in earlier resolution. In addition, the hidden terminal problem is solved [BhD94, Gol94, IEE96, WeW95]. Figure 2-19 demonstrates the hidden terminal scenario. If station A sends to B, station C, being out of A's range, will not sense the channel busy. Instead, station C could attempt to transmit causing a collision at B. If RTS/CTS is used, station A transmits a RTS to B. All stations hearing this update their NAV and start the countdown. Upon receiving the RTS, station B responds with a CTS (assuming of course it is ready to receive). Stations hearing the CTS, including C, update their NAV and wait. This scenario ensures all stations that might cause a collision are aware of a pending transmission.

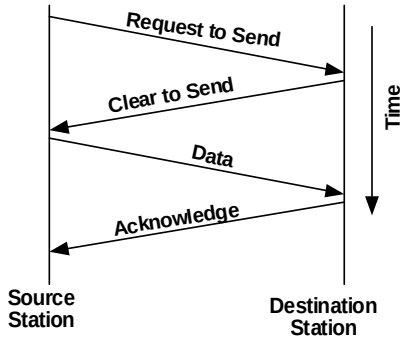

Figure 2-18. RTS/CTS Mechanism [Gol94]

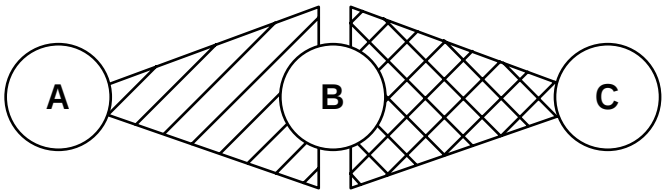

Figure 2-19. Hidden Terminal Situation

The indiscriminate use of RTS/CTS may degrade network performance, since it adds overhead in the form of transmitting two extra frames. Small (less than 100 bytes) data frames may forego the RTS/CTS mechanism to improve performance [Gol94]. For this reason, each station is allowed to manage RTS/CTS use. A station may decide to always use it, never, or only when frames are of a specified size or larger.

2.3.2.2 Physical Layer Specifications

The IEEE Project 802.11 Committee was tasked with developing a MAC capable of handling multiple physical transmission layers. Specifically, three physical layer implementations are proposed by 802.11—direct sequence spread spectrum (DSSS) in the 2.4 GHz ISM band, frequency hopping spread spectrum in the 2.4 GHz ISM band, and diffused baseband infrared in the 850 to 950 nanometer range [Che94, IEE96]. As shown in Figure 2-20, this is accomplished through the use of a physical layer convergence function and a physical medium dependent sublayer. The convergence function maps MAC service requests into appropriate medium-dependent frame formats for sending and receiving. The physical layer convergence procedure (PLCP) sublayer is responsible for this convergence function.

2.4 Related Research Efforts

Wireless local area networks are currently enjoying a concentration of research efforts focused in two areas which, not coincidentally, correspond to either the network/transport layers or the MAC/physical layers. The following sections discuss this research with emphasis on the MAC and physical layers. It is shown that most efforts study one layer at a time while dismissing inter-layer management and how to exploit this to improve system performance.

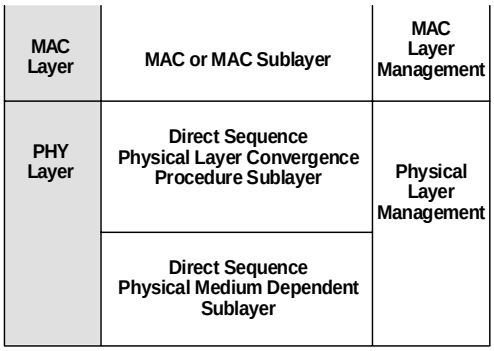

Figure 2-20. Direct Sequence Protocol Reference Model [IEE96]

2.4.1 IEEE 802.11

2.4.1.1 Modified Backoff Schemes

The methods spelled out in the proposed IEEE 802.11 standard [IEE96] are analyzed for a better understanding of network performance under various conditions. Woesner, *et al.* [WoW95] analyze the backoff mechanism and propose alternates. They explain that DFWMAC uses a uniformly distributed probability function when selecting a random backoff slot and how this distribution skews the overall probability of selecting an earlier slot (a winning slot). Since a station entering the contention for the first time selects from the same uniform distribution as remaining stations, the probability of earlier slots being in use is much higher than later slots. For example, when the network is in equilibrium under high load, slot one is more likely to be in use by a station than slot two. This increased probability leads to more collisions.

Woesner, *et al*. propose an alternative—force new stations arriving in the contention to select from later slots. Not only does this reduce the probability of collisions, it also ensures a level of fairness amongst stations since new stations are not allowed to barge ahead of waiting stations. Simulation results indicate an improvement of up to 20 percent in throughput and 15 percent in average access delay.

2.4.1.2 RTS/CTS Mechanism

Weinmiller, *et al*. [WeW95] recognize the dichotomy of the RTS/CTS mechanism and offer solutions to the question of when to use this optional feature, since prudent use of this mechanism will improve performance. Collision probabilities will decrease since all stations, including hidden terminals, will know of the impending transmission. Also, when collisions do occur, they involve the relatively

small control frames. However, performance can simultaneously be degraded since two additional frames (RTS and CTS) must be sent. Their research attempts to strike a balance.

Weinmiller, *et al*. find that, even though network performance increases significantly, using RTS/CTS does not completely solve the hidden terminal problem. Using a simulation of eight stations exchanging data, they discover multiple hidden stations cause a breakdown. Since mutually hidden stations cannot hear each other, they attempt to transmit simultaneously and ultimately become synchronized further exacerbating the situation.

Based on their simulation configuration and parameters, Weinmiller, *et al*. feel the threshold for when to use RTS/CTS is 200 bytes. In other words, use RTS if the packet is over 200 bytes. Otherwise, overhead negates improvements. For best performance they believe this threshold should be adaptively varied based on system load to increase the threshold under high load.

2.4.2 Wireless Medium Access Control

Beyond the proposed IEEE 802.11 standard, researchers are exploring other forms of medium access control (MAC) for wireless LANs. They investigate various access techniques and how performance is affected.

2.4.2.1 Demand Assigned Multiple Access (DAMA)

Cheah [Che92] proposes a DAMA scheme, which is based on slotted ALOHA, for the 802.11 MAC. He feels this scheme will satisfy 802.11 requirements while maintaining implementation simplicity by designating a head-end to act as the coordinating agent for a basic service area (BSA). He defines a BSA as the physical area serviced by one head-end.

The head-end mentioned by Cheah is similar to the point coordinator used in the point coordination function of 802.11. His implementation, however, addresses both infrastructure and *ad hoc* networks via the use of management slots within the data stream. He also suggests that any node could assume the role of a head-end.

2.4.2.2 MACAW

MACAW is a research effort by Bharghavan, *et al*. [BhD94] designed to improve on existing wireless medium access protocol based on collision avoidance. They feel that initial work in this area was commendable, but perhaps a better scheme existed. Based on the Multiple Access, Collision Avoidance (MACA) protocol by Karn [Kar90] and subsequently refined by Biba [Bib92], MACAW enhances MACA's performance.

Bharghavan, *et al*. feel that since congestion is location dependent, the relevant contention is located at the receiver instead of the sender thereby rendering the carrier sense mechanism ineffective. Further, they feel congestion information, including contention synchronization, should be broadcast explicitly via the medium access protocol to all stations.

MACA is based on the same RTS/CTS mechanism found in 802.11. After simulating MACA, Bharghavan, *et al*. find that the binary exponential backoff algorithm used in MACA allows a level of unfairness amongst the stations. They explain the unfairness is a result of the way the backoff time is determined. A station with a much smaller backoff time will win the contention as well as subsequent contention periods (assuming it has more packets to send) while the losing stations continually increase their backoff times. They contend the winning stations will maintain control of the channel indefinitely if a maximum backoff time is not used. Their remedy is to embed the winning backoff time in the transmitted packet. All stations hearing this packet copy this time into their own timers. Bharghavan, *et al*. find that broadcasting the backoff time ensures fairness.

Bharghavan, *et al*. also propose modifications to the RTS-CTS-DATA-ACK scheme used in 802.11. They show that under certain network conditions the insertion of a data-sending packet (RTS-CTS-DS-DATA-ACK) improves performance when a CTS is lost. This DS packet also provides contention synchronization to the rest of the network.

2.4.2.3 Analysis of Client-Server Traffic and Capture

LaMaire, *et al.* [LaK94] devise an efficient medium access control for single-cell wireless LANs. They consider a generic channel access protocol based on a time-slotted reservation scheme where one slot accommodates a single packet. A fixed-length frame is composed of data periods and a reservation periods. Data periods are used for the actual transmission of data packets, whereas the reservation periods inform the base station (server) of a data request. Access to the reservation slots is dictated using a slotted ALOHA mechanism with the probability of transmission in a slot defined as p.

Using detailed analytical models and simulations, LaMaire, *et al.* analyze the effects of p and capture on network performance. Specifically, they investigate performance for a fixed p and a variable (optimized) p. They find that if the number of contending stations can be estimated, the optimal-p case yields best results. If there are no transmission errors, a fixed value of $p = 1$ (1-persistent) gives the best performance. In addition, they consider the effects of capture on performance. They develop a model to describe the increased throughput of stations in the presence of capture. If capture is ignored, a worst-case

estimate of throughput is obtained. Their results show throughput can increase by as much as 34 percent when a single remote (far away from all others) station experiences capture.

2.5 Summary

This chapter highlighted various wireless local area network issues pertinent to this research effort. An all encompassing presentation of wireless LAN aspects was not warranted. Although minute details and irrelevant issues were not discussed, sufficient information was provided to facilitate an appreciation of the unique contributions made by this effort.

An overview of computer networks was provided with emphasis on local area networks. The division of networks into the OSI seven layer model to facilitate modular design and manageability was presented; each layer and their interrelationships were explained. Next, the ring, bus, and star LAN topologies were briefly presented along with their pros and cons. The IEEE 802 family of standards and how LANs require a medium access control protocol to allow stations equitable access to the channel were discussed. Variations of the ALOHA and CSMA medium access control protocols were discussed. It was shown that CSMA is superior to ALOHA and, if feasible, CSMA/CD is superior to both.

Spread spectrum techniques were presented next. The impetus behind its growing popularity were discussed. A brief tutorial of direct sequence spread spectrum, its capabilities, possible implementations, and why it is becoming increasingly accessible were also presented. It was shown that by virtue of its process gain, direct sequence spread spectrum offers an effective means to combat interference while simultaneously enhancing network security.

Wireless LANs were discussed next. A brief explanation of the two predominate wireless LAN network configurations (*ad hoc* and infrastructure) is followed by a presentation of wireless LAN standardization efforts in the U.S. (IEEE 802.11) and abroad (HIPERLAN). More germane aspects of 802.11 were presented including the distributed foundation wireless medium access control and the physical layer specifications.

Finally, related research at the MAC sublayer of wireless LANs was presented. Various research efforts investigating how to improve network performance by changing different aspects of the MAC sublayer were presented.

Chapter 3. Objectives and Methodology

After all, the ultimate goal of all research is not objectivity, but truth.
—Helene Deutsch (1884–1982), U.S. psychiatrist.

This chapter discusses the objectives and assumptions of this research as well as the methods used to achieve these objectives. According to Jain [Jai91], there is a systematic approach common to all performance evaluation efforts. He lists ten steps to help avoid common mistakes in this type of investigation [Jai91]. These steps are listed below; the first four are discussed in this chapter with the remaining being discussed in subsequent chapters.

1. State goals and define system boundaries
2. List system services and possible outcomes
3. Select performance metrics
4. List system model parameters
5. Select factors and their values
6. Select evaluation techniques
7. Select the workload
8. Design experiments to offer maximum information with minimal effort
9. Analyze and interpret the data
10. Present the results

3.1 Problem Definition

Chapter 2 discussed the various aspects of wireless LANs, including a brief background of networking systems, spread spectrum techniques, 802.11, and a review of past and present research into MAC protocols for WLANs. One underlying theme permeates these research efforts. They assume the

medium is a fixed asset similar to coax used in Ethernet in that its characteristics cannot be altered dynamically to improve performance. The medium is modeled as a worst-case scenario. Wireless communications, by its very nature, experience wide perturbations which indubitably change the characteristics of the medium. Wireless LANs implemented with spread spectrum technology offer a means to vary the spread spectrum parameters during a communication session to combat these perturbations. A more optimistic approach to medium access is taken in this investigation.

3.1.1 Research Objective and Methodology

The overall objective of this research effort is to determine how and when the spread spectrum parameters can be dynamically adjusted to improve network performance and how this affects the data-link and transport layers of the OSI model. To achieve this objective, this investigation addresses the following specific areas.

A new, unique MAC protocol is developed for wireless LANs to exploit the dynamic nature of the channel. This protocol is based on the proposed IEEE standard 802.11 with modifications and enhancements to the distributed coordination function. Specifically, the improved protocol monitors network performance at the MAC sublayer and dynamically changes spread spectrum parameters at the physical layer to increase process gain when needed. Depending on the severity of channel degradation, time required to reconfigure the spread spectrum transceivers, and length of the data packet, throughput should increase. Since spread spectrum transceivers must agree on certain transmission parameters, such as the PN code, a unilateral decision by one station to change its parameters is not possible. The new protocol ensures proper coordination between stations.

Retransmissions at the MAC sublayer affect the time required to transmit a packet successfully. Since the transport layer also monitors network performance using timers, it might be impacted by these retransmissions. To this end, TCP's performance is investigated to ensure the new MAC protocol does not severely degrade TCP performance.

Adaptively altering the spread spectrum characteristics to improve network performance is a novel approach to wireless LANs. As shown in Chapter 2, other researchers focus on different aspects (MAC, network, and transport layers) of wireless LANs, but do not address how to improve performance by taking an opportunistic view of the channel. This research views the physical layer as a valuable resource to be dynamically altered to match network requirements.

Successful completion of this research effort requires solving new problems in the field of wireless LANs. As mentioned above, a station cannot unilaterally change spread spectrum parameters.

This effort investigates how to coordinate such a change given an already stressed (increased BER) channel. During coordination, stations exchange management frames. Losing one of these frames is a consideration; recovery involves retransmissions which could further degrade network performance. Changing spread spectrum parameters too often can adversely affect overall system performance; not changing them at all yields worst-case utilization of the channel. A compromise is required to maintain a viable network. Informing the transport layer of impending, yet acceptable, delays is another problem. Service primitives can be used to provide this inter-layer communication, but the transport layer must be able to adapt its timeout mechanism. Current implementations of timeout mechanisms allow variation in timeout periods, but may be too slow to adapt.

This research effort extends the existing body of knowledge within the wireless LAN domain. A novel approach to improving network performance is proposed along with supporting studies. Specifically, this investigation extends existing knowledge by defining when and how a direct sequence spread spectrum wireless LAN can take advantage of dynamic channel management. Anticipated benefits include increased overall throughput of the system which leads to a more robust wireless LAN. Although links between stations experience more traffic, the end result is a more reliable link with fewer retransmissions at the transport layer.

This effort uses two proven methods to investigate these problems—simulation and emulation. First, a throughput analysis for wireless LANs is performed using simulation. These performance metrics are explained later in this chapter. A wide range of operating parameters are used to determine an optimal time to initiate adaptive control of the MAC sublayer such that the channel is utilized to its fullest. The second method is to implement an emulation of the adaptive protocol on a wireless network testbed and study the effects of these parameters by measuring system performance on a real system.

Computer networking covers a wide range of scenarios including configurations, protocols, and standards. The following sections define the scope of this research effort and further explain the issues discussed above.

3.1.1.1 Network Under Investigation

This research focuses on performance issues related to wireless LANs. MANs and WANs do not lend themselves well to high-speed wireless networking given today's technology; they also are not forced to contend for one medium as does a LAN. As such, MANs and WANs are not discussed.

3.1.1.2 Transport Protocol

The transport protocol chosen for a particular network often dictates system performance. A protocol is often selected based on its past performance and acceptance by the networking community. Based on its proven past and still promising future, TCP/IP is selected as the foundation for this effort.

3.1.1.3 MAC and Physical Layer Implementations

Maintaining conformance with standards is an underlying objective in order to hasten acceptance by the wireless LAN community. Although still in draft form, the 802.11 standard is followed as much as feasible throughout this effort. Its medium access control is used as the basis of all network transactions.

Several factors determine which physical medium to use. The ability to penetrate walls, performance in the face of multipath, and component availability are just a few reasons this investigation uses direct sequence spread spectrum as the physical layer. Also, the ability to vary the process gain of transceivers dynamically forces the selection of direct sequence spread spectrum.

3.2 System Services

The system services supported include data communications via packet-switched transmission of data frames. Each computer station is allowed to send frames to another station within the wireless network.

Two possible outcomes exist for this service. A successful transmission in which the frame arrives without error at the intended destination is certainly possible. This, after all, is the reason for the network and is the desirable outcome. Another outcome could be that the packet is lost due to a collision or bit errors caused by excessive, corrupting noise in the channel or interfering propagation phenomenon; this is undesirable.

3.3 Performance Metrics

Virtually all research efforts studying LANs use throughput as the predominate metric for determining system performance. This research is no different. Throughput is typically defined as the number of bytes of user data transferred per second from one station to another. Usually represented in bits per second and normalized to the channel speed, this metric determines the ability of the network to move data in spite of overhead.

3.4 System Model

The complexity of local area networks and radio communications necessitates some simplifying assumptions to make modeling tractable. As is seen in the analytical derivations of the MAC protocols (e.g., ALOHA, CSMA) in Chapter 2, such assumptions are common. In addition, the somewhat unsure character (still in draft) of 802.11 necessitates assumptions. Regardless of assumptions made, the thrust of this research is to show a relative improvement in performance. The following sections define the operating assumptions for the system model.

3.4.1 Network Topology

Given the dynamic nature of mobile computers and the interconnecting network, a myriad of network configurations could exist. This effort assumes the stations are fully meshed in a bus-style topology. In other words, all stations hear the transmissions of all other stations. This does not preclude a temporary fade or degradation in a link between two stations. The implication of this is that the hidden terminal situation is avoided. Although important, hidden terminals are not the focus of this effort.

3.4.2 Capture

The spread spectrum receivers will not experience capture of frames. As explained in Chapter 2, capture occurs when two or more frames collide and at least one receiver successfully receives the frame addressed to it. This could be due to one transmitter transmitting at a much higher power level. Without capture, if a collision occurs, all colliding frames are destroyed beyond recovery.

3.4.3 Number of Stations

Network performance is affected by the number of stations contending for the channel. This investigation uses 10 stations as a network for the simulation. This is a common network configuration for simulations [CrW96, KoN92, Ryp92, WaN92, WeW95, WoW95].

3.4.4 Power Considerations

Mobile computing conjures up an image of a person carrying a laptop computer around while maintaining connectivity to a LAN. Since laptops rely on batteries, power consumption is always a concern. Some wireless LAN adapters consume almost as much power as the rest of the computer while continually sensing the medium. Consequently, some adapters incorporate a sleep mode to conserve power. This research effort does not dwell on power consumption, nor does it dismiss it altogether. To this end, network traffic is minimized when feasible to decrease the number of times the transceiver is activated.

3.4.5 Distributed Foundation Wireless Medium Access Control (DFWMAC)

Chapter 2 discusses the mechanisms comprising the DFWMAC. Since the distributed coordinating function (DCF) is the foundation of the 802.11 MAC and offers all requisite capabilities, DCF is used exclusively in this effort. The point coordination function (PCF) is not addressed. The PCF requires a central controlling station such as an access point connected to a wired backbone. In other words, this research effort addresses the *ad hoc* network instead of an infrastructure network.

3.4.6 Transmission Mode

Since LANs typically experience bursty traffic [BeG92], a packet switchingmechanism is used.

3.4.7 Addressing Modes

Three types of addressing are available to a transmitting station: unicast to a single station, multicast to a set of stations, and broadcast to all stations. This effort supports all transmission modes but only modifies unicast transmissions.

3.4.8 Propagation Delay

Since wireless LANs have stations located in close proximity to each other (less that 1000 feet), propagation delay of the radio signal is negligible compared to other system parameters such as the actual frame transmission time. This is a common assumption for wireless LANs [CrW96].

3.4.9 Packet Generation Rates

Each station uses a Poisson process to create packets. This distribution represents a good model for systems such as this that have an uncertain arrival rate [BrP95, Jai91]. The actual rate the packets are generated is determined by the system load which is described in detail in Chapter 5.

3.4.10 Frame Length

Frame lengths are fixed at 8000 bits for this investigation. This represents a reasonable estimate of frame length given the different types of TCP traffic (e.g., FTP, e-mail, TELNET) expected over the LAN and is based on actual network traces [DaJ91, KlT75] and other research efforts [CrW96].

3.4.11 System Queues

All system queues are based on first-come-first-served. Each station is equipped with a finite capacity queue within the MAC layer. Each queue is capable of holding up to 10 frames.

3.4.12 Forward Error Detection and Correcting

Beyond the error detection provided by the CRC (cyclic redundancy check) in 802.11, no other detection or correction techniques are employed.

3.4.13 Destination Addresses

All stations transmit to the station with the next highest address; the last station transmits to the lowest numbered station. For example, station 1 transmits to 2, 2 to 3, and 3 to 1 if there are three stations in the network. When considering the system, the amount of traffic arriving at each station is approximately equal, which closely approximates a discrete uniform distribution. This assumption facilitates a balanced load across all stations; no one station is receiving all frames which could skew results if that station is experiencing problems.

3.4.14 Direct Sequence Spread Spectrum Transceiver Parameters

Transceiver parameters define the operation and limitations of the physical layer. The following sections define these parameters.

3.4.14.1 PN Code Sequence

IEEE 802.11 specifies the use of the 11 chip Barker sequence:

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

The leftmost chip is output first, aligned with the start of the transmitted symbol, and is symbol synchronous (11 chips per symbol). The 11 chips satisfy the FCC requirement of a process gain of at least 10; the Barker sequence has a gain of 11.

This research effort alters the spread sequence during a session. Depending on the channel conditions, each station uses either the 11 chip Barker code or a 63 chip code.

3.4.14.2 Modulation Format and Channel Data Rates

Two modulation formats and corresponding data rates are supported by 802.11. The first is the basic access rate using Differential Binary Phase Shift Keying (DBPSK) modulation that provides 1 Mbps data rate. This basic access rate is supported by all stations. The second access rate is called the enhanced access rate which uses Differential Quadrature Phase Shift Keying (DQPSK) and provides 2 Mbps data rate. This research will only address the DBPSK format (also referred to as BPSK in this effort).

3.5 Summary

This chapter presented the objectives of this research effort as well as the methods used to attain these objectives. As an introduction, the chapter presented a proven systematic approach to computer

systems performance evaluation as described by Jain [Jai91]. Used as a road map for this research effort, Jain's approach was further expanded in subsequent sections in the chapter. An introduction to the problem was presented followed by the novel solution provided by this research. System boundaries were discussed to include assumptions made regarding the type of network to be investigated; this research investigated the performance of an 802.11 wireless local area network operating under TCP/IP. System services (a packet-based communication system) were explained. Throughput was identified as the performance metric followed by a discussion of system model assumptions that were used to make this research effort tractable. The model discussed includes a fully-meshed, 10 station network operating without capture and power control. Frames with a length of 8000 bits of data were injected into the network using a Poisson process.

Chapter 4. CATER—An Adaptive MAC-Level Protocol

ca•ter vb to supply what is required or desired
—Webster's Dictionary

Opportunities multiply as they are seized.
—Sun Tzu

This chapter describes an adaptive protocol developed during this research effort called CATER (Code Adapts To Enhance Reliability). True to its name, CATER provides what is necessary to make a link between two computers more reliable. The reliability is built into the MAC sublayer by dynamically varying the PN code length of frame transmissions. The protocol is best understood by going through examples; thus, three scenarios are presented in this chapter. Following the examples is a top level description presented via flowcharts. Section 4.1 introduces the erratic environment mentioned and how this affects wireless LANs. This section also addresses the rationale for reconfiguring the links between stations. Section 4.2 provides the first insight into CATER's inter-workings by way of examples. Three scenarios are given along with CATER's response. A more formal explanation of CATER is given in Section 4.3 when flowcharts are used to describe its detailed operation.

4.1 The Link Environment within a Wireless LAN

There are two primary causes for a frame error at the receiver in a WLAN—a collision with another frame or bit errors. The exponential backoff mechanism of 802.11 alleviates the collision problem but cannot help with bit errors. Bit errors often result from multipath fading which causes the

received signal power to fluctuate by as much as 30 dB within a span of 1 meter [AnR95, Mit91, Pah85]. If the channel is severely degraded and experiencing numerous bit errors, it is unlikely that retransmissions alone will successfully transmit the frame in question before the retransmission limit is exceeded.

If the channel is well behaved with relatively few bit errors, 802.11's retransmission scheme will suffice. CATER builds on and expands 802.11's capabilities. That is, if a frame experiences excessive retransmissions, 802.11 simply gives up and signals upper layer protocols of this occurrence. Fortunately, the capabilities of DSSS allow us to address the noisy channel by reconfiguring DSSS parameters in real time.

The key feature in CATER is the ability to change the PN code length of a direct sequence spread spectrum signal. The adaptive protocol uses two PN code lengths—the 11-chip Barker code selected by 802.11 and a 63-chip maximal sequence code. Succinctly stated, CATER changes the PN code length of two communicating stations from 11 chips to 63 chips if a frame continually fails transmission. The adaptive protocol is the mechanism that controls when this transition occurs as well as when a station is to revert back to using 11 chips. All stations in the network use 11 chips as the default, reconfigure to 63 chips when necessary, and some time later revert back to 11 chips. Reconfiguring PN code length is done on a per station pair basis; the entire network is not reconfigured.

Selecting the longer PN code length involves an analysis of channel characteristics and current transceiver hardware limitations. The signal-to-noise ratio of a binary phase-shift keying signal increases by 7.58 dB when the DSSS PN code is increased from 11 to 63 chips [Cou93]. Assuming a bit error rate (BER) of 10^{-2} for a distressed channel, the transition to 63 chips decreases the BER to 10^{-5} [Cou93]. A BER of 10^{-5} to 10^{-6} represents a practical minimal error rate of wireless transceivers using BPSK without encoding (forward error correction coding) and is often used in simulations as the BER experienced by network traffic as well as analytical derivations of channel performance [Bis93, BrB95, GuM94, HaK89, Vit85, WaN92]. This conservative lower bound is chosen for an acceptable BER to be 10^{-5}. Thus, CATER uses 63 chips for the longer PN code, although the concept could be applied to other code lengths or to additional code length levels.

A consequence of using a longer PN code is that the effective data rate decreases. For example, assuming a fixed chipping rate of 11.264 million chips per second (Mcps), a PN code of 11 chips results in a data rate of 1.024 Mbps (11.264 Mcps/11). If the PN code is changed to 63 chips, the corresponding data rate is 187 kbps (11.264 Mcps/63). This drastic reduction in the data rate implies that using 63 chips should be considered only when the probability the frame is failing due to bit errors is higher than the probability of collisions. This is accomplished by monitoring the number of frame retransmissions; as the

number of retransmissions increases, so does the probability the frame is failing due to bit errors. To this end, if a link continually experiences an unacceptable BER using 11 chips, CATER reconfigures the communicating stations to use 63 chips.

4.2 Demonstrating CATER

CATER is demonstrated via Figure 4-1 through Figure 4-3 and the accompanying narrative. The figures demonstrate the protocol at various times during frame transmissions from station A to station B. Figure 4-1 and Figure 4-2 depict unsuccessful reconfiguration attempts and the implications. Figure 4-3 illustrates a successful transaction.

The vertical axis represents time, and the circled numbers represent a PN code length change. All transmissions and receptions from that point forward use the specified PN code length.

4.2.1 Unsuccessful Reconfigure Request Frame

Figure 4-1 illustrates CATER being invoked while station A is attempting to send frame 2 to station B. Frame 1 is successfully sent and acknowledged using the 802.11 DCF with 11 chips. Station A then attempts to transmit frame 2. Since the acknowledgment timer expires before an ACK is received from station B, the transmission is assumed to have failed. As noted earlier, this could be a result of collisions or of a poor channel, but neither station has a way of knowing which. Failing on its first attempt, station A selects a backoff time for frame 2 (following 802.11 guidelines) and waits for its timer to expire. Once the backoff timer expires, station A retransmits frame 2. Assuming this transmission also fails, station A once again selects a backoff time and waits. Although only two stations are depicted in this illustration, they are not monopolizing the channel; other stations in the network are also communicating on the same channel.

This adaptive protocol requires each station to maintain several parameters. The first parameter is *Start Reconfigure Limit* (hereafter called *Start*). This parameter controls when a link between two stations should be reconfigured and is measured in failed frames. For example, if it is set to two as used in this illustration, two failed frames are permitted before the source station initiates the reconfigure protocol. Therefore, the next transmission attempt of frame 2 will initiate reconfiguration.

Returning to Figure 4-1, when the backoff timer expires this second time, station A does not retransmit frame 2 again. Instead, it transmits a short reconfigure request frame to station B. Due to its relatively short length, this frame is far more likely to arrive successfully at station B. Moreover, although not currently implemented, increasing the transmitted power for this frame could also increase the probability of successfully receiving it. Immediately after station A sends the reconfigure request frame, it

reconfigures its transceiver to 63 chips and waits for an acknowledgment from station B that it received the reconfigure request frame successfully and is ready to receive a data frame using 63 chips. Assuming the reconfigure request frame does not arrive at station B, the reconfigure acknowledgment timer on station A expires. The station reverts back to 11 chips, selects another backoff time for the data frame, and waits.

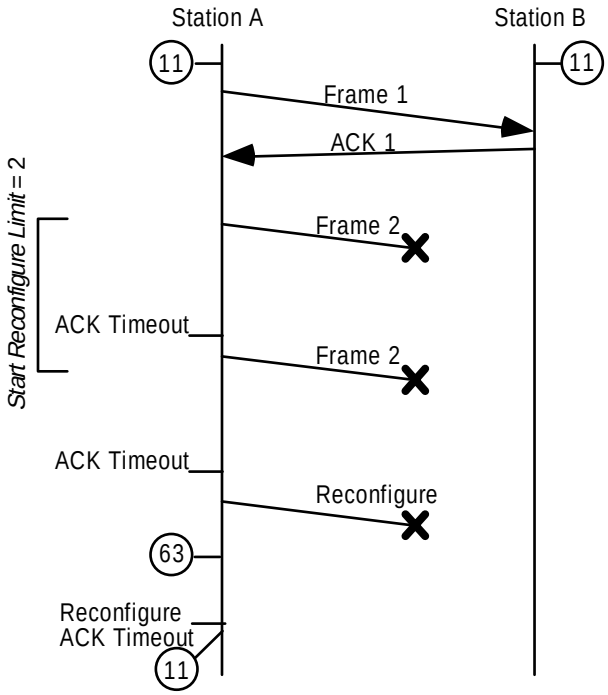

Figure 4-1. Unsuccessful Reconfigure Request Frame

4.2.2 Successful Reconfiguration But Unsuccessful Data Frames

Figure 4-2 continues the example after the backoff timer of station A expires. At this point, frame 2 is once again held at station A, and another reconfigure request frame is sent. Note that once CATER is initiated for a data frame, it remains in control of that frame until it is successfully sent or experiences excessive retransmissions. Moreover, subsequent retransmission attempts for this frame automatically result in a reconfigure request being generated. In the illustration, this is depicted by a reconfigure request frame being sent instead of requiring two more failed attempts at sending frame 2.

This reconfigure request frame successfully arrives at station B, and station A goes to 63 chips at the end of the transmission in anticipation of station B's acknowledgment. Station B meanwhile also reconfigures to 63 chips and then sends the reconfigure acknowledgment.

Before proceeding with the figure narrative, it is vital to understand that station B reverts back to 11 chips after receiving data frames from station A. This brings up an important issue—if a data frame is never received by station B it will be fixed at 63 chips and essentially locked out from the rest of the network; no other stations will be able to communicate with it until they also go to 63 chips. However, station B will never be able to acknowledge a reconfigure request from another station since the reconfigure request frame is sent using 11 chips. To protect against this lockout, Station B starts a *Data Not Received* timer immediately after sending the reconfigure acknowledgment frame. If a data frame is not received before this timer expires, the reconfigure attempt is declared a failure and station B reverts back to 11 chips. Station B uses another network parameter called *Reconfigured Transmissions Limit* to compute the *Data Not Received* timer. This limit represents the number of times frame 2 is allowed to be sent using 63 chips before station A declares the reconfigure a failure.

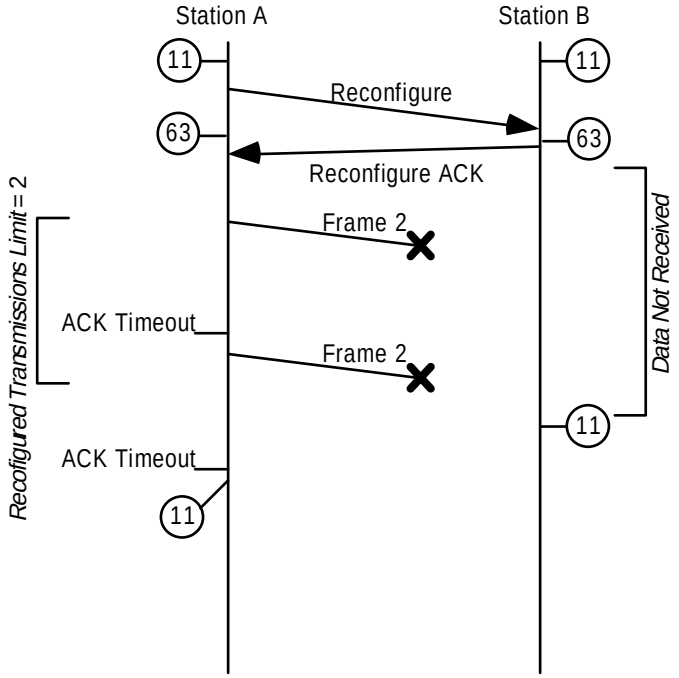

Figure 4-2. Successful Reconfiguration But Unsuccessful Data Frame

Assuming station A receives the reconfigure acknowledgment for station B, it immediately sends frame 2 using 63 chips. If frame 2 fails, instead of selecting a backoff time and unnecessarily repeating the link reconfigure after backoff, station A immediately retransmits frame 2 after sensing the medium for a carrier. If a carrier is present, another station must have sensed an idle medium and started transmitting first. If this is the case and a carrier is present, station A does not attempt to transmit frame 2 a second time. Instead, station A sends the frame to backoff, and reverts to 11 chips. If, however, no carrier is present, station A sends frame 2 and waits for an acknowledgment. Once again frame 2 fails in this scenario. Station A's acknowledgment timer expires, it realizes that it has made the maximum number of attempts at transmitting the reconfigured frame as controlled by *Reconfigured Transmissions Limit*, and it reverts back to 11 chips. At this time, frame 2 is sent to backoff. Meanwhile, station B's *Data Not Received* timer expires causing it to also revert back to 11 chips.

Since multiple, unsuccessful reconfigure request frames could be sent on behalf of a data frame, simply counting the number of failed data frames to determine excessive retransmissions is not feasible. Therefore, in determining when to declare excessive retransmissions, CATER counts not only the number of failed data frames but also the number of failed reconfigure request frames.

4.2.3 Successful Reconfigure Request Frame and Data Frames

Figure 4-3 continues this example. Both stations are using 11 chips at this time. After station A's backoff timer expires, it sends a reconfigure request, reconfigures to 63 chips, and waits for station B's acknowledgment. Upon receiving the reconfigure request from station A, station B reconfigures to 63 chips and transmits a reconfigure acknowledgment. Station A successfully receives this acknowledgment.

Station A is now ready to transmit frame 2 once again. There is, however, an additional step purposely left off previous discussions for the sake of brevity and clarity. Before station A transmitted the reconfigure request frame, it checks its transmit buffer for frames queued to be sent to station B and compares this number against another network parameter—*Max Additional Frames to Transmit During Reconfigure* (hereafter called *Max*). This parameter sets an upper limit on the number of additional frames to send over a reconfigured link. The rationale behind this parameter is that if a link has been reconfigured, it most probably is experiencing excessive bit errors. To just send one frame over this reconfigured link would require future frames (queued frames) to experience the added overhead discussed above before the link is ultimately reconfigured again. For example, station A sends frame 2 only during this reconfigured period, and frame 3 is queued for transmission to station B as well. Frame 3 will have to fail twice (*Reconfigured Transmissions Limit* = 2) before it can benefit from a reconfigured

link. Sending queued frames over the reconfigured link eliminates some of the overhead by allowing additional frames to immediately follow frame 2.

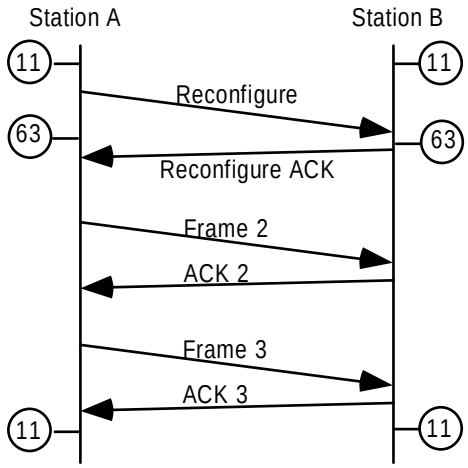

Figure 4-3. Successful Reconfigure Request Frame and Data Frames

Therefore, station A determined how many additional frames to attempt to transmit during the reconfigured period, inserted this number into the reconfigure request frame, and sent it to station B. Station B strips off this field and records this number. At this point, station A sends frame 2. Station B receives the frame and sends an acknowledgment back to station A. If there are no additional frames to be sent to station B, both stations revert back to 11 chips. If, on the other hand, there are additional frames to send, both stations remain at 63 chips, and station A continues to transmit until it sends all of the additional data frames or an acknowledgment is not received from station B. When either one of these occurs, both stations revert back to 11 chips and the last unsuccessful data frame is sent to backoff. Station B uses a feature similar to the *Data Not Received* timer mentioned above to prevent lockout for each data frame.

4.3 Describing CATER

This section presents a top-level description of CATER. Five flowcharts completely describe the details of the protocol and are presented in the following sections.

4.3.1 Process Packet Transmission Request from TCP/IP

As Figure 4-4 illustrates, the primary functions in this flowchart are to accept a packet from TCP/IP or some similar upper level protocols, convert the packet to an 802.11-compliant frame, and transmit the frame after some processing. The first step is to accept the frame from TCP/IP by means of a service primitive request called MAUNITDATA.request. Upon receiving the packet, the protocol encapsulates it within a frame structure suitable for transmission over an 802.11 wireless LAN. At this point, CATER determines if the station is already in a reconfigured state as a result of earlier frame transmissions. A station is reconfigured if the station PN code length is set to 63 chips. If the station is reconfigured, CATER determines if this station is the source station during the reconfigure period. If so, the protocol determines if this frame is one of the additional frames to be sent during reconfiguration. If it is, the frame is sent to the *Transmit Frame* flowchart after a SIFS period has elapsed and the MAC Protocol Data Unit (MPDU) identification number is recorded. The MPDU ID is used to verify that the received acknowledgment corresponds to the last data frame transmission. If the frame is not one of the additional frames to be sent during reconfigure, the frame is sent to the *Backoff* flowchart. This occurs when the maximum number of additional frames to send during reconfigure has already been reached and the station has not had an opportunity to revert back to 11 chips and, therefore, no longer reconfigured.

If the station is not the source station in a reconfigured connection or reconfigured at all, the station attempts to transmit the frame after verifying three criteria. First, the station must sense the channel has been idle for at least DIFS. If this time has not elapsed, send the frame to the *Backoff* flowchart. Second, if the number of transmissions for this frame is less than the *Start Reconfigure Limit* parameter, send the frame to the *Transmit Frame* flowchart after recording the MPDU ID. If the *Start Reconfigure Limit* has been exceeded, create a reconfigure request frame for transmission to the destination station while storing the original data frame in a buffer. Third, a carrier must not be present just prior to transmitting the reconfigure request frame. This ensures this station does not barge in on another pair of stations communicating over a reconfigured link. If a carrier is present, the station sends the data frame to the *Backoff* flowchart and discards the reconfigure request frame.

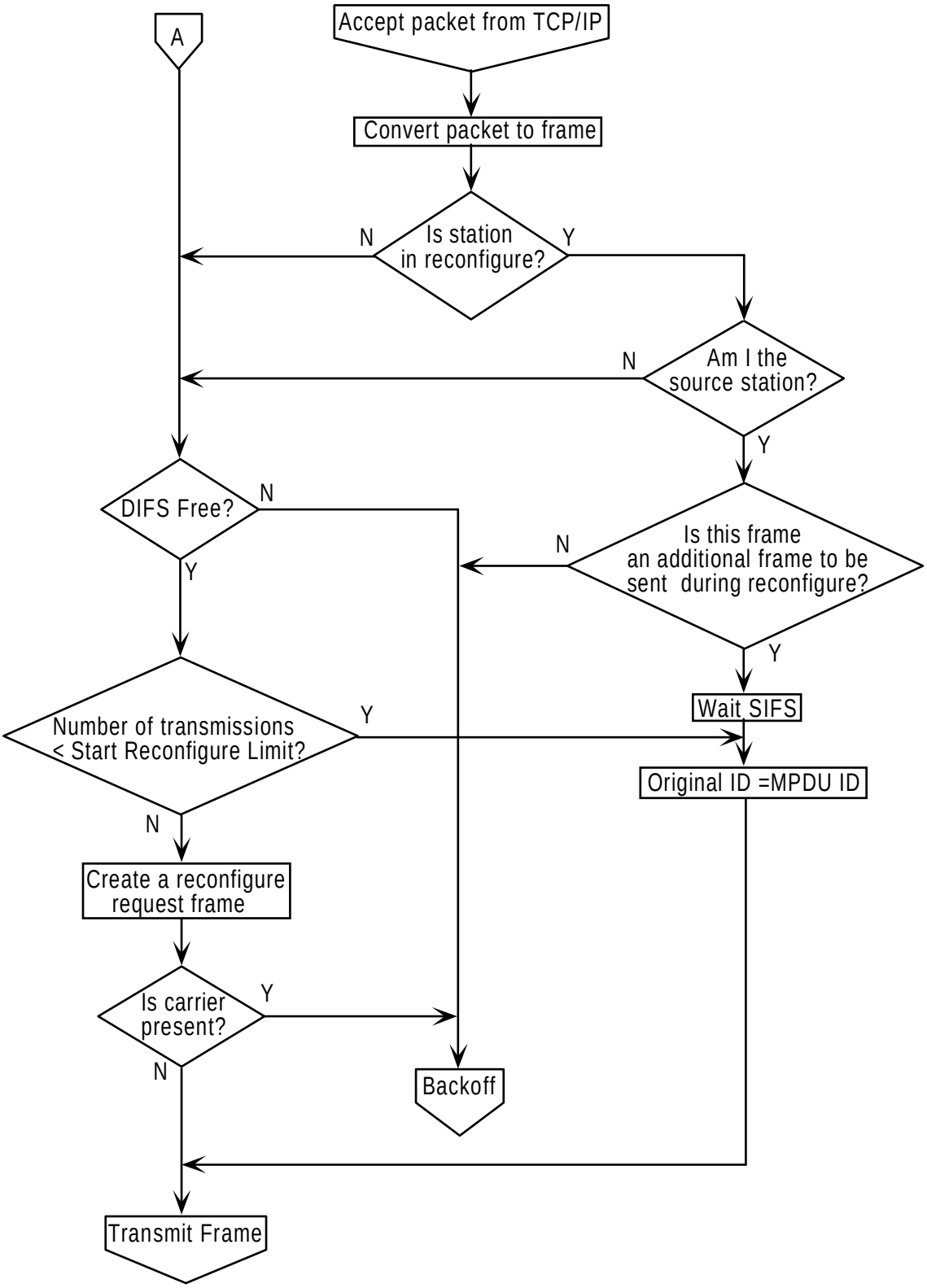

Figure 4-4. Process Packet Transmission Request from TCP/IP Flowchart

4.3.2 Transmit Frame

The *Transmit Frame* function flowchart in Figure 4-5 is responsible for transmitting a frame onto the channel. If the data timer is set on the station about to transmit a frame, the station must have been on the receiving end of a reconfigured link. A request to transmit a frame from this station implies the channel was quiet (no carrier present) for DIFS. As previously mentioned, the data timer is used by the destination station to determine if a reconfigured connection has failed. A channel that has been quiet for DIFS implies the frame that was suppose to be received during the reconfigured period was unsuccessful. Therefore, the reconfigured exchange must have failed and the reconfigured link should revert back. If this is the case, the station cancels the data timer and sets the station PN code length back to 11 chips.

The next step is to increment the number of transmission attempts for this frame. This information is used to determine when to declare an excessive retransmissions situation for a frame.

If the frame to transmit is a reconfigure acknowledgment, the station sets the PN code length to 63 chips and sends the frame to the physical layer for subsequent transmission onto the channel. If this frame is not a reconfigure acknowledgment, the station simply sends the frame to the physical layer without changing the PN code length.

After the frame is completely transmitted, the station determines the type of frame just sent and acts on that information. If the frame was a data frame, the station sets the acknowledgment timer. If the frame was a reconfigure request frame, the station sets the reconfigure acknowledgment timer and changes the station PN code length to 63 chips. An acknowledgment frame will set the data timer if there are additional frames to be received during a reconfigured period. If there are no additional frames to receive during reconfigure, the station changes the PN code length to 63 chips. A reconfigure acknowledgment frame will set the data timer.

4.3.3 Accept Frame From Channel

As shown in Figure 4-6, *Accept Frame From Channel* is responsible for routing incoming frames from the channel to the proper functions. If the incoming frame is a data frame, the station compares the MPDU ID against the last received frame's MPDU ID as stored in the MPDU ID cache. The station discards the frame if the two numbers match. This situation can occur when the destination station successfully receives a data frame, but the source station never receives an acknowledgment for that frame and then retransmits the data frame. Only one data frame is required, so subsequent receptions of the same frame are discarded. If the two ID numbers do not match, the station stores the MPDU ID in the

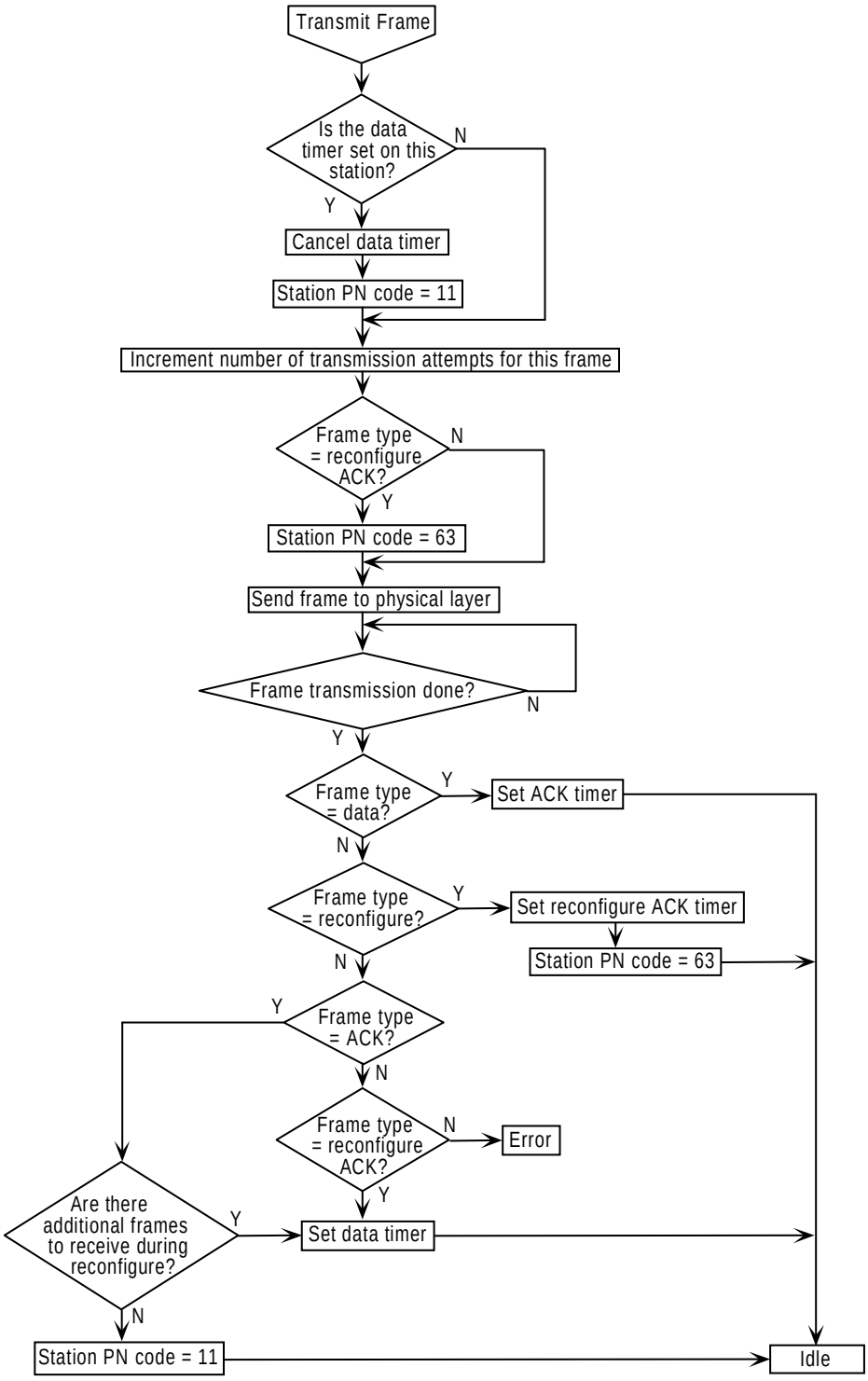

Figure 4-5. Transmit Frame Flowchart

MPDU ID cache for future comparisons, sends the data frame to the TCP/IP stack, and then cancels the data timer if set. The station creates an acknowledgment frame for this data frame and sends it to *Transmit Frame* after waiting SIFS.

If the frame is an acknowledgment, the station compares the MPDU ID of the acknowledgment frame against the stored MPDU ID. These two number should match indicating this acknowledgment is for the last data frame. If the two number do not match, the station discards the acknowledgment frame. Otherwise, the station cancels the acknowledgment timer. If there are no additional frames to send during a reconfigure period, the station changes the PN code length back to 11 chips and sends confirmation to TCP/IP that the data frame was successfully transmitted. The station then requests the next packet for transmission from the MAC buffer and sends it to *Accept Packet From TCP/IP* for processing.

When a reconfigure request frame is received, the station reads the number of additional frames to be sent during the reconfigured period. This information is embedded within the frame. In addition, the station creates a reconfigure acknowledgment frame and sends it to *Transmit Frame* after waiting SIFS.

Finally, if the frame is a reconfigure acknowledgment, the station cancels the reconfigure acknowledgment timer, waits SIFS, retrieves the data frame from local memory, records the MPDU ID, and sends the data frame to *Transmit Frame*.

4.3.4 Backoff

Backoff (Figure 4-7) controls the exponential backoff mechanism as discussed in Chapter 2. An excessive retransmission is declared if the total number of transmissions plus the total number reconfigure request frames sent on behalf of this data frame exceeds the retransmission limit. If this limit is not exceeded, the station computes the backoff time for the frame and starts the timer when the medium has been clear for at least DIFS. The station stops the timer each time the medium is sense busy and restarts it when the medium is clear for at least DIFS. The station sends the data frame to *Accept Packet From TCP/IP* when the backoff timer expires.

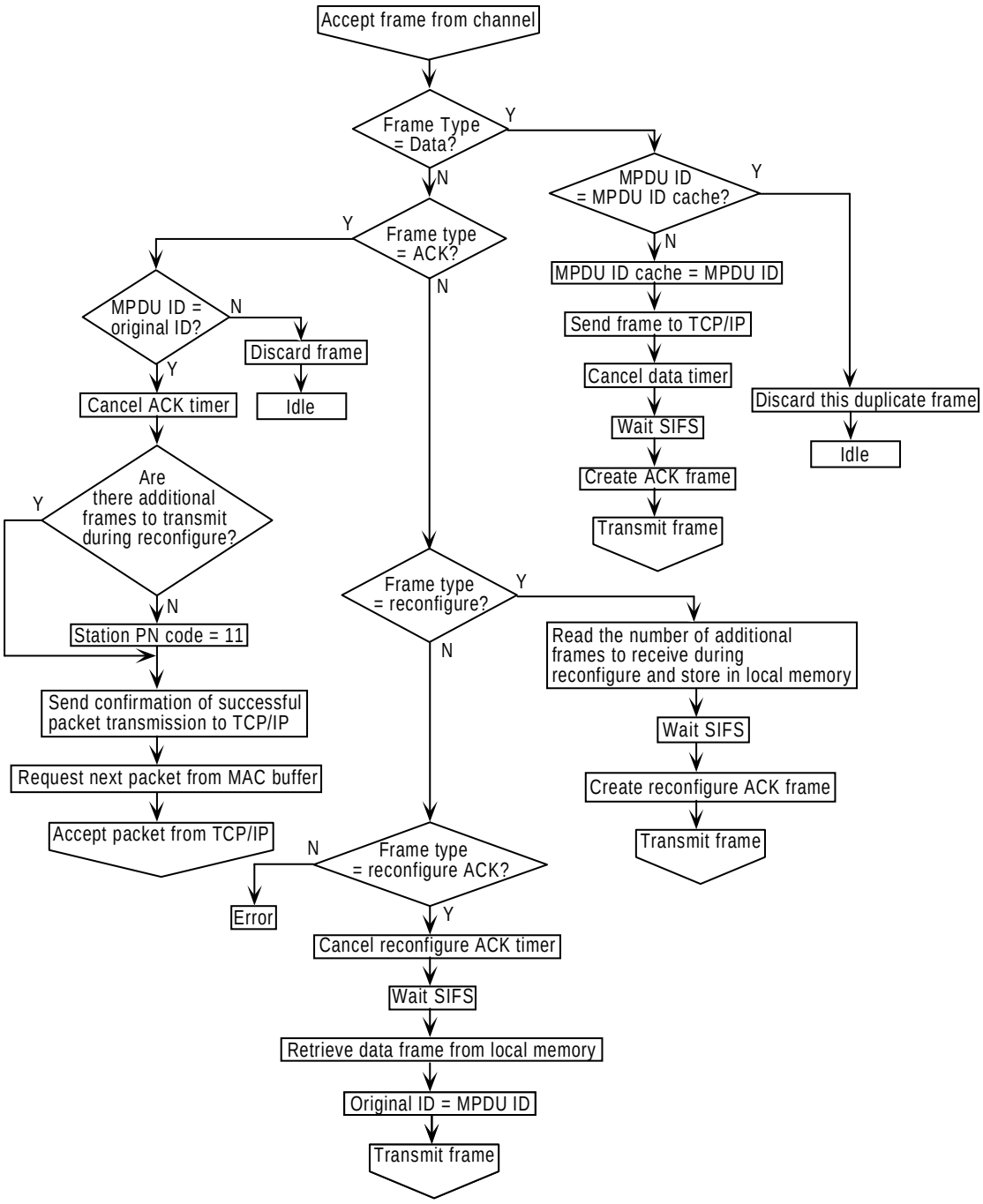

Figure 4-6. Accept Frame from Channel Flowchart

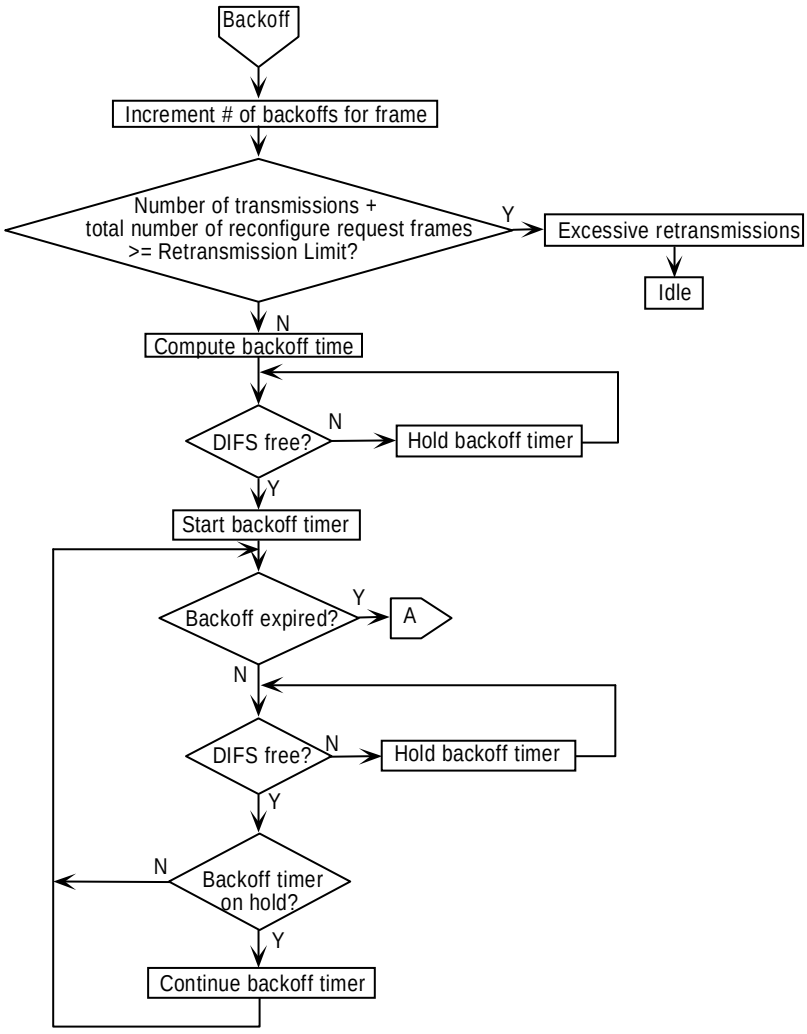

Figure 4-7. Backoff Flowchart

4.3.5 Service MAC Timers

Figure 4-8 illustrates three flowcharts collectively referred to as *Service MAC Timers*. The first is *ACK Timer Expires*. This flowchart describes the actions when the acknowledgment timer expires. If the station is not in a reconfigured state, the station changes the PN code length to 11 chips if necessary and sends the data frame to backoff. The station then sends the data frame to *Transmit Frame* after recording the MPDU ID if all of the following are true: the station is in a reconfigured state, the number of

reconfigured transmissions of this data frame is less than the *Reconfigured Transmission Limit*, the data frame is not an additional frame sent during reconfigure, and a carrier is not present. If any of the aforementioned conditions are not true, the station changes the PN code length to 11 chips if necessary and sends the data frame to backoff.

The second flowchart is *Reconfigure ACK Timer Expires*. This flowchart simply shows that when the reconfigure acknowledgment timer expires, the PN code length is changed to 11 chips and the data frame is sent to *Backoff*.

The third flowchart is *Data Timer Expires*. The PN code length is changed to 11 chips when this timer expires.

4.4 Summary

This chapter presented a description of wireless environments and the adaptive protocol developed during this research effort. The potentially erratic environment in which a wireless LAN must operate was presented along with the affect that this has on wireless LANs. The first insight into CATER's inter-workings was presented by way of examples; three scenarios were given along with CATER's response to each. Finally, a more formal explanation of CATER was given via flowcharts and accompanying narrative. Designed for performance enhancement and robustness, CATER dynamically varies the PN code length of frame transmissions to compensate for an erratic channel. CATER focuses on enhancing system throughput of an *ad hoc* local area network controlled by 802.11's distributed coordination function (DCF). Although the point coordination function of the DFWMAC is not implemented, there is no reason to believe CATER will interfere with any of its functions.

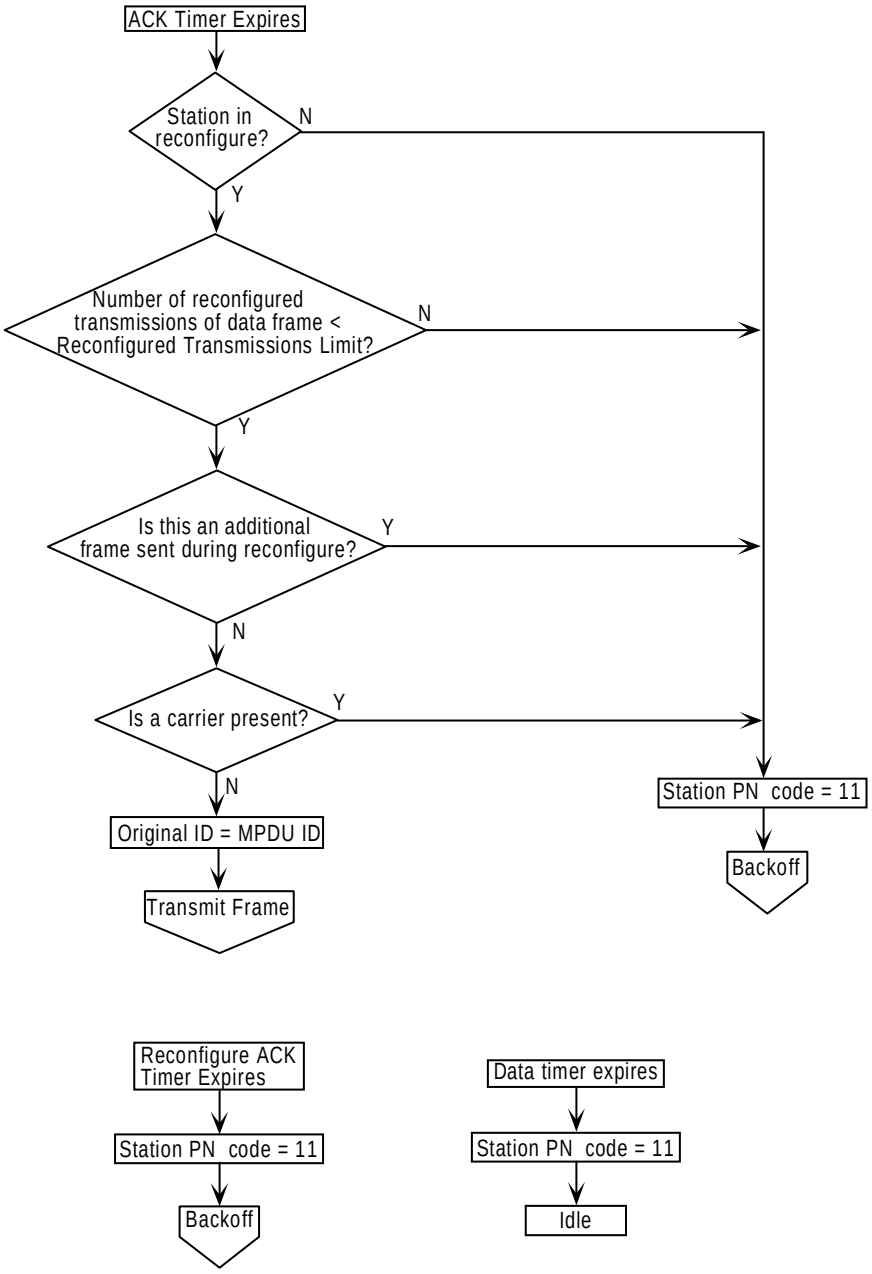

Figure 4-8. Servicing MAC Timers Flowchart

Chapter 5. Simulation

I do not fear computers. I fear lack of them.
—Isaac Asimov

Computers are useless, they can only give you answers.
—Pablo Picasso

Three evaluation techniques are commonly used for computer system research efforts: analytical modeling, simulation, and measuring a real system [Jai91]. Since the research effort detailed in this document involves more than a basic medium access control protocol, analytical modeling is prohibitively time consuming for the level of accuracy provided by the model. This is not to say analytical modeling does not have its place in this effort; simulation results are compared against other models such as CSMA for validation.

The predominate evaluation technique is simulation due to its versatility and accuracy. An entire wireless LAN is modeled via simulation including the CATER MAC sublayer and the DSSS physical layer. The specifics of the simulation are presented in the subsections below.

Finally, the wireless LAN is emulated on a wireless LAN testbed. There is a synergistic relationship between the simulation and emulation; each validates the results of the other. The simulation is ideal for tradeoff evaluations. Once an idea is simulated with good results, it can be transferred to the emulation for further evaluation and validation. Likewise, the emulation evaluation may produce insight into an area of the research not yet considered which in turn can be simulated.

This chapter discusses the simulation of a wireless LAN implementing both native IEEE 802.11 and CATER. The chapter begins with a discussion of the simulation tool in Section 5.1 followed by how the system is modeled within the simulation in Sections 5.2 (data structures) and 5.3 (simulation model). The next three sections present the parameters and variables within the simulation—Section 5.4 presents the parameters; factors are presented in Section 5.5; and Section 5.6 discusses the response variable. The simulation platform is presented in Section 5.7. Sections 5.8 and 5.9 discuss verification and validation, respectively. The chapter concludes with a summary in Section 5.10.

5.1 Simulation Tool

Selecting a proper language or tool for a simulation affects many aspects of the simulation and its results. Two avenues are available for the simulation designer—implement the simulation using either a high order language (HOL) or a tool built specifically for simulations. Writing a custom simulation using a HOL is rarely practical for larger efforts. In most cases, an appropriate simulation tool already exists with numerous application-specific libraries. This is the case for this research effort.

This investigation uses Designer™ version 2.6 [Com93] by Comdisco Systems, Inc. for all simulation studies. (Designer™ has been acquired by Cadence Design Systems, Inc. since the release of version 2.6.) Designer™ is a discrete-event simulator specializing in modeling computer communication networks by capturing the network design using hierarchical, data-flow block diagrams. The hierarchical nature of the tool allows the development of a network system from lower level blocks. The network system is modeled at the top layer and is further decomposed into lower layers. Designer™ models networks by propagating data structures. Analysis is assisted by placing probes within the model to collect statistics and data which are later used by a post-processor to display results.

The following sections describe the data structures and modules used to implement the wireless LAN. Presentation order of the modules follows the same hierarchical character upon which the system is based (top level first followed by lower levels). Only the upper, most significant modules are presented. All supporting modules are found in Appendix A.

5.2 Data Structures

As mentioned above, data structures traverse the simulation model much the same way as actual data packets. Each structure has a variable number of fields to mimic a data packet such as the source address and destination address. Not all fields required in a data packet are, or need be, present in the data

structure. Only enough information to accurately model the network is included. Conversely, to facilitate the gathering of statistics, additional fields are added such as transmission times.

Table 5-1 lists the fields in the most common structure in this effort—a frame. The frame is the only structure exchanged between stations; all other structures are service primitives that allow communication amongst the lower layers of the OSI model. The source address, destination address, data, and data length fields are obtained from the MA_DATA.req primitive. A ROOT-OBJECT data type allows the insertion of any other data structure within the field. This is used to encapsulate a packet within a frame data field. The data length field holds the number of bits within the actual data itself excluding overhead bits. The MAC model appends the remaining information in the appropriate fields; these fields are discussed when appropriate during the explanation of the simulation modules. The 802.11 frame types can be either Data, ACK, Reconfigure, or Reconfigure ACK.

Table 5-1. Frame Data Structure

Field	Data Type
Source Address	Integer
Destination Address	Integer
Data	ROOT-OBJECT
Data Length (in bits)	Integer
Frame Length (in bits)	Integer
Frame Generation Time	Real
Initial Transmission Time	Real
Number of Transmissions	Integer
Number of Backoffs	Integer
MAC Protocol Data Unit ID	Integer
Latest Time Transmitted	Real
Frame Type	802.11 Frame Type
Collision	Integer
Reconfigure Attempts	Integer
Total Reconfigure Frames	Integer
PN Code Length	Integer
Data Generation Time	Real
Additional Frames to Transmit During	Integer

Table 5-2 lists the fields in the MA_DATA.req data structure, which models the service primitive request (hence the .req extension) from the logical link control (LLC) sublayer to the MAC sublayer requesting a packet be sent.

Table 5-2. MA_DATA.req Data Structure

Field	Data Type
Source Address	Integer
Data	ROOT-OBJECT
Destination Address	Integer
Length	Integer

Table 5-3 lists the fields in the MA_DATA.ind data structure, which models the service primitive indication (.ind extension) from the MAC sublayer to LLC sublayer indicating the receipt of a frame. MAC-Reception-Status can only assume the value of Reception OK since the MAC only forwards correct packets up the stack.

Table 5-3. MA_DATA.ind Data Structure

Field	Data Type
Source Address	Integer
Data	ROOT-OBJECT
Destination Address	Integer
Length	Integer
Reception Status	MAC-Reception-Status

Table 5-4 lists the fields in the MA_DATA.conf data structure, which models the service primitive confirmation (.conf extension) from the MAC sublayer to LLC sublayer of the source station conveying the status of the last transmission attempt. Possible values for MAC-Transmit-Status include Transmit OK and Excessive Retransmissions.

Table 5-4. MA_DATA.conf Data Structure

Field	Data Type
Transmit Status	MAC-Transmit-Status
Data Frame Generated Time	Real

5.3 Wireless LAN Simulation Model

This section describes the simulation model implemented in Designer™. The discussion starts at the upper most layer of abstraction (the system) and progresses downward through the model hierarchy. Although all modules are essential for the successful operation of the model, only the most significant modules are shown here; all others are shown in Appendix A.

The *10 stations* module is the "system" for this effort and is shown in Figure 5-1. It is composed of 10 wireless LAN stations (*Wireless Workstation Component*) running CATER and sharing the same channel called Channel Memory. Stations transmit frames by sending them to the Channel Memory where all other stations immediately sense the new arrival and retrieve the frame. Each station is connected to the channel sink via the *Received Data Frame* port; the sink is used to collect aggregate data. When a station successfully receives a data frame, it is forwarded to the *Channel Sink* where a simulation probe collects pertinent data. The *Channel Control* module controls the bit errors over all links when running the simulation with a dynamic bursty channel (i.e., the BER is not static). The list down the left side represents the parameters and memory variables used throughout the simulation. Memory variables are allowed to be altered during the simulation and are denoted with a capital M inside a box. Parameters are set once during simulation initiation and are not allowed to change during the simulation; parameters are represented by a capital P just to the right of an up arrow. This up arrow is an indication that the parameter is exported to the next layer up in the model hierarchy. In this case, the next layer up is the simulation control module where all parameters are set.

5.3.1 Wireless Workstation Component Module

Each wireless workstation, Figure 5-2, is equipped with an *802.11 MAC* and a packet generator (*Traffic Source*) used to simulate transmission requests from the LLC sublayer. Upon initialization, *Traffic Source* starts generating packets for transmission following a Poisson distribution. When a packet is generated, it is sent to the *802.11 MAC* where it attempts to contend for the channel and transmit the packet. Regardless of its success, the *802.11 MAC* reports its transmission status back to *Traffic Source* which schedules another packet for transmission. *Traffic Source* also calculates the round-trip delay experienced by each packet; this data is used to determine how TCP is affected by the enhanced protocol.

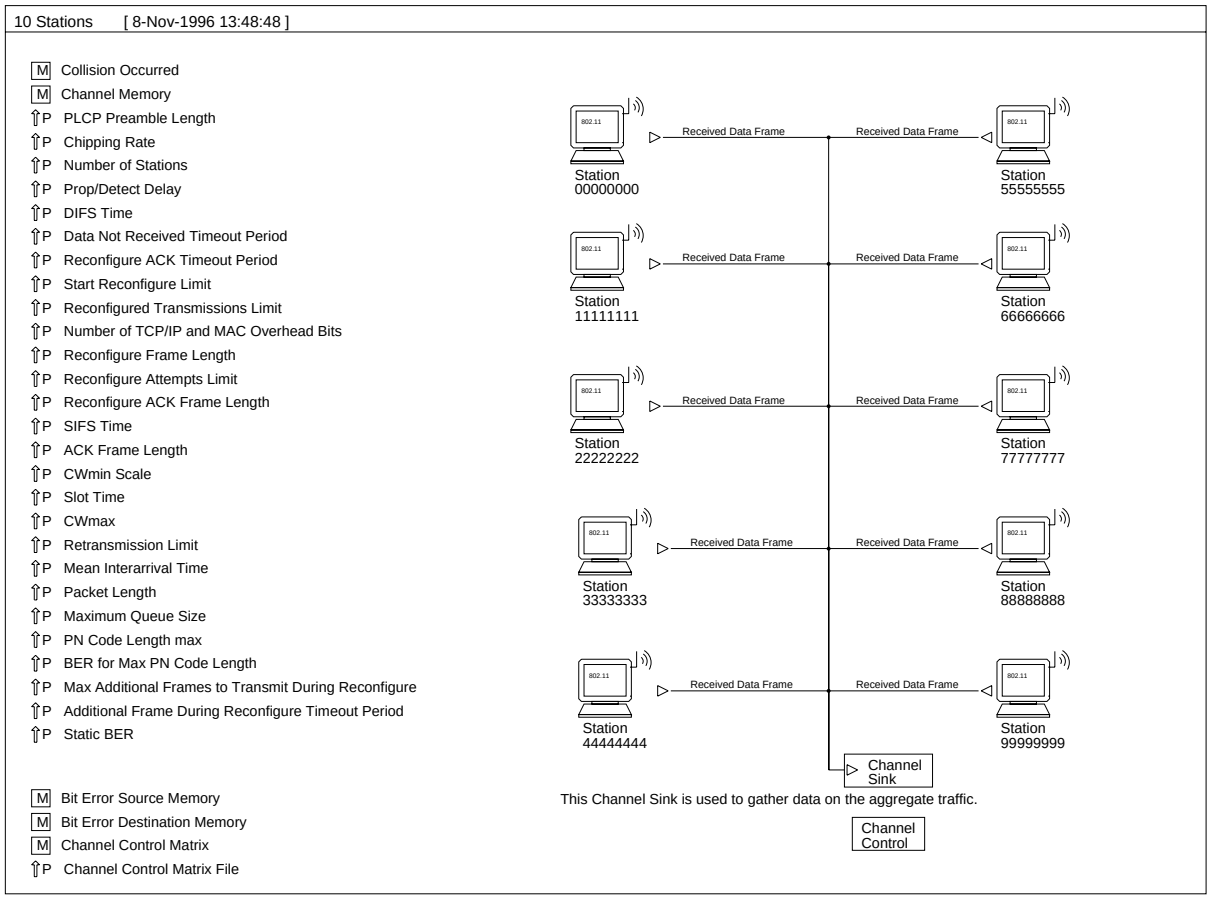

Figure 5-1. *10 Stations* Simulation Module

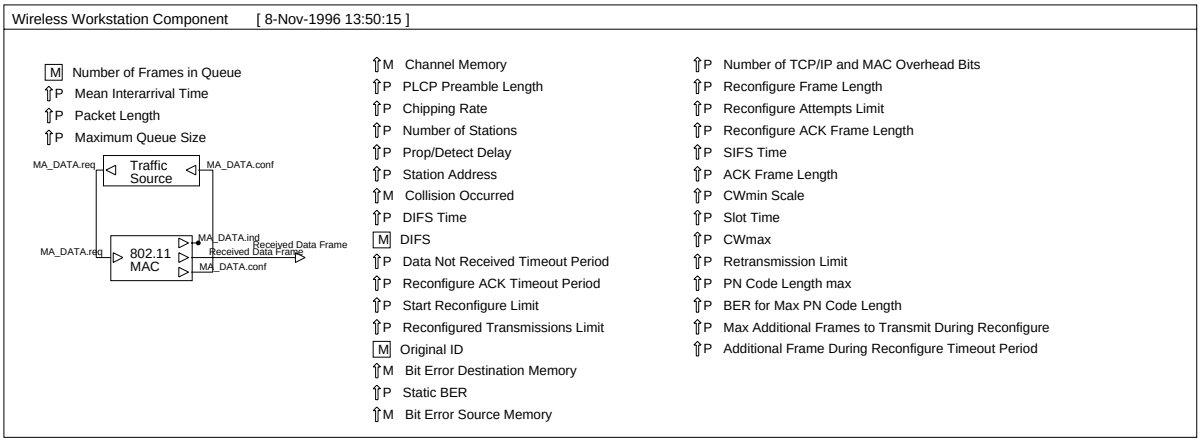

Figure 5-2. *Wireless Workstation Component* Simulation Module

5.3.1.1 MAC Module

Figure 5-3 illustrates how the *802.11 MAC* module is composed of three modules called *Control*, *Transmit*, and *Accept Frame from Channel*. Each performs the task associated with its name and is explained in the following sections.

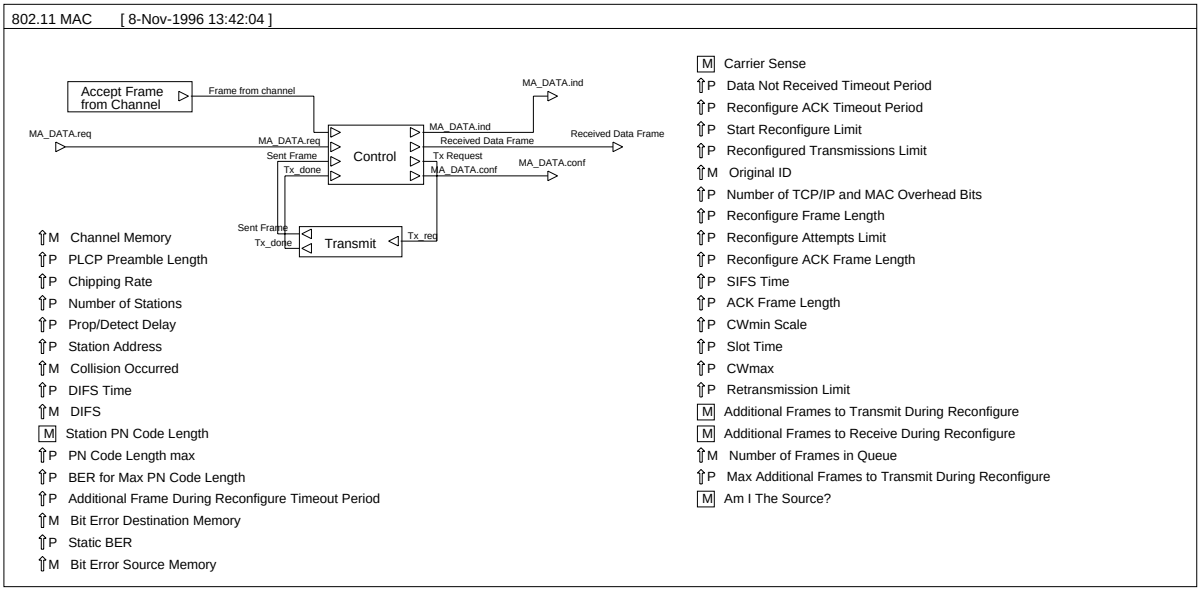

Figure 5-3. *802.11 MAC* Simulation Module

5.3.1.1.1 *Control* Module

The *Control* module is responsible for the overall timing of the *802.11 MAC*. As shown in Figure 5-4, it implements most of the MAC protocol.

Discriminate Type routes the incoming frame (received frame from the channel) to the appropriate output port based on the frame type field. For example, if the frame is a data frame, it is sent to *Data Received.*

Data Received compares the incoming data frames against data frames previously received by this station. Duplicate data frames are discarded. If it is not a duplicate, the data packet is stripped from the frame and sent up the stack in the form of a MA_DATA.ind primitive. In addition, the data frame is sent to the *Channel Sink* via the *Received Data Frame* port. *Data Received* also cancels the data timer, and then requests an ACK transmission by sending the data frame to the *Transmit ACK* module.

Transmit ACK accepts the incoming data frame and generates an ACK frame to send back to the source station. Specifically, this module first waits for the SIFS period to elapse. Then, it creates an ACK frame with all germane fields filled. *Control* then sends this ACK frame to the *Transmit* module after the data timer is canceled using *Cancel Data Timer if Active.*

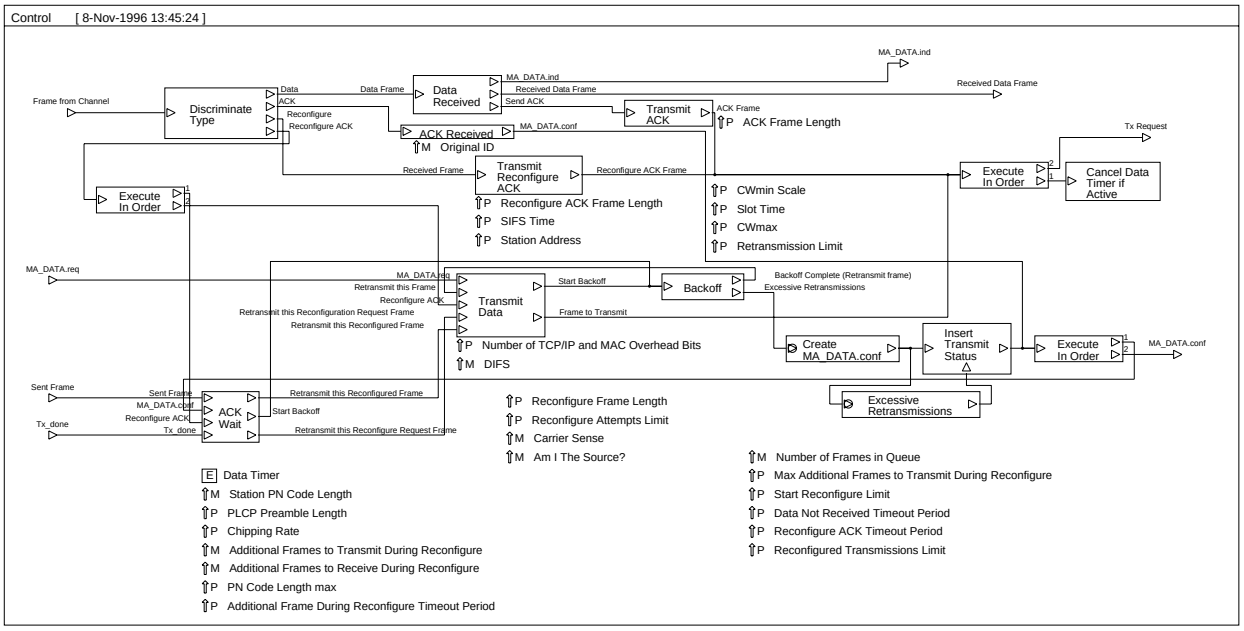

Figure 5-4. *Control* Simulation Module

ACK Received accepts an ACK frame and verifies it is for the data frame last sent from this station. If this is the corresponding ACK frame, a MA_DATA.conf is generated and the time the data frame was generated is inserted into the appropriate field. *Control* first sends the MA_DATA.conf to *ACK Wait* and then to *Traffic Source* requesting another packet for transmission.

Transmit Reconfigure ACK accepts a reconfigure frame. It then reads the *Additional Frames to Transmit During Reconfigure* field and stores this number in the *Additional Frames to Receive During Reconfigure* memory. This mechanism is the means by which communicating stations can coordinate how many additional frames to send after a link is reconfigured. This module then waits for SIFS time to elapse before it creates a reconfigure ACK frame. *Control* then sends this frame to the *Transmit* module after the data timer is canceled.

The *Transmit Data* module (Figure 5-5) implements most of the reconfigure protocol. As such, it accepts one of five inputs, three of which are for the adaptive protocol. There are two possible outcomes from this module—transmit the data or reconfigure frame, or send the data frame to *Backoff*. The MA_DATA.req input accepts this primitive type from *Traffic Source* and converts it to a frame via the *Convert MA_DATA.req to Frame* module. If the station is not in reconfigure (*In Reconfigure?*), it determines if DIFS has elapsed. If it has not, the data frame is sent to *Backoff*. If DIFS has elapsed, *Transmit Data* determines whether to initiate CATER by comparing the number of transmissions for that data frame against *Start Reconfigure Limit*. If it is determined that the station should be reconfigured, the data frame is sent to *Reconfigure Control*; otherwise, the data frame is sent to *Transmit* after the MAC Protocol Data Unit ID (MPDU ID) is recorded. Assuming the station has been reconfigured and it is the source station, an additional frame is sent to *Transmit* after a SIFS delay and the MPDU ID is recorded.

Reconfigure Control is shown in Figure 5-6. This module accepts a data frame on the *Start Reconfigure* port and stores it for future use. The module then creates a reconfigure frame via *Create Reconfigure* and verifies the maximum number of reconfigure attempts have not already been made via the *Reconfigure Attempts Limit* parameter. If this limit has been exceeded, the data frame is sent to *Backoff* after the *Total Reconfigure Frames* field is updated to include the number of reconfigure frames sent on its behalf. If the limit is not exceeded, the medium is once again sensed for a carrier. Assuming the carrier is not present, the reconfigure frame is sent to *Transmit*; else, the data frame is sent to *Backoff* with the *Total Reconfigure Frames* field updated as discussed above. The *Reconfigure Control* module also accepts a reconfigure ACK frame, which causes the stored data frame to be sent to *Transmit* after a SIFS delay. This module will also retransmit reconfigure frames accepted on the *Retransmit this Reconfiguration Request Frame* port after checking for excessive reconfigure attempts as discussed above.

Lastly, this module retransmits data frames that were unsuccessful on previous attempts over the reconfigured link. These data frames enter on the *Retransmit this Reconfigured Frame* port.

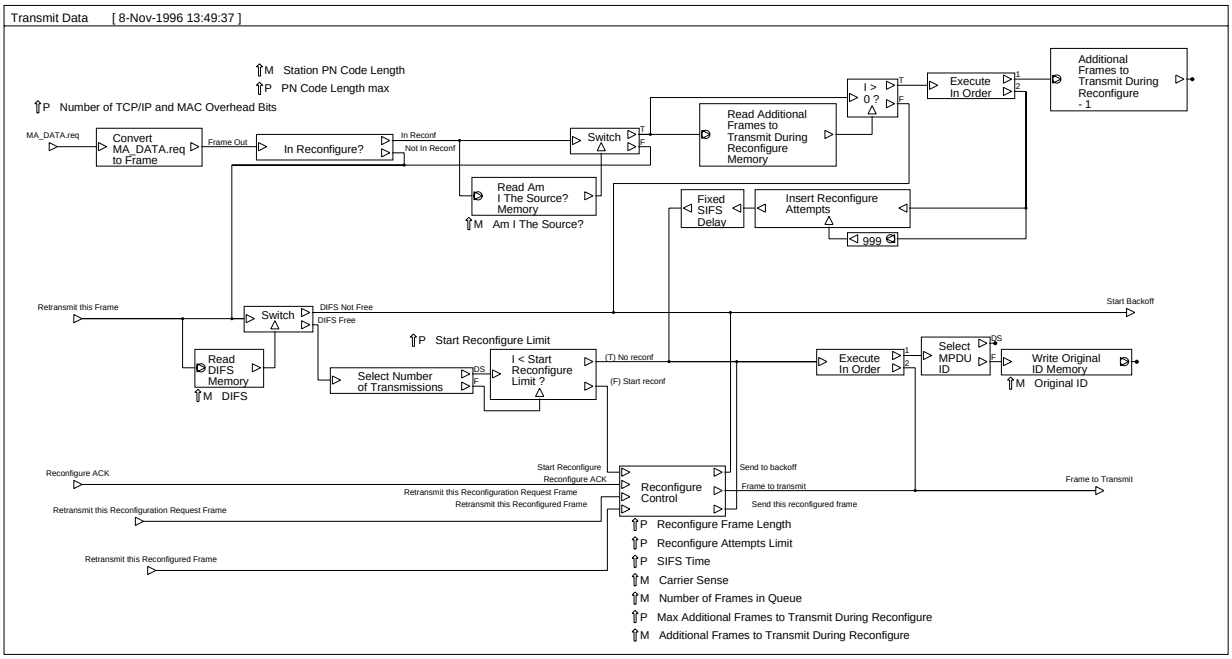

Figure 5-5. *Transmit Data* Simulation Module

The *Backoff* module accepts a data frame and first increments the number of backoff attempts. The module then verifies the sum of the number of transmissions and the number of reconfigure frames sent on behalf of this data frame does not exceed the retransmission limit. If this limit is exceeded, an excessive retransmissions is declared and an appropriate MA_DATA.conf is sent up the stack. If, however, the retransmission limit is not exceeded, the frame is held in memory and *Compute Backoff Time* selects a backoff time. This time is rounded to the nearest slot time using *Round to Nearest Slot Time*; this ensures that if two stations start transmitting almost at the same time, the resulting collision is detected. *Backoff* controls when the backoff timer is allowed to decrement by sensing when DIFS has expired. As dictated by 802.11, the only time this timer is allowed to count down is when DIFS has been detected. In fact, if a station uses the medium while the backoff timer is counting, it must be frozen until a DIFS period is once again detected. Each time the timer is frozen, the residual time is rounded to the nearest slot time as previously explained. Once the backoff timer elapses, the data frame is sent to *Transmit Data* via the *Backoff Complete* port.

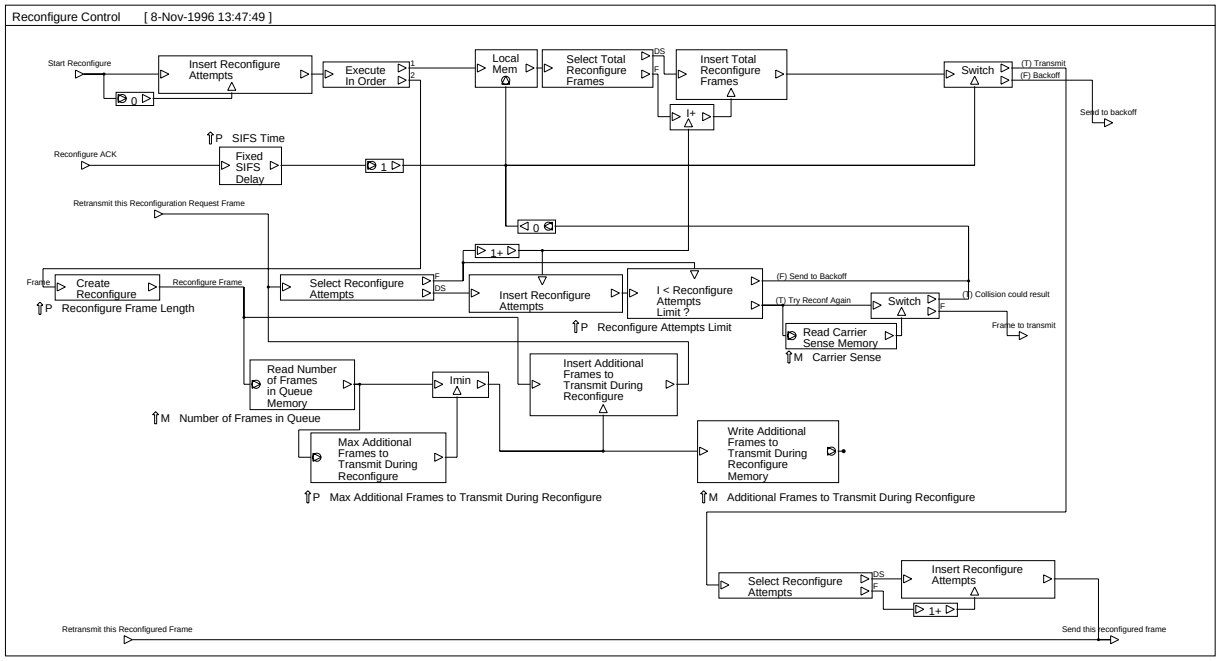

Figure 5-6. *Reconfigure Control* Simulation Module

Compute Backoff Time calculates the appropriate backoff time using the following scheme. A uniformly distributed random number is selected between 0 and $\min(2^{\text{Number Of Backoffs}+4}-1,\ 255)$ where min is a function that returns the smaller of the two arguments. The resulting number is then multiplied by the *Slot Time*.

Round to Nearest Slot Time rounds the incoming time to the nearest slot time and attempts to remove roundoff error. It implements the following equation:

$$\text{Time} = \text{Round}\left[\frac{\left[\text{Round}\left(\frac{\text{Time}}{\textit{Slot Time}}\right)*\textit{Slot Time}\right]}{100000}\right]*100000 \qquad (5\text{-}16)$$

The *ACK Wait* module controls most of the timeout functions of the protocol as well as PN code length selection. After a frame is transmitted using the *Transmit* module, a copy of the frame is sent to *ACK Wait* on the *Tx_done* port. This frame is then sent to the appropriate module functions based on its type. Data frames are sent to *Compute ACK Timeout Period* where an appropriate timeout is calculated;

the ACK timer is then started using this value. ACK frames trigger a process whereby the number of additional frames to receive during reconfigure is checked. If additional frames are expected from the source, the data timeout timer is set; otherwise, the PN code length for the station is set to 11 chips. Reconfigure frames set a reconfigure ACK timeout timer and the station PN code length to 63 chips. Reconfigure ACK frames set the data timeout timer.

The *ACK Wait* module also cancels the reconfigure ACK timeout timer when it receives a reconfigure ACK. After receiving a MA_DATA.conf primitive, it cancels the ACK timeout timer and then resets the station PN code length to 11 chips if there are no additional frames to be sent over the reconfigured link.

Before a frame is transmitted using the *Transmit* module, a copy of the frame is sent to *ACK Wait* on the *Sent Frame* port. This frame is then routed to the appropriate module functions based on its type. If the sent frame is a reconfigure ACK, *ACK Wait* sets the station PN code length to 63 chips. Data and reconfigure frames are stored in local memory. If the reconfigure ACK timeout timer expires, the station PN code length is set to 11 chips and the stored reconfigure frame is sent to *Transmit Data* via the *Retransmit this Reconfigure Request Frame* port. If the ACK timeout timer expires, the stored data frame is sent to *Backoff* if the station is not in reconfigure or if the *Reconfigured Transmissions Limit* is exceeded. If this limit is exceeded, the station PN code length is reset to 11 chips before the data frame is sent to *Backoff*. Assuming the station is in reconfigure and the limit has not been exceeded, the data frame is sent to *Transmit Data* on the *Retransmit this Reconfigured Frame* port if a carrier is not present. If a carrier is present, set the station PN code length to 11 chips and send the data frame to *Backoff*. *ACK Wait* also resets the station PN code length to 11 chips if the data timeout timer expires.

Compute ACK Timeout Period calculates the appropriate time to wait before declaring a data transmission a failure. This module first determines the transmission time of the ACK frame by dividing the total number of bits in the ACK frame by the data rate of the link at that moment. Two SIFS periods are then added to the transmission time. One is the specified period between the end of the data frame and the beginning of the ACK frame; the other is a time buffer. To this end, the module implements the following equation:

$$\text{ACK Period} = \left\lceil \frac{\textit{PLCP Preamble Length} + \textit{ACK Frame Length}}{\left(\frac{\textit{Chipping Rate}}{\textit{Station PN Code Length}} \right)} \right\rceil * \left(2 * \textit{SIFS Time}\right) \qquad (5\text{-}17)$$

5.3.1.1.2 *Transmit* Module

The *Transmit* module, Figure 5-7, is responsible for the actual transmission of frames onto the channel. This module accepts the frame from the *Tx_Req* port, updates its transmission time fields, increments the number of transmissions field, and sends the frame to Control via the *Sent Frame* port. *Transmit* then updates the PN code length field and sends the frame to *Frame Transmission Delay*. The module then determines if this station is the source or destination station based on the frame type being sent. Before sending the frame to the channel (a global memory named *Channel Memory*), *Transmit* determines if a collision has occurred using the *Check for Collisions* module.

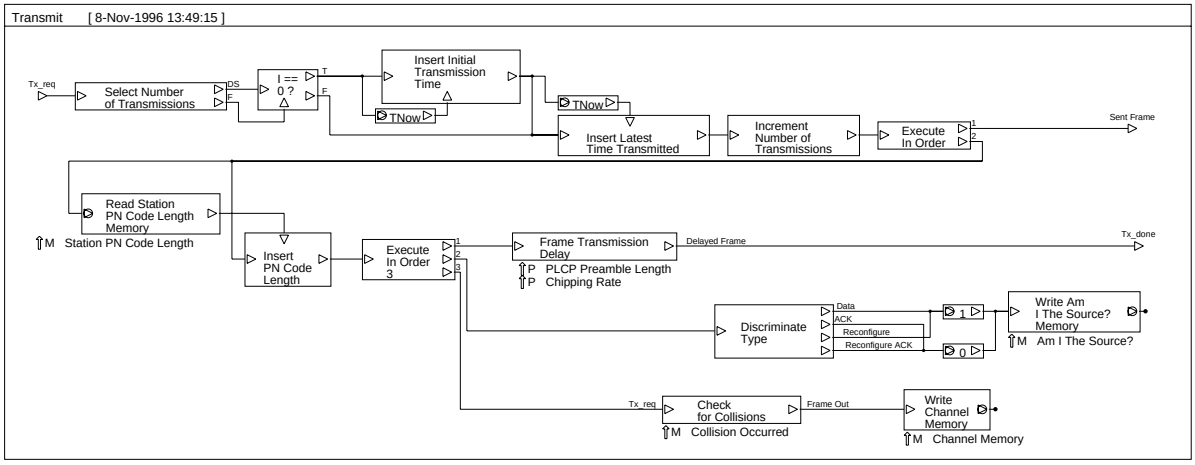

Figure 5-7. *Transmit* Simulation Module

The *Frame Transmission Delay* module calculates the time required to transmit a frame given the frame length, chipping rate, and PN code length. Specifically, it uses the following equation:

$$\text{Transmission Time} = \frac{\text{Frame Length} + \textit{PLCP Preamble Length}}{\left(\frac{\textit{Chipping Rate}}{\textit{PN Code Length}}\right)} \qquad (5\text{-}18)$$

Frame length is the sum of the packet length and the overhead bits. Once the transmission time is calculated, this module holds the frame until that time has elapsed at which time the frame is sent out the *Delayed Frame* port.

The *Check for Collisions* module compares the contents of *Channel Memory* versus the frame about to be sent. If both frames have the same transmission time, a collision has occurred. The *Round to Nearest Slot Time* module discussed earlier ensures that transmission times can be offset by up to one slot

time and still result in a collision. If a collision does occur, the frame length of the last frame is updated to reflect the larger of the two frames; this mechanism is used to ensure that all stations sense the channel is occupied for the larger of the two frames. Also, the frame is marked as being in a collision by writing a 1 to the *Collision* field. A global memory called *Collision Occurred* is updated to indicate whether a collision happened. *Collision* and *Collision Occurred* are used in *Accept Frame from Channel* to determine if the frame is recognized upon reception. The reason two mechanisms are required is that the first frame sent over the channel cannot see subsequent frames sent at the same simulation time; therefore, the first colliding frame is discarded in *Accept Frame from Channel* by using *Collision Occurred* and future colliding frames are discarded using the frame field *Collision*.

5.3.1.1.3 *Accept Frame from Channel* Module

The *Accept Frame from Channel* module, Figure 5-8, is responsible for discarding frames received from the channel that have either collided, are not addressed to that station, were sent with a PN code length that does not match the receiving station's PN code length, or contain errors.

This module first reads a frame sent to the channel as soon as it is written by the source station. If the receiving station is active (i.e., address is less than the number of stations in the network), the frame is allowed to pass through; if not, the frame is discarded. This feature is used to improve simulation performance. The absolute delay module is a first-in first-out queue used as a synchronization mechanism, since the propagation and detection delay time is set to 0. This delay module accumulates all transmitted frames until the simulation clock increments indicating the end of the collision vulnerability period. As soon as all frames sent during the same simulation time are received by the delay module, they are sent to *Purge Initial Collided Frame* where the first frame sent is discarded if a collision occurred.

The module *Carrier Sense and DIFS Control* is responsible for determining when a carrier is sensed on the channel and when DIFS time has elapsed. After accepting a frame, this module updates the DIFS memory and carrier sense memory, which are used by various parts of the simulation. Then, the frame is sent to *Frame Transmission Delay* to simulate the time required to transmit the frame. A copy of the frame is then sent to *Check Frame Error* to mark the beginning of the frame. Then, the station PN code length is recorded at the beginning of the frame. After *Frame Transmission Delay* releases the frame (simulating the end of the transmission time), the current station PN code length is compared against the code length at the beginning of the frame. The result of this comparison is written to memory for later use in *Check PN Code*. The module then updates the carrier sense memory to reflect that a carrier is not present, starts the DIFS timer, and sends the frame to *Select Collision* via the *End of Frame* port.

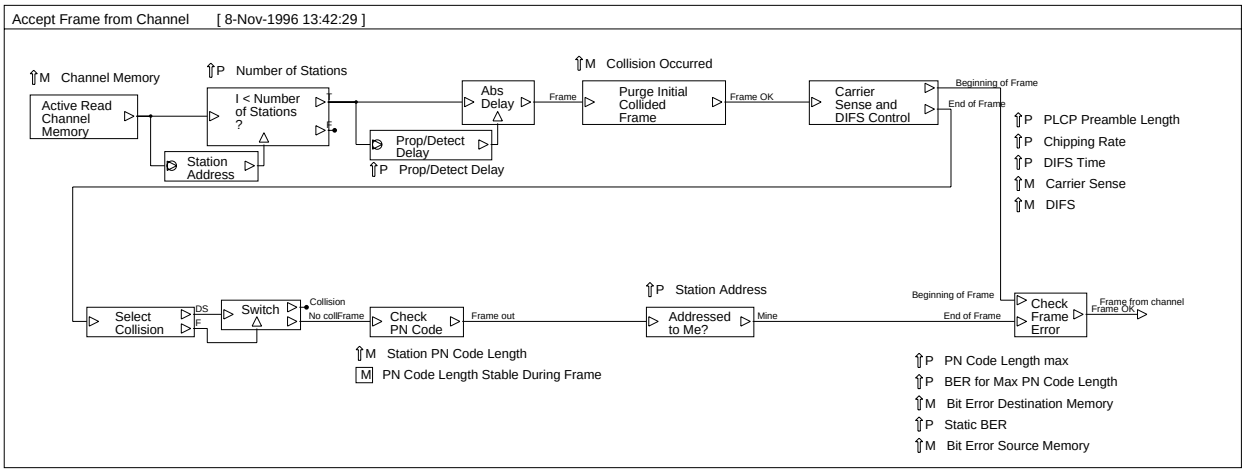

Figure 5-8. ***Accept Frame from Channel*** **Simulation Module**

The *Select Collision* and *Switch* modules simply discard any frames that have their collision field set. It is vital that these frames are not discarded until after *Carrier Sense and DIFS Control* because although the colliding frames are not readable, they still occupy the channel, which affects carrier sense and DIFS.

The next two modules filter out frames based on PN code length and destination address. *Check PN Code Length* verifies the PN code lengths of the source and this station match. Moreover, it verifies that the same code is used throughout the entire transmission of the frame using the *PN Code Length Stable During Frame* memory. *Addressed to Me?* verifies this frame is addressed to this station.

Check Frame Error (Figure 5-9) determines if an error has occurred within the received frame. It has two modes of operation—static BER and dynamic BER. The static BER mode is implemented if the *Static BER* parameter is set to anything other than 1. If this is the case, a binomial distribution is used to determine if an error has occurred within the frame. The module *Binomial Rangen* accepts two values; they are the number of bits in the frame and the BER. The total number of bits is found by adding the frame length and the PLCP preamble length. The BER is either the *Static BER* parameter if the PN code length is set to 11 or the *BER for Max PN Code Length* if the PN code length is set to 63.

The dynamic BER is found by monitoring the channel performance via the *Channel Control* and *Initialize and Wrapup* modules. These modules simply open and read a data file containing the times at which bit errors occur as well as which stations are affected and posts this information to the network via

the Bit Error Memories. The format of the file is the following information on one row: simulation time when the error occurs, source station, and destination station. For example, a valid entry could be 0.000123 3 4 representing a bit error at simulation time 0.000123 between stations 3 and 4.

Setting *Static BER* to 1 forces the simulation to use the dynamic bit errors read from the file. Each time a bit error is read from the file and posted to the simulation, each station reads this error and determines if it affects any frames currently being received. A bit error between two stations affects traffic in both directions; as in the example above, an error in a frame sent to station 4 from station 3 could also cause an error in a frame going in the opposite direction at that moment. If any errors occur in a received frame, the frame is discarded.

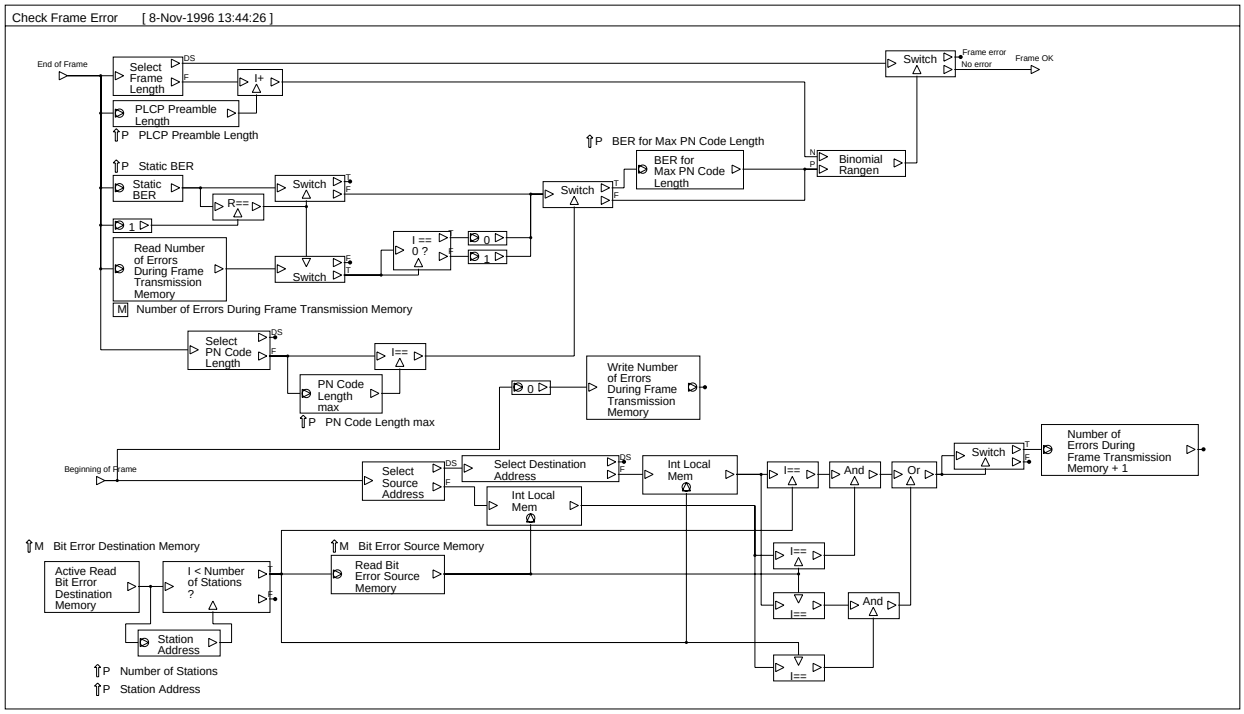

Figure 5-9. *Check Frame Error* Simulation Module

5.3.2 Wireless Channel Model

Simulating a wireless channel is an issue drawing much attention especially with the current heightened emphasis on ubiquitous computing. Numerous techniques exist to characterize channel

behavior and performance at various levels of abstraction. This research effort focuses on channel performance as it pertains to bit errors in a wireless LAN environment.

The symmetric binary channel is the classic model for noisy binary channels. A memoryless model, the symmetric binary channel has the same probability of generating a bit error over all time. In other words, the probability of error is constant and independent of all previous bits. In practice, however, this model is not suitable since channels do exhibit memory implying bit errors can and do occur in isolated bursts thereby negating the error independence assumption of a symmetric binary channel [Gil60].

Gilbert pioneered channel modeling using Markov chains in the early 1960s. He developed a model of a burst-noise channel to characterize error clustering using a 2-state Markov chain [Gil60]. As shown in Figure 5-10, the two states are denoted as G (good) and B (bad). A binary noise digit is generated before each transition of the model. This digit corresponds to a bit error if it is 1 and no bit error if it is a 0. The noise digit is always 0 in state G; state B may generate a noise digit of 1, or not. Once in state B, an error noise digit (1) is generated with probability 1-*h* thereby modeling the fact that good noise digits exist within an error burst. The bursty characteristic is achieved by keeping the inter-state transition probabilities (P and p) small forcing a clustering of bad states and, consequently, a clustering of good states. Adding (modulo 2) the noise digits to the source binary stream produces the output of the channel and the input into receiving station.

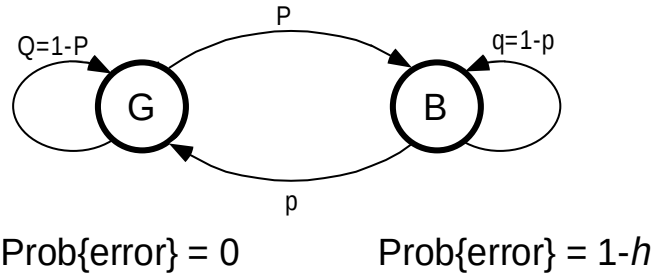

Figure 5-10. Gilbert Model of Burst Noise Channels [Gil60]

Since the introduction of Gilbert's model, several other Markov models have been developed that simulate a channel (both wired and wireless) using increasingly sophisticated models and mathematics [BeM63, Ell63, Fri67, KaS78, MuB89, Tur88, Tur90, TuS93, WaM95]. To validate these models, many researchers have applied empirical data and analytical models against the developed models ranging in

complexity from telephone lines to wireless radio to satellite links [ChH72, DrM88, DuR96, SwF91, SwF93, SwF94, TrL95, Tsa69].

This research effort, as have others [CrW96], uses a Gilbert model to emulate the bursty channel. Although simplistic in design, the Gilbert model provides sufficient granularity for bit errors. Since a single bit error will render an entire 8000-bit frame in error, it is not essential to determine the number or pattern of bit errors within the frame. Other Markov models are more complex and offer a better match to true channel performance, but at the price of more complexity. In addition, the number of parameters (e.g., P, p, and *h* as used in the Gilbert model) that must be set increases. Determining these parameters is either accomplished via comparison with empirical data or through analytical models. Since neither of these techniques are currently available for the environment in question, a sensitivity analysis is performed using the Gilbert model's three parameters—P, p, and *h*.

Although the three Gilbert parameters could be varied over a wide range of probabilities, only a small subset is used. For example, if the probability of a transition from the good state to the bad state (P) is greater than vice versa (p) (i.e., P > p), the resulting BER approaches unity and the channel is unusable at that point. The error rate within the bad state (1-*h*) takes on values of either 0.2 or 0.8; these two values provide maximal coverage of bit error rates using just two settings. Table 5-5 illustrates the parameter values applied to the simulations. Each row in the table represents an individual simulation run. These values are entered into a small C program that generates the aforementioned ASCII error file. The column labeled file size is the size of the resulting file in kilobytes. This file is then used by the Designer™ simulation to model a bursty channel. The average BER can be estimated from the three Gilbert parameters using the following expression [Gil60]:

$$(1-h)*\left(\frac{P}{p+P}\right). \tag{5-19}$$

To summarize, twelve error files using the three parameter values indicated in Table 5-5 are used during simulations. The average BER was calculated for each row and verified against the data provided by the small C program.

5.3.3 TCP Model

As discussed in Chapter 2, TCP is responsible for the reliable end-to-end transport of data, error recovery and flow control by providing an idealized, full-duplex connection between two end systems [DiL92]. The simulation does not model a fully functional TCP. Instead, the delay, also known as round trip time (RTT), associated with each TCP packet is recorded during simulations for later analysis;

specifically, the time required to receive an acknowledgment for each TCP packet is recorded. By not implementing TCP within the simulation the true performance improvements of the adaptive protocol can be investigated without calling into suspicion any TCP interactions. Moreover, the degree of simulation complexity is kept at a manageable level; adding TCP could double the complexity and, therefore, increase the runtime with marginal increase in simulation accuracy over simply collecting RTT values. The RTTs are analyzed using a spreadsheet designed to calculate when TCP would timeout and force another retransmission.

Table 5-5. Gilbert Model Parameter Values

P	p	1-*h*	Average Bit Error Rate	File Size (KB)
0.000001	0.001	0.2	0.0002	1,540
0.000001	0.001	0.8	0.0008	7,463
0.000001	0.01	0.2	0.00002	168
0.000001	0.01	0.8	0.00008	713
0.000001	0.1	0.2	0.000002	18
0.000001	0.1	0.8	0.000008	71
0.0001	0.01	0.2	0.002	15,999
0.0001	0.01	0.8	0.008	64,092
0.0001	0.1	0.2	0.0002	1,600
0.0001	0.1	0.8	0.0008	6,485
0.001	0.1	0.2	0.002	15,984
0.001	0.1	0.8	0.008	63,847

The spreadsheet implements the retransmission timer algorithm found in the 4.4BSD-Lite release as described in [WrS95]. This version was released to the public in April 1994 and contains enhancements to the 1988 4.3BSD Tahoe release, the 1990 4.3BSD Reno release, and the 1993 4.3BSD release [WrS95]. The algorithm [Ste94] is

$$Err = M - A,$$

$$A = A + g * Err,$$

$$D = D + h * (| Err | - D),$$

$$RTO = A + 4 * D,$$

where M is the measured round-trip time (RTT) for a packet and A is the smoothed, or average, round-trip time. The value g is the gain associated with the average and is held constant at 0.125; h is the gain for the

deviation and is assigned the value 0.25. D is the smoothed mean deviation, and RTO is the retransmission timeout value.

5.4 Simulation Parameters

Two types of simulation parameters are used in this effort. The first controls such simulation behavior as when to stop. The second is a parameter that defines a physical attribute or characteristic (e.g., number of bits) of the system. Both types are discussed in the following subsections.

5.4.1 Start Time

Start time refers to when data collection begins within the simulation. The obvious answer of the beginning is not always correct, especially if analysis based on steady-state performance is desired as is the case in this effort. Transient removal is used to eliminate initial data to allow the system to reach steady-state.

To decrease the effect and duration of the transient, the simulation is started in a state closely approximating steady-state. Each station is allow to send one frame in turn. For example, station 1 sends one frame at startup; station 2 sends when station 1 finishes and so on until all stations have sent at least one frame. If all stations are allowed to send at startup, a massive collision would result causing a flurry of backoffs and retransmissions. Eventually, the stations would either successfully send or give up due to excessive retransmissions. In either case, the time required for the system to settle into steady-state would be greater than the initialization scenario discussed above. Although this technique lessens the duration of the transient, it may not be representative of steady-state particularly for lightly loaded systems. Therefore, the transient must be removed.

Figure 5-11 illustrates the behavior of the simulation at startup for system loads of 0.1 and 0.9 when *Start* is set to 5 (*Max* = 6) and 999. The legends indicate the load and *Start* levels (e.g., 0.1-999 indicates the simulation was run using load = 0.1 and *Start* = 999). Figure 5-12 shows the same data but is zoomed in on the region between 0 and 2 seconds. The figures depict the instantaneous throughput over 0.1 second intervals. It is seen that the network reaches steady-state after one second. Therefore, all simulations start collecting data after this one second transient is removed.

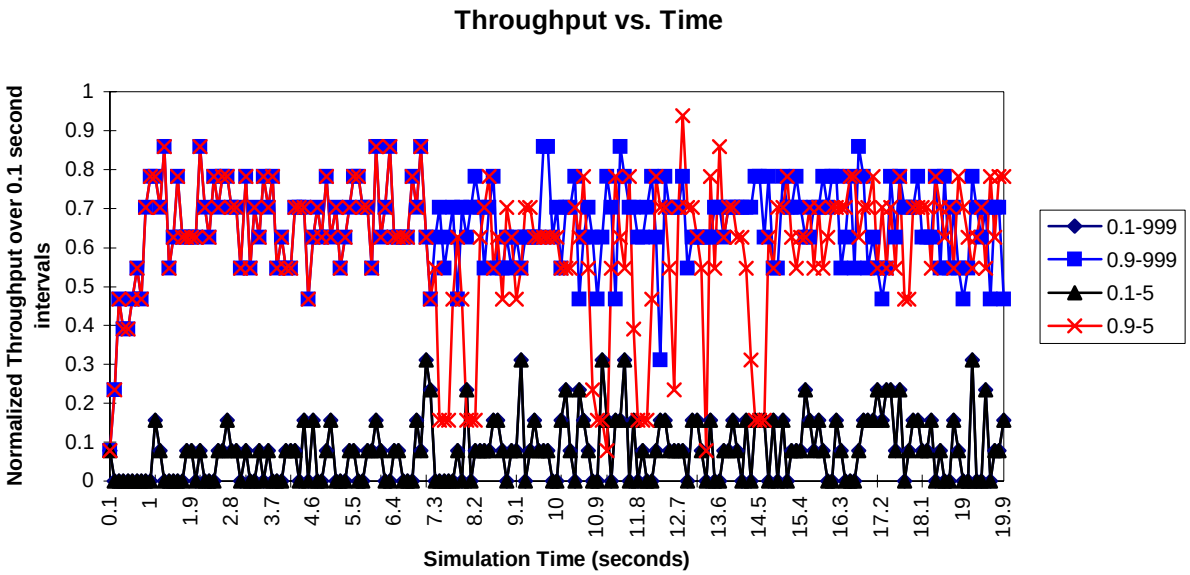

Figure 5-11. Normalized Throughput versus Time over 0.1 second intervals (*Start*=5 and 999 Load=0.1 and 0.9)

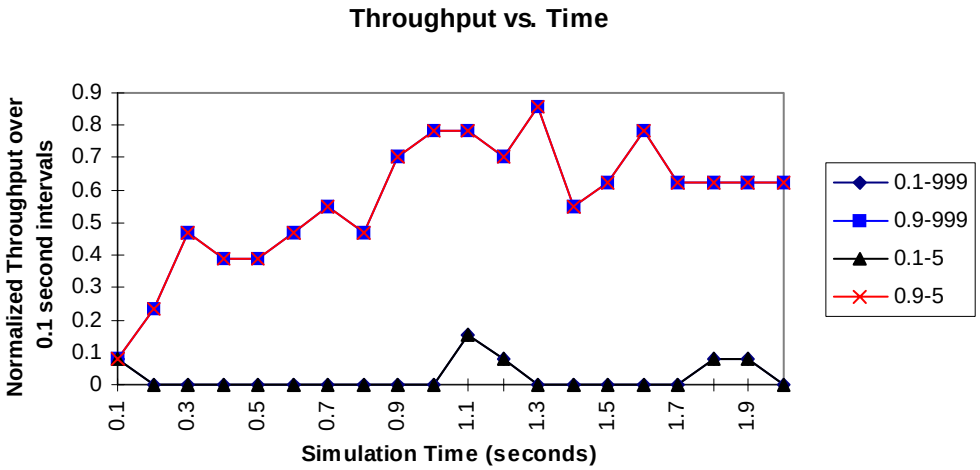

Figure 5-12. Zoomed Version of "Normalized Throughput versus Time over 0.1 second intervals (*Start*=5 and 999 Load=0.1 and 0.9)"

5.4.2 Stop Time

Equally important when determining simulation parameters is when to stop the simulation. Stopping too early may lead to misleading results, and running the simulation too long wastes computing resources. There is a point when a longer simulation duration yields very little improvement in accuracy. In addition, some files generated by the simulation become too large to be practical. Pilot simulations are used to determine a suitable compromise between accuracy as measured via confidence intervals and resources as measured by file sizes and runtimes. It is found that a stop time of 51 seconds provides the best balance between accuracy and computing resources utilization. Since data are not gathered until after the one second startup interval, the 51 second stop time allows 50 seconds of monitored, steady-state simulation.

There are two exceptions to using 51 seconds as the simulation duration. The first is when fairness of the protocols is determined. The second is when the impact of the protocols upon TCP is determined. In both cases, 500 seconds is used as the simulation duration to facilitate a comprehensive, long-term view of network behavior.

5.4.3 CW_{min}

The contention window is used to determine the maximum number of slots a station should backoff after a collision. CW_{min} is the size of the window before a backoff occurs and is set to 31 in accordance with 802.11 [IEE96]. The simulation does not use this parameter directly; instead, it is used in the calculation of CW_{min} *Scale*, which is used in the simulation.

5.4.4 CW_{min} Scale

This parameter is used to calculate contention window sizes for successive backoffs of a data frame. As specified by 802.11, the contention window follows a truncated exponential backoff mechanism with the following increments for the direct sequence physical layer: 31, 63, 127, 255 [IEE96]. The following formula finds the power of 2 that results in CW_{min}-1:

$$CW_{min}\,Scale = \left(\frac{\log(CW_{min} + 1)}{\log 2} \right) - 1 \tag{5-20}$$

This calculated value is used in the *Compute Backoff Time* module to determine the next contention window size.

5.4.5 CW_{max}

This parameter represents the largest size of the contention window and is set to 255 as directed by 802.11 [IEE96]. Once the window grows to 255, it stays there; subsequent backoffs for the same data frame will use 255.

5.4.6 Number of Stations

This parameter represents the number of stations in the *ad hoc* network. The simulation uses 10 stations for all simulations, which is a common network configuration for simulations [CrW96, KoN92, Ryp92, WaN92, WeW95, WoW95].

5.4.7 Retransmission Limit

The *retransmission limit* determines when the MAC protocol stops its attempts to retransmit a frame. If this limit is reached, the protocol declares an "excessive retransmission" condition and signals the upper layer protocols. This limit is set to 15. Since the first draft of 802.11 [IEE94] did not specify this limit, an approximation was made based on the known limit of 16 implemented in IEEE 802.3 (CSMA/CD) [IEE85].

5.4.8 Maximum Queue Size

The parameter determines how many packets may be queued at the MAC-layer first-in-first-out (FIFO) queue awaiting transmission. As currently configured, 10 packets may be queued.

5.4.9 PN Code Length max

This parameter is set to 63 to indicate that when reconfiguration does occur, the PN code consists of 63 chips.

5.4.10 PN Code Length min

When a station is not reconfigured, it transmits using the minimum PN code length of 11 chips.

5.4.11 Chipping Rate

The *chipping rate* is set to 11.264 million-chips-per-second (Mcps) to match the hardware configuration of the wireless LAN testbed.

5.4.12 PLCP Preamble Length

In accordance with 802.11, the physical layer convergence procedure (PLCP) preamble for direct sequence spread spectrum is set to 192 bits [IEE96].

5.4.13 ACK Frame Length

IEEE 802.11 specifies the length of ACK frames to be 112 bits.

5.4.14 Packet Length

Packet Length represents the number of data bits in the packet. This parameter is held constant at 8000 bits (1000 bytes) throughout the simulations. This value corresponds to roughly half of 802.11's maximum of 2312 bytes.

5.4.15 Prop/Detect Delay

This parameter models the delays associated with signal propagation and detection. For the *ad hoc* network configuration chosen, the maximum feasible distance is 300 meters. Using $3x10^8$ meters/second as the speed of light, the time required for a frame to propagate from one station to another is 1 microsecond, which is negligible for the given network. Therefore, this parameter is set to 0.

5.4.16 Slot Time

This parameter is set to 50 microseconds as specified in the first draft of 802.11 and subsequent working papers [IEE94]. Slots are used to facilitate timing within the network including the calculation of backoff times.

5.4.17 SIFS Time

The Short Interframe Space (SIFS) time is defined by 802.11 as the shortest time interval between frames and is used when a station must maintain control of the medium for a period of time [IEE96]. *SIFS Time* is set equal to *Slot Time*.

5.4.18 DIFS Time

The Distributed Coordination Function (DCF) Interframe Space (DIFS) is used to transmit data frames [IEE96]. This parameter is set equal to 3 x *SIFS Time*.

5.4.19 Start Reconfigure Limit

This parameter determines when the adaptive protocol is initiated as previously explained. Setting this parameter to any value above *Retransmission Limit* will disable the adaptive protocol since an excessive retransmissions condition is declared before *Start Reconfigure Limit* is reached.

5.4.20 Reconfigured Transmissions Limit

Reconfigured Transmissions Limit determines how many times a reconfigured frame is sent. In other words, if the first attempt at transmitting the data frame over the reconfigured link is unsuccessful

and this parameter is set to two, the source station will attempt another transmission assuming the medium is free (a carrier is not present). *Reconfigured Transmissions Limit* is set to 2.

5.4.21 Reconfigure Frame Length

This parameter is the length of the frame sent from the source station to attempt a link reconfiguration. It is set to 160 bits.

5.4.22 Reconfigure Attempts Limit

Reconfigure Attempts Limit is a parameter used during initial development of the adaptive protocol and has since been deemed unnecessary. However, it was not removed from the overall simulation in order to maintain a certain baseline configuration. Thus, this parameter should always be set to 1.

5.4.23 Reconfigure ACK Frame Length

This parameter is the length of the reconfigure ACK frame sent from the destination station to acknowledge a reconfiguration attempt by the source station. It is set to 112 bits.

5.4.24 Reconfigure ACK Timeout Period

This parameter determines how long the source station is to wait before declaring the reconfigure frame a failure. The period includes the time to transmit the reconfigure ACK frame back to the source station using the slower data rate (use longer PN code length) plus one *SIFS Time*. Also included in the period calculation is a small time buffer of *SIFS Time*. *Reconfigure ACK Timeout Period* is set using the following:

$$(2 * SIFS\ \ Time) + \frac{Reconfigure\ ACK\ \ Frame\ Length + PLCP\ Preamble\ Length}{\left(\frac{Chipping\ Rate}{PN\ Code\ Length\ max}\right)} \qquad (5\text{-}21)$$

5.4.25 Additional Frame During Reconfigure Timeout Period

This parameter determines when a destination station should consider the reconfiguration a failure as previously explained. *Additional Frame During Reconfigure Timeout Period* is calculated using the following:

$$(2 * SIFS\ Time) + \frac{\beta}{\left(\frac{Chipping\ Rate}{PN\ Code\ Length\ max}\right)} \qquad (5\text{-}22)$$

where

$\beta = Packet\ Length + Number\ of\ TCP/IP\ and\ MAC\ Ovehead\ Bits + PLCP\ Preamble\ Length$

5.4.26 Max Additional Frames to Transmit During Reconfigure

This parameter determines how many additional frames to transmit after the original data frame during a reconfiguration period.

5.4.27 BER for Max PN Code Length

This parameter is the BER used during periods when the maximum PN code length is used. It is set to 10^{-5}.

5.4.28 Data Not Received Timeout Period

This parameter also determines when a destination station should consider the reconfiguration a failure as previously explained. *Data Not Received Timeout Period* is calculated using the following:

$$(2 * SIFS\ Time) + Reconfigured\ Transmissions\ Limit * \left[\frac{\beta}{\rho} + (2 * SIFS\ Time\right] \qquad (5\text{-}23)$$

where

$\beta = Packet\ Length + Number\ of\ TCP/IP\ and\ MAC\ Overhead\ Bits +$
$Reconfigure\ ACK\ Frame\ Length + (2*PLCP\ Preamble\ Length)$

and

$$\rho = \frac{Chipping\ Rate}{PN\ Code\ Length\ max}$$

5.4.29 Best Service Time

This parameter defines the best possible time that a frame can be serviced by the MAC. It includes the time required to send the frame and wait for the acknowledgment. *Best Service Time* implements the following:

$$DIFS\ \ Time + SIFS\ Time + \left(\frac{\beta}{\rho}\right) \qquad (5\text{-}24)$$

where

$\beta = Packet\ Length + Number\ of\ TCP/IP\ and\ MAC\ Overhead\ Bits + ACK\ Frame\ Length +$
$(2 * PLCP\ Preamble\ Length)$

and

$$\rho = \frac{Chipping\ Rate}{PN\ Code\ Length\ min}$$

5.4.30 Number of TCP/IP and MAC Overhead Bits

This parameter is used to model the addition of overhead bits from TCP, IP and the 802.11 MAC as the packet flows down the OSI stack. TCP and IP add 320 bits of overhead, and 802.11 adds another 272 bits to make the total number of overhead bits equal to 592.

5.4.31 Mean Interarrival Time

The rate at which packets are generated by a station is determined by this parameter. In other words, system load is implemented using the Mean Interarrival Time. It is defined as

$$\frac{(\textit{Best Service Time}*\textit{Number of Stations})}{\textit{Load}} \tag{5-25}$$

5.4.32 Global Seed

The global seed is used by all random number generators within the simulation. Designer™ is designed to use this seed along with the instance number of each generator to form a reliable sequence of random numbers. MacDougall [Mac92] points out that running a simulation with just one set of seeds gives results valid for that specific set. He recommends using at least 5 to 10 sets to decrease the confidence intervals. This research effort uses 5 global seeds. Therefore, every simulation is run five times (once for each seed), and the simulation results are averaged over the five.

5.5 Factors

Factors are the parameters that are varied during an evaluation such that they significantly impact system performance when altered [Jai91]. A minimum set of factors is identified as fitting this criteria. The factors include the BER (when not reconfigured), offered system load, *Start*, and *Max*. These are listed in Table 5-6 along with levels for each factor.

Table 5-6. Simulation Factors

Factor	Levels
BER	0.00001, 0.00002, 0.0001, 0.0002, 0.001, 0.002, 0.01
System Load	0.1, 0.3, 0.5, 0.7, 0.9
Start	3, 4, 5, 6, 7, 8, 9, 999 (disable reconfigure protocol)
Max	0, 2, 4, 6, 8, 10

5.5.1 Bit Error Rate (BER)

Commonly used as a figure of merit for digital links, the BER is the probability that a bit is received in error at the destination [PrB86]. The dynamic characteristics of a wireless link necessitates an evaluation using several BERs. Numerous simulations are run to determine network performance over a wide range of BERs.

5.5.1.1 Static BER

Two means of introducing bit errors are allowed in the simulation—static BER and dynamic BER. *Static BER* means the BER does not change during a simulation. For example, setting *Static BER* to 0.0001 causes all frames traversing the network to experience 1 bit error in every 10,000 bits based on a uniform distribution. Any value less than one will cause this BER to be applied to the entire network. If *Static BER* is set to 1, the error-generation mechanism just discussed is disabled, and the individual bit errors are read from the ASCII error file as explained in the following paragraph. Regardless of the method (static or dynamic) of simulating bit errors, when the station enters the reconfigured state, all frames experience the conservative lower bound estimate of 10^{-5} mentioned earlier.

5.5.1.2 Channel Control Matrix File

This file is used to read bit errors between two stations and when they occurred. The file is formatted such that the first row contains the total number of entries in the file. The remaining rows contain the actual errors with the time the error occurred first followed by the source station number and then the destination station number. The following is an example of a properly formatted file.

```
6
0.001   4       5
0.083   8       9
```

This example generates two bit errors within the simulation; the first occurs at 0.001 seconds between stations 4 and 5. The second error occurs at 0.083 seconds between stations 8 and 9. The 6 in the first row is the total number of entries in the file and is required by the simulation.

5.5.2 Offered System Load

The number of stations contending for the channel and their packet transmission rate dictate system load. When several stations attempt to transmit packets, the channel becomes more utilized and may actually saturate. This is characteristic of a heavily loaded system. Conversely, few stations contending for the channel infrequently constitutes a light load. Instead of nebulous terms such as heavy and light, load is typically expressed as a percentage of the maximum capacity of the channel. Most researchers use 10 percent (0.1) to 90 percent (0.9) as the lower and upper bounds. The number of steps between these bounds varies, but 0.1 (i.e., steps of 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9) or 0.2 (i.e., steps of 0.1, 0.3, 0.5, 0.7, and 0.9) are often seen. This investigation uses the 0.2 increment to conserve simulation resources with little loss in accuracy.

5.5.3 Start

Start controls when a link between two stations should be reconfigured and is measured in the number of times a frame fails transmission. For example, if it is set to two, two failed attempts are permitted before the source station initiates the reconfigure protocol. Therefore, the next transmission attempt of the failed frame will initiate our protocol. When *Start* is set to 999, our adaptive protocol never initiates since attempts halt due to excessive retransmissions before 999 failed attempts.

5.5.4 Max

Max determines an upper limit on the number of additional frames to send over a reconfigured link. The actual number of additional frames to send over the reconfigured link is also bounded by the number of frames awaiting transmission to the reconfigured destination. That is, *Max* equal to 6 does not guarantee 6 additional frames are transmitted when only 2 frames are awaiting transmission.

5.6 Response Variable

The simulation variable of interest is aggregate throughput of the system. A simulation probe is used to measure this throughput for each simulation run. The objective is to maximize the throughput over all factor levels.

5.7 Simulation Platform

Designer™ runs on a Sun workstation. All simulations are run on the Sun workstations found in the Virginia Tech Information Systems Center (VISC). The majority of the simulations are run on a Sun UltraSparc 1 with 256 MB of RAM and a 170 MHz processor.

5.8 Model Verification

Jain [Jai91] states that verification is the process to determine if the model is correctly implemented. Often called debugging, it is concerned with whether the model does what it is intended to do.

A large portion of the verification process is automatically performed by the Designer™ tool. Built-in blocks have been verified by the tool's designer. However, the interaction of the blocks must still be verified.

Model verification is performed by running numerous interactive simulations and noting the behavior of the system as frames traverse the network. Probes are used to collect data and subsequently plot them to confirm there are no discontinuities in the plot. The model behaves in accordance with the proposed 802.11 standard.

5.9 Model Validation

Proving the simulation model is representative of a real system is validation [Jai91]. It addresses the validity of the operating assumptions, input parameter values, and output values [Jai91]. Three validation techniques exists: expert intuition, real system measurement, and theoretical results. These are discussed below.

The operating assumptions and parameter values are presented earlier in this chapter and follow guidelines set forth by 802.11 and sound engineering practice. The output values are validated using all three techniques discussed above. Simulation results are compared with what seems reasonable using expert intuition. For example, as the BER increases, throughput should decrease and, in fact, ultimately reach zero.

The simulation parameters are altered to make the 802.11 MAC act very similar to a 1-persistent CSMA access scheme. This comparison is shown in Figure 5-13. As can be seen, the 802.11 model behaves comparably with theoretical 1-persistent CSMA.

We also validated our simulation model by comparing our nominal 802.11 throughput with other published results [WeW95, WoW95]. In all cases, our simulation results matched the other models.

Finally, the most comprehensive validation occurs when simulation results are compared against emulation results in Chapter 7.

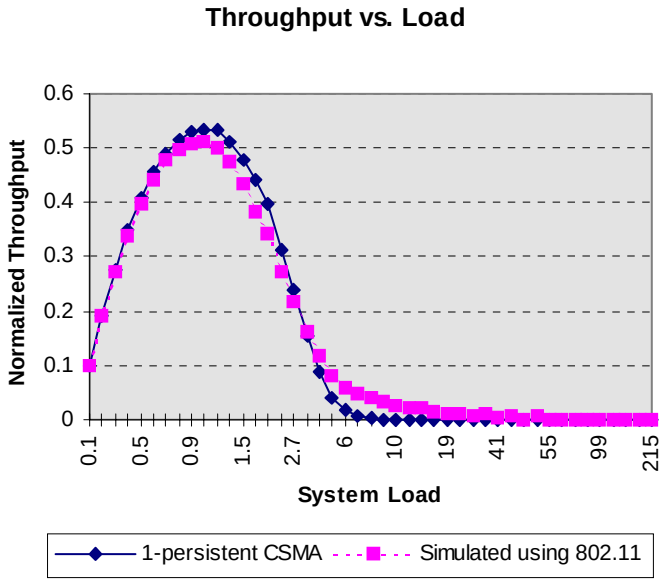

Figure 5-13. Validation of the Simulation Model

5.10 Summary

This chapter presented the simulation environment used to attain the objectives listed in Chapter 3. This chapter discussed the simulation of a wireless LAN using native IEEE 802.11 and CATER as the underlying MAC protocols. Designer™ was identified as the simulation tool used to construct, simulate and analyze the two protocols. It was shown that Designer™ used the propagation of data structures to mimic network traffic; these data structures were presented and explained in this chapter. The actual wireless LAN simulation model was then presented and explained in detail. The upper modules of the model's hierarchy were then presented along with an explanation of their general operation. Gilbert's two-state Markov chain model was then introduced as the mechanism used to simulate bursty errors on a wireless channel. Since reliable bit error characteristics were not available for the channel in question, a sensitivity analysis was performed over the three Gilbert model parameters. The TCP model was then presented. Round trip times were collected for each successful frame and later analyzed using a spreadsheet designed to duplicate the timeout mechanism within TCP with the intent of determining how many times TCP would timeout. Simulation parameters were discussed followed by system factors. Aggregate throughput was then identified as the response variable to be maximized. Model verification and validation efforts were then discussed.

Chapter 6. Simulation Results

A little inaccuracy sometimes saves a ton of explanation.
—H.H. Munro (Saki) (1870-1916)

The chapter describes the results obtained during simulation evaluations. Divided into four main sections, the chapter presents a comparison between the proposed IEEE 802.11 and CATER. Specifically, Sections 6.1 and 6.2 discuss how the response variable (aggregate throughput) is affected by both protocols in the presence of static and dynamic (Gilbert) BERs, respectively. Section 6.3 discusses the fairness issue amongst competing stations. Finally, Section 6.4 presents results pertaining to how TCP is affected by the protocols followed by a summary in Section 6.5.

6.1 Aggregate Throughput with Static BER

Defined as the rate at which frames from all stations are successfully serviced, aggregate throughput is a common metric for measuring system performance for a LAN. This effort attempts to maximize aggregate throughput across all four factors. To this end, a full factorial experiment with replications is chosen.

Designer™ uses a global seed along with the instance number of each random number generator to form a reliable sequence of random numbers. At least five different global seeds are required to obtain suitable confidence intervals for the aggregate throughput [Mac92]. This is analogous to replications as defined by Jain [Jai91]. Therefore, every simulation run is replicated five times (once for each seed), and the aggregate throughputs obtained for the five runs are averaged.

A comprehensive analysis of all factor interactions requires all combinations of factors with one exception—when the reconfigure protocol is disabled (*Start* = 999), *Max* is meaningless. Therefore, the following simulation runs are made: all levels of BER (7), all levels of system load (5), all levels of *Max* (6), the first 7 levels (i.e., *Start* = 3 through 9) of *Start* (7), and 5 global seeds. This yields 7,350 simulation runs.

In addition, simulations are run with the reconfigure protocol disabled. This requires 7 levels of BER, 5 levels of system loads, and 5 levels of global seeds, resulting in another 175 simulation runs. Therefore, a total of 7,525 simulation runs are made. This took approximately 250 CPU hours on a lightly loaded Sun UltraSparc 1 with 256 MB of RAM and a 170 MHz processor running Solaris 2.5.

CATER offers interesting and noteworthy performance improvements over native 802.11. When compared with the proposed 802.11 protocol, CATER offers substantial improvements in throughput during periods of high BERs while operating with minimum degradation during low BERs. This degradation is attributed to protocol overhead, which consists of transmissions of management frames (e.g., reconfigure request frames) and decreased data rate for a reconfigured link due to a longer PN code. The protocol objective is to reconfigure a link between two stations only when frames are lost due to bit errors and not collisions. Thus, reconfiguring a link should be attempted only after a "sufficient" number of data frame retransmissions. Since collisions also cause lost frames, an early reconfiguration caused by collisions instead of bit errors results in more network traffic. This occurs at a time when the network is already congested further exacerbating the situation. Therefore, CATER should not initiate reconfiguration until several attempts are made to resolve collisions using 802.11's backoff mechanism. However, attempting too many retransmissions wastes channel bandwidth. A balance is required to maximize throughput. Similarly, a balance is essential for the number of additional frames sent over a reconfigured link. To illustrate this behavior, simulation results for aggregate throughput of the proposed 802.11 protocol followed by the results of CATER are presented.

6.1.1 Standard IEEE 802.11

Figure 6-1 shows the simulation results for 802.11. The two independent variables are BER and offered system load. The dependent variable, normalized aggregate system throughput, is plotted on the vertical axis. Throughput is normalized to the maximum data rate corresponding to the default PN code of 11 chips (1.024 Mbps). As could be expected, throughput increases with increasing load and decreasing BER. The graph illustrates the failure of 802.11 when confronted with high BERs—no frames

are able to traverse the channel. No amount of retransmissions successfully send the frames, and all frames experience excessive retransmissions!

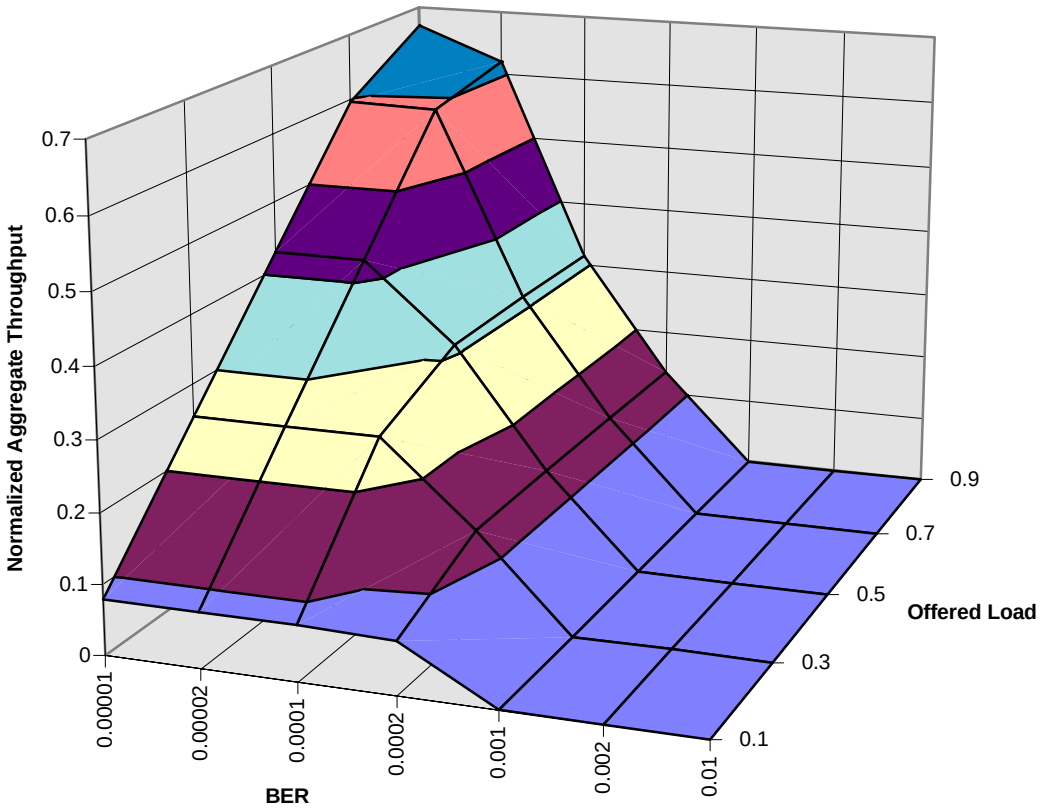

Figure 6-1. Aggregate Throughput with the Reconfigure Protocol Disabled (Standard 802.11)

6.1.2 CATER

Figure 6-2 through Figure 6-5 illustrate CATER's performance over various values of *Start* and *Max*. Figure 6-2 shows protocol performance with *Start* set to 3 and *Max* set to 0—meaning the protocol will initiate after 3 failed frames and will only send one frame after the link is reconfigured. The interesting region of this graph is when the BER is high (BER ≥ 0.0002). Notice the throughput does not go to 0 as in the 802.11 protocol (See Figure 6-1). If *Start* is held at 3 and *Max* is now changed to 6, CATER exhibits the behavior shown in Figure 6-3. Throughput increases at high BERs. However, at low

BERs this configuration results in a discernible decrease in throughput due to protocol overhead mentioned earlier—reconfiguring the link too early and wasting bandwidth by tying the channel up with additional frames sent during reconfiguration. As the two reconfigured stations occupy the channel, other stations are holding frames for transmission in backoff. The longer the channel is occupied, the more likely another collision will occur when the two reconfigured stations relinquish control.

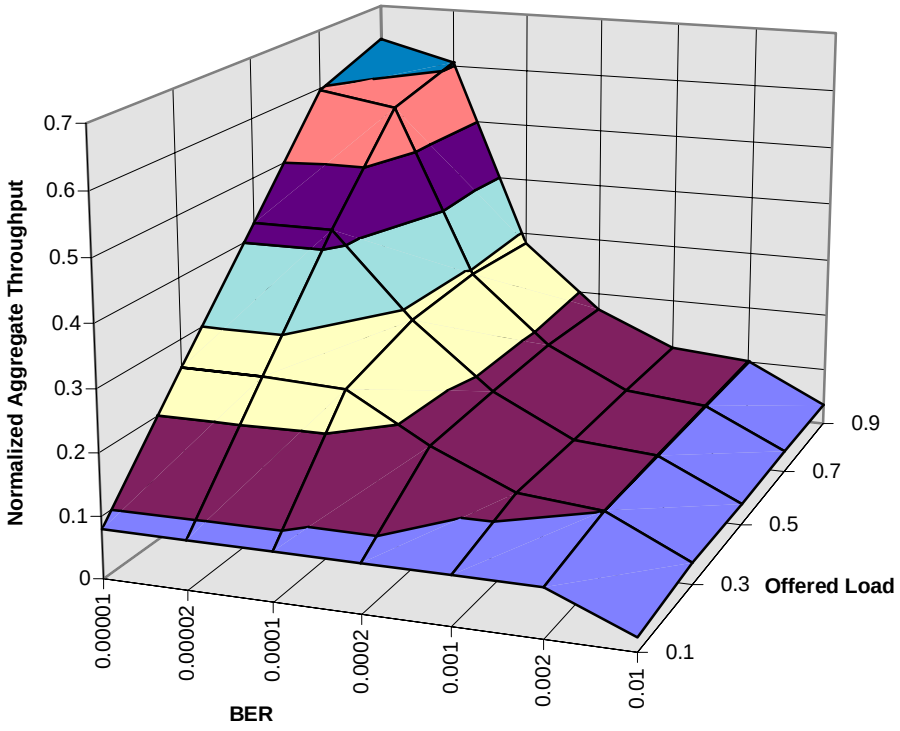

Figure 6-2. Aggregate Throughput when *Start* = 3 and *Max* = 0

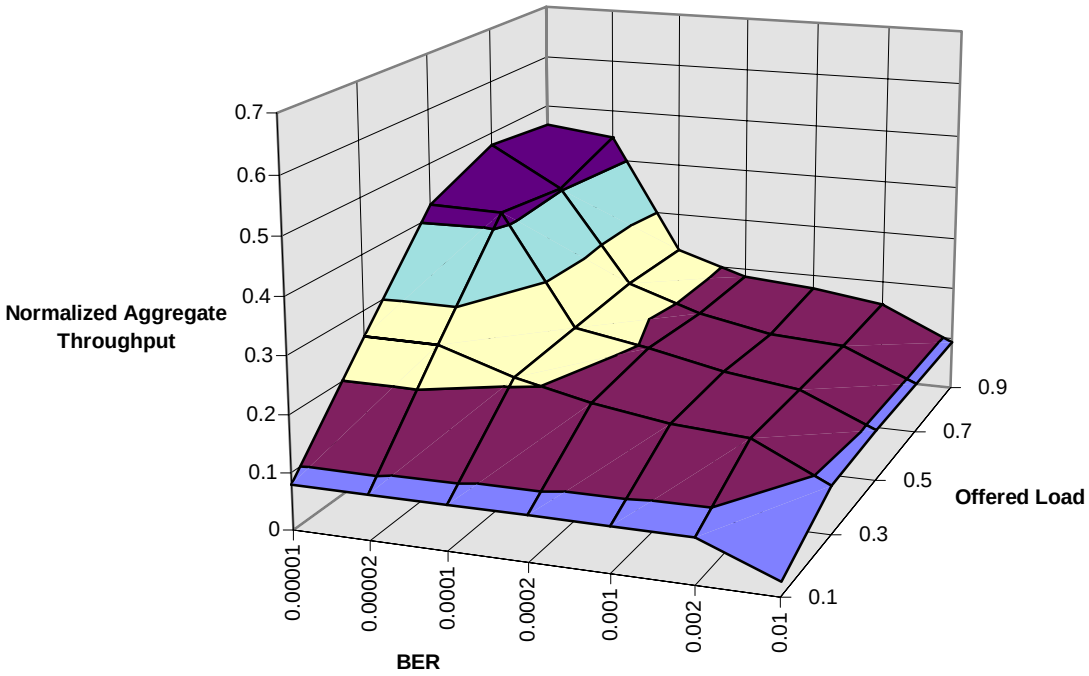

Figure 6-3. Aggregate Throughput when *Start* = 3 and *Max* = 6

Now consider the case when *Start* is set to 9 and *Max* is set to 0. Figure 6-4 shows this configuration's performance is similar to *Start* = 3 and *Max* = 0 with slightly lower throughput at high BERs. Figure 6-5 illustrates the case when *Max* is set to 6. Notice how throughput once again increases at high BERs. The throughput, however, does not significantly decrease for low BERs. In this configuration, 9 failed transmission attempts are required before our protocol is initiated—drastically reducing the probability of reconfiguring a link too early. Optimum values for *Start* and *Max* are found in subsequent sections of this chapter. Appendix B contains a compendium of all surface plots.

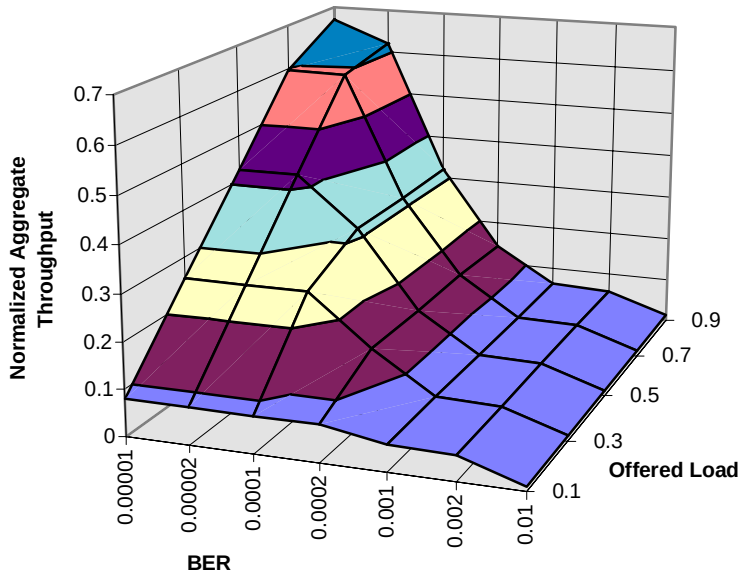

Figure 6-4. Aggregate Throughput when *Start* = 9 and *Max* = 0

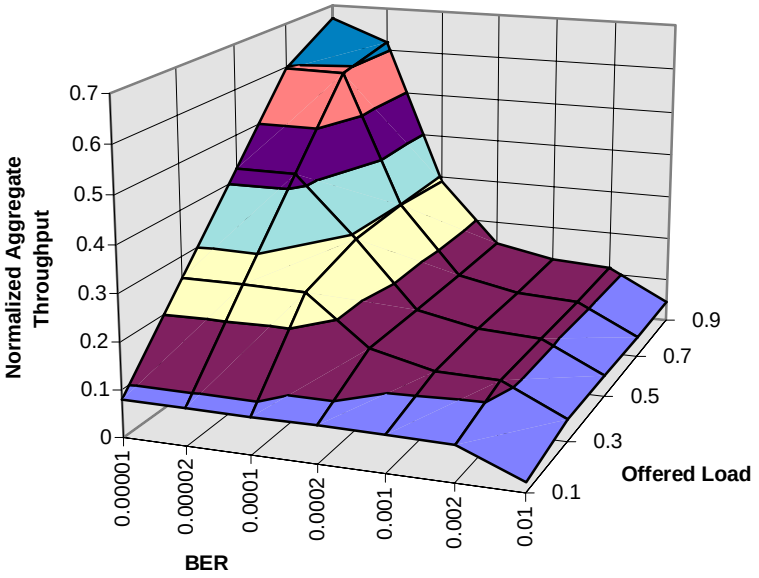

Figure 6-5. Aggregate Throughput when *Start* = 9 and *Max* = 6

6.1.3 Confidence Intervals

Instead of showing the confidence intervals for all 1,470 surface data points (7 levels of BER, 5 levels of Load, 7 levels of *Start*, and 6 levels of *Max*) when CATER is used, the average relative confidence interval half width is computed for each combination of *Start* and *Max*. The same metric is calculated for the 35 (7 levels of BER and 5 levels of Load) data points for native 802.11 as well.

Average relative confidence interval half width is derived by finding the mean, standard deviation, and 90 percent confidence interval half width for all data points on the surface plots. As noted earlier, each data point is an average of the values obtained from the simulation using 5 different random seeds (5 replications). Specifically, setting n equal to 5 for the 5 replications, the following formula is used to determine the mean value for data points on the surface plots:

$$\text{Mean Surface Plot Value} = \frac{1}{n}\sum_{Seed=1}^{n} x_{Seed} \ . \tag{6-26}$$

Standard deviation is calculated using

$$\text{Standard Deviation} = \sqrt{\frac{n\sum x^2 - \left(\sum x\right)^2}{n(n-1)}} \ . \tag{6-27}$$

The 90 percent confidence interval half width (H) is calculated from the following equation [Mac92]:

$$\text{H} = 2.132 * \left(\frac{\text{Standard Deviation}}{\sqrt{n}} \right), \tag{6-28}$$

where 2.132 is the value of the *t* distribution for a confidence level of 0.90 with n-1 observations. Once the half width is found, the relative confidence interval half width is calculated using

$$\text{Relative H} = \frac{H}{Mean} * 100 . \tag{6-29}$$

This relative half width provides a measure of accuracy of the estimate of the mean and is derived for all surface data points. Table 6-1 is an example of relative H when *Start* is set to 3 and *Max* is set to 0.

The average relative confidence interval half width is found by averaging all values in Table 6-1. This represents a figure of merit for the goodness of the data when *Start* is 3 and *Max* is 0. The remaining relative half width tables are found in Appendix C. The averaged relative half widths are shown in Table 6-2. The low values for the relative half widths indicate the means derived from the 5 replications closely approximate the true mean. In addition, this is a good indication that the simulation duration for each run is sufficient.

Table 6-1. Relative Confidence Interval Half Width for *Start*=3 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.759136	2.814589	2.874153	2.757621	4.980026
0.3	3.859965	3.859965	2.482929	4.702612	2.619931	0.649541	1.904545
0.5	2.340103	2.365847	3.948412	3.816846	1.731569	1.257758	2.847065
0.7	2.118679	1.969644	2.725683	3.198647	1.514357	1.548119	3.103756
0.9	2.147969	2.336103	4.555127	3.928625	1.510161	1.349706	4.000359

Table 6-2. Average Relative Confidence Interval Half Width

	Start						
Max	3	4	5	6	7	8	9
0	2.6	2.8	2.7	2.8	3.3	3.2	3.2
2	4.5	3.7	3.5	3.5	3.4	3.7	3.7
4	4.9	4.3	4.0	3.9	3.8	3.9	3.9
6	5.9	4.6	4.4	3.7	3.8	3.7	3.8
8	6.1	5.0	4.6	3.5	3.8	3.7	3.7
10	6.4	4.8	4.1	4.0	4.1	3.7	3.6
IEEE 802.11 average relative half width = 2.6							

6.1.4 Analysis of Variance

A four-factor analysis of variance (ANOVA) is used to investigate the statistical significance of the four factors and their interactions as they pertain to aggregate throughput. The goal is to determine which factors contribute to the variance of the aggregate throughput in a significant way. An ANOVA is applied to a subset of the simulation data gathered using the data analysis software package called SAS® [SAS85a]. (A subset of the data is used to keep the SAS® analysis tractable both from the standpoint of volume of data required by the analysis and execution time.) Table 6-3 details the factors involved as well as their values used during the ANOVA analysis. Table 6-4 shows the results of the ANOVA.

Of the values shown in Table 6-4, R-Square provides the best indication of how well the model fits the simulation data. This parameter is a measure of how much variation in aggregate throughput can be accounted for by the system model [SAS85b]. Ranging in value from 0 to 1, R-Square indicates the best match when it is closest to 1. A high degree of confidence is indicated by the R-Square value of 0.997 as shown in Table 6-4. In addition, it is seen that the model is statistically significant by observing

the low value for the significance probability (Pr > F). Now that the overall model is known to be good, the statistical significance of each factor is determined.

Table 6-3. ANOVA Factor Levels

Factor	Factor Levels
Start	3, 5, 7, 9
Max	0, 2, 4, 6
BER	0.00001, 0.00002, 0.0001, 0.0002, 0.001, 0.002, 0.01
Load	0.1, 0.3, 0.5, 0.7, 0.9

Table 6-4. ANOVA Results

Source	Degrees of Freedom	Sum of Squares	Mean Square
Model	559	96.266337	0.172212
Error	2240	0.219429	0.000098
Corrected Total	2799	96.485767	
Model F Value = 1757.99		Pr > F = 0.0001	

R-Square	Coefficient of Variance	Root MSE	Aggregate Throughput Mean
0.997726	4.777553	0.0099	0.2072

Statistical significance is easily found by examining the F Value and significance probability (Pr > F) generated by the ANOVA analysis. The ANOVA output for each factor and interactions are shown in Table 6-5. The F Value is an indication of how much a factor's variance or factor interactions affect aggregate throughput. As such, a larger F Value (much greater than 1) indicates the factor (or interaction of factors) is assumed to explain a significant fraction of the variance [Jai91]. If the F Value is less than 1, the mean square of the factor is less than the model's mean square error; thus, experimental error contributes more to the variance than does the factor, and the factor is not statistically significant. Inspecting Table 6-5, one finds that all factors and factor interactions are statistically significant. As expected, Load and BER significantly affect aggregate throughput variance. It is also seen that S*tart* and *Max* are statistically significant!

Table 6-5. ANOVA Results for Factor Significance

Source	DF	SS	Mean Square	F Value	Pr > F
Start	3	0.012820	0.004273	43.62	0.0001
Max	3	0.026228	0.008743	89.25	0.0001
Start * *Max*	9	0.037281	0.004142	42.29	0.0001
BER	6	53.119576	8.853263	90376.71	0.0001
Start * BER	18	0.723137	0.040174	410.11	0.0001
Max * BER	18	0.444092	0.024672	251.86	0.0001
Start * *Max* * BER	54	0.095171	0.001762	17.99	0.0001
Load	4	18.090277	4.522569	46167.72	0.0001
Start * Load	12	0.149764	0.012427	127.40	0.0001
Max * Load	12	0.029128	0.002427	24.78	0.0001
Start * *Max* * Load	36	0.047169	0.001310	13.38	0.0001
BER * Load	24	22.624365	0.942682	9623.17	0.0001
Start * BER * Load	72	0.562610	0.007814	79.77	0.0001
Max * BER * Load	72	0.148902	0.002068	21.11	0.0001
Start * *Max* * BER * Load	216	0.155818	0.000721	7.36	0.0001

6.1.5 Comparison Analysis

In an attempt to quantify the impact *Start* and *Max* have on aggregate throughput, another analysis technique is used. In addition, the optimum settings for these two factors are sought.

Simulations for all 42 combinations of *Start* and *Max* are compared against the simulation of the 802.11 protocol. The nominal 802.11 values shown in Figure 6-1 are subtracted from the data in each of Figure 6-2 through Figure 6-5 and an average is taken of these differences. For example, each data point in Figure 6-1 is subtracted from the corresponding data point in Figure 6-2 resulting in a difference surface plot as shown in Figure 6-6. The objective is to obtain a large positive difference indicating CATER is performing better than the standard 802.11 protocol. As can be seen in Figure 6-6, CATER performs better at higher BERs, while 802.11 performs slightly better at lower BERs. This is attributed to the added overhead of the CATER protocol; specifically, the adaptive protocol is initiated when perhaps bit errors were occurring due to collisions instead of interference. Examining the difference plots in Appendix D reveals CATER's behavior over different values of *Start* and *Max*. In general, as *Start* and *Max* increases, the overhead for low BERs decreases.

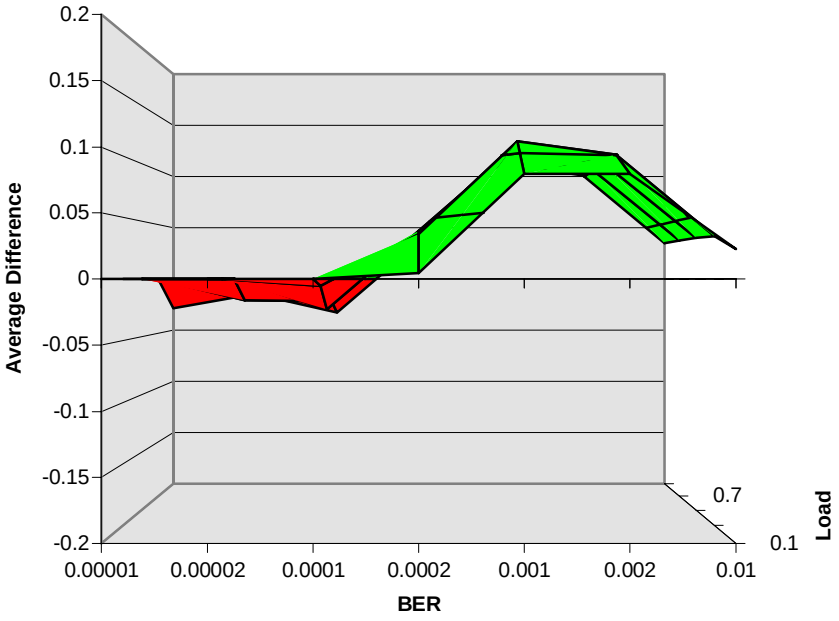

Figure 6-6. Difference Between 802.11 and CATER with *Start*=3 and *Max* =0

Given the wide range of *Start* and *Max*, a selection must be made for the wireless LAN. In an attempt to find these optimum values, a single relative improvement metric is derived by averaging the data points on each difference plot (i.e., all 35 surface data points across the 7 BERs and 5 loads). This metric is called Average Difference and is plotted in Figure 6-7 for all values of *Start* and *Max*.

6.1.6 Optimizing CATER (Selecting Values for *Start* and *Max*)

Selecting the proper values for *Start* and *Max* requires an understanding of the time involved in a reconfigure attempt as well as the time spent in reconfigure. A larger value of *Start* increases the time before a link is reconfigured thereby increasing the time delay seen by TCP. The same is true of *Max*; a larger value allows more frames to be sent during the reconfigured period once again increasing the delay experience by TCP. A balance is required to maximize aggregate throughput, minimize the impact of delay on TCP and prevent one station from monopolizing the channel for an extended period of time (*Max* set too high). Therefore, even though the graph indicates larger differences (improved performance) for values of *Start* greater than 5 and *Max* greater than 6, these two values are selected for this network configuration over what, on the surface, appear to be better combinations with little impact on Average

Difference. In fact, there is only a three percent increase in Average Difference if *Start* is set to 6 and *Max* is set to 10, which corresponds to the maximum data point on the graph; however, the delay experienced by TCP increases by approximately 60 percent over setting *Start* to 5 and *Max* to 6.

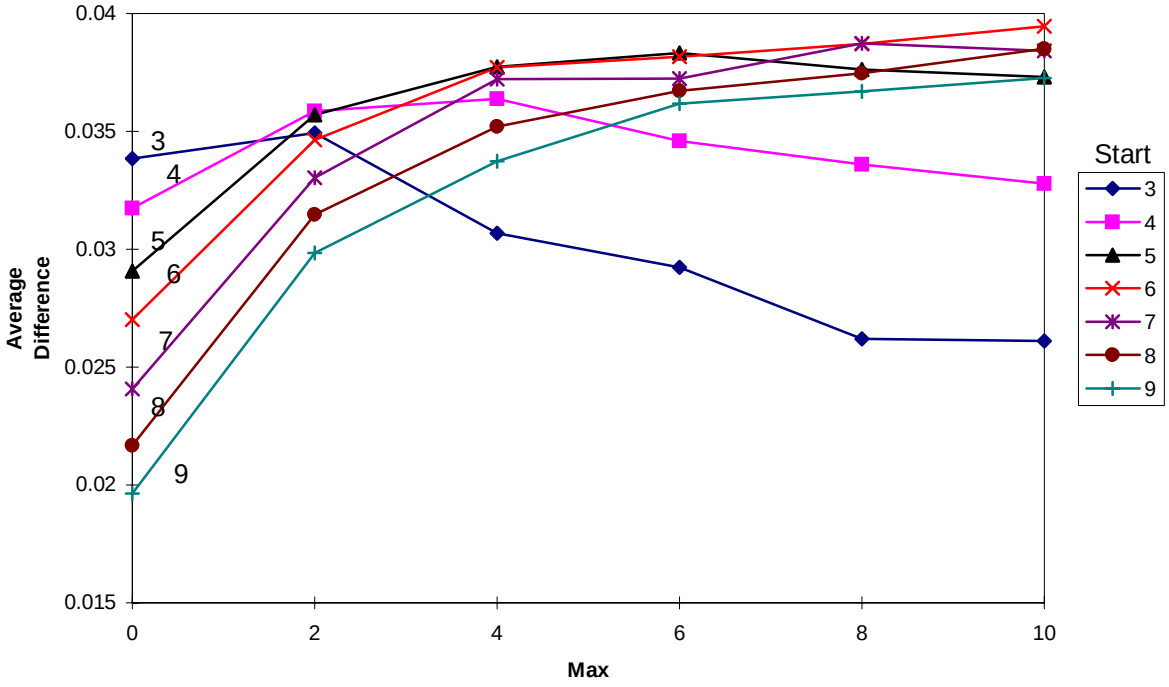

Figure 6-7. Average Difference for Values of *Start* and *Max*

Figure 6-8 shows system performance when *Start* is set to 5 and *Max* is set to 6. A comparison of 802.11 against CATER at these factor levels shows that throughput increases by 273 percent when operating at high BERs (between 0.0002 and 0.01 inclusive) and decreases by 6.6 percent when operating at low BERs (between 0.00001 and 0.0001 inclusive). The most notable region is at high BERs where standard 802.11 fails to transmit frames yet CATER still enables communication over such a degraded channel. In fact, instead of no throughput using 802.11, CATER supports up to 14 percent aggregate throughput—a substantial improvement for BERs up to 10^{-3}.

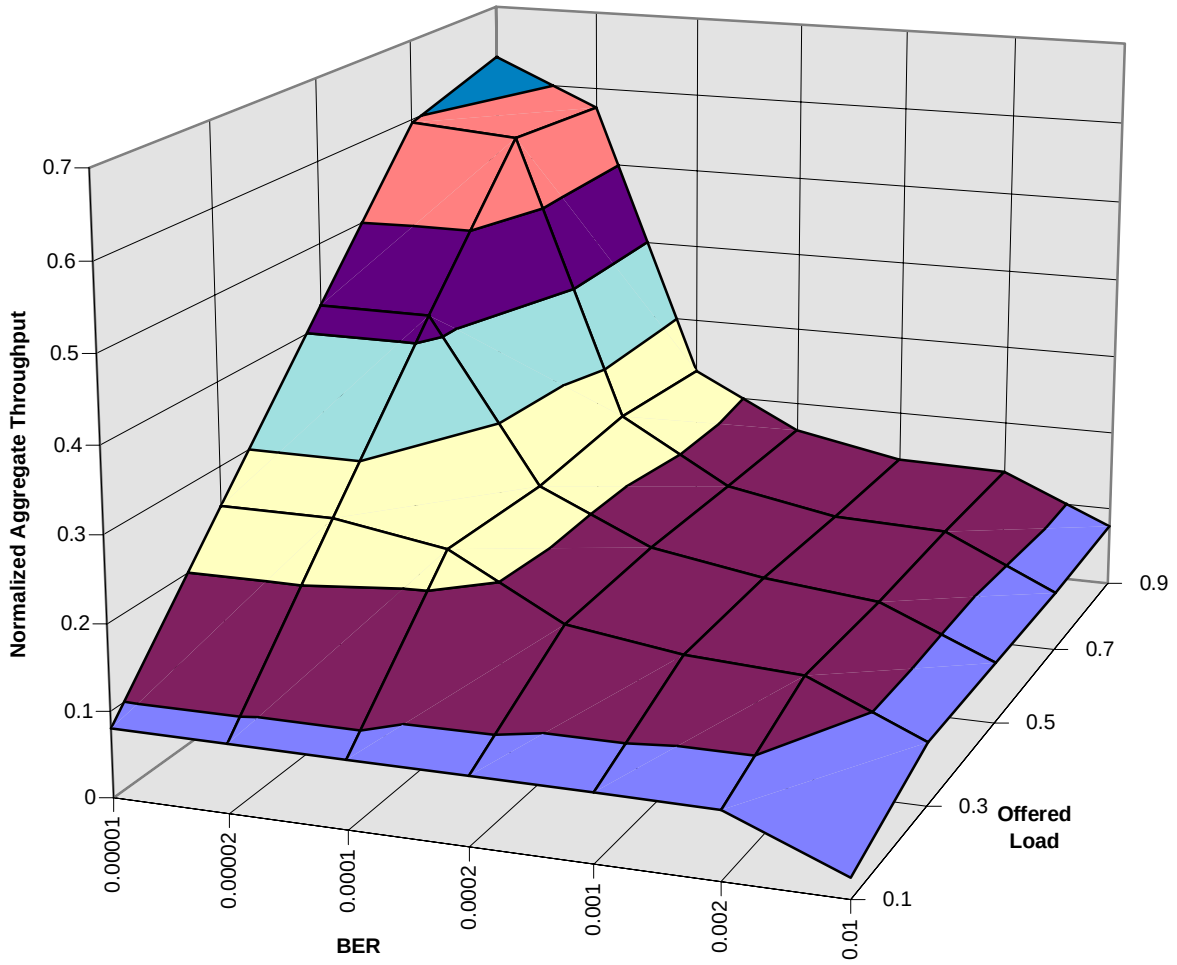

Figure 6-8. Aggregate Throughput when *Start* = 5 and *Max* = 6

A critical feature of CATER is the low overhead found at low BERs. By reconfiguring the link only after experiencing "*Start*" failed frames, overhead is significantly reduced versus reconfiguring the link immediately. As noted above, using this network configuration, CATER experiences a slight 6.6 percent decrease in aggregate throughput for low BERs. A station operating within a WLAN implementing CATER will perform at approximately the same level as an 802.11 network. However, when the station experiences a degraded link due to fading, multipath, or other disruptive phenomena, CATER significantly enhances overall network performance by transforming the corrupt link into a viable one.

6.2 Aggregate Throughput with Dynamic BER (Gilbert Model)

This section presents the results of simulations run using a dynamic BER. The BER is controlled by the bit error file discussed in Chapter 5 which varies bit errors as simulation time progresses. A total of 600 simulations are run comparing 802.11 and CATER. The twelve error files discussed in Chapter 5 are used as the source of bit errors. Load is varied from 0.1 to 0.9 in 0.2 increments. Rather than run the 12 error files against all possible combinations of *Start* and *Max* as was done for the Static BER simulations, the aforementioned optimum setting are used. Thus, a comparison is made between native 802.11 (*Start* set to 999) and CATER (*Start* set to 5 and *Max* set to 6). The global seed is assigned 5 different values. As can be seen from Figure 6-9, the aggregate throughputs for 802.11 and CATER are plotted against Load. Rather than show all simulation results here, the three most significant plots are shown. Appendix E contains all simulation plots. These three plots shown are representative samples of all plots; in other words, all plots not shown in this chapter have the same characteristics as the ones that are shown. Figure 6-9, Figure 6-10, and Figure 6-11 demonstrate simulation behavior using various values for P, p, and 1-*h*.

6.2.1 Protocol Comparison

Figure 6-9 illustrates the comparison of the two protocols when the channel is relatively free of bit errors. It is seen that both perform the same until the system becomes heavily loaded. When this occurs, more bit errors start occurring due to collisions which causes CATER to initiate. As seen in Table 6-6, a slight 2 percent drop is observed at a load of 0.9 which is attributed to the added overhead associated with CATER.

Figure 6-10 illustrates the comparison when the channel is experiencing a moderate amount of bit errors. In this particular situation, CATER does not perform as well as standard 802.11. Table 6-7 shows this comparison.

Figure 6-11 shows the comparison of the two protocols for a degraded channel. CATER performs quite well even though 802.11 is crippled by the channel. In other words, 802.11 only allows a few frames to pass which effectively renders the link useless since TCP is constantly timing out and retransmitting. Table 6-8 highlights CATER's improvements.

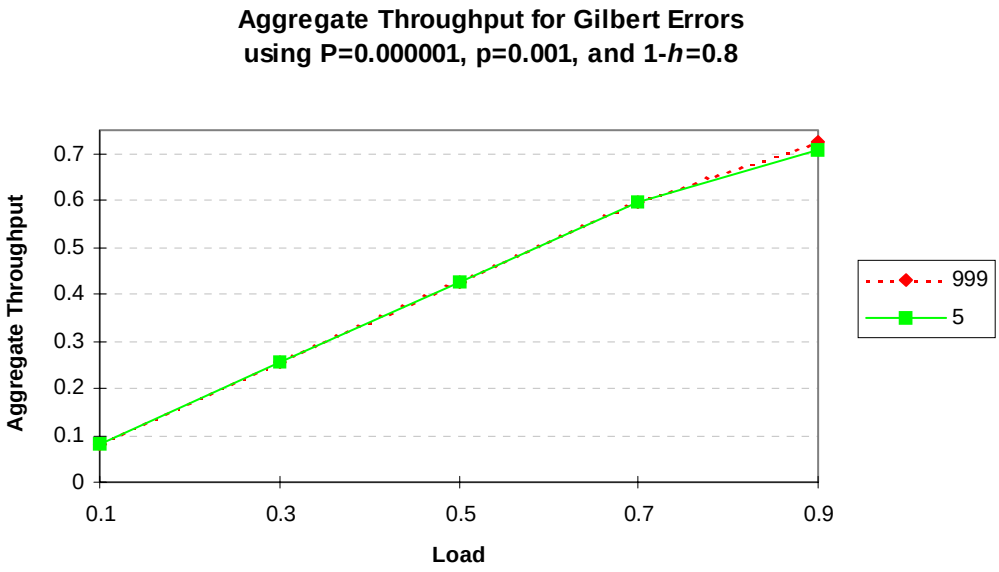

Figure 6-9. Aggregate Throughput for Gilbert Errors (P=0.000001, p=0.001, and 1- *h*=0.8)

Table 6-6. Comparison of Aggregate Throughput for 802.11 and CATER for Gilbert Errors using P=0.000001, p=0.001, and 1- *h*=0.8

	Start		
Load	999	5	Percent Improvement
0.1	0.079719	0.079719	0
0.3	0.253656	0.253656	0
0.5	0.424375	0.424375	0
0.7	0.597031	0.597031	0
0.9	0.722688	0.706813	-2.19666

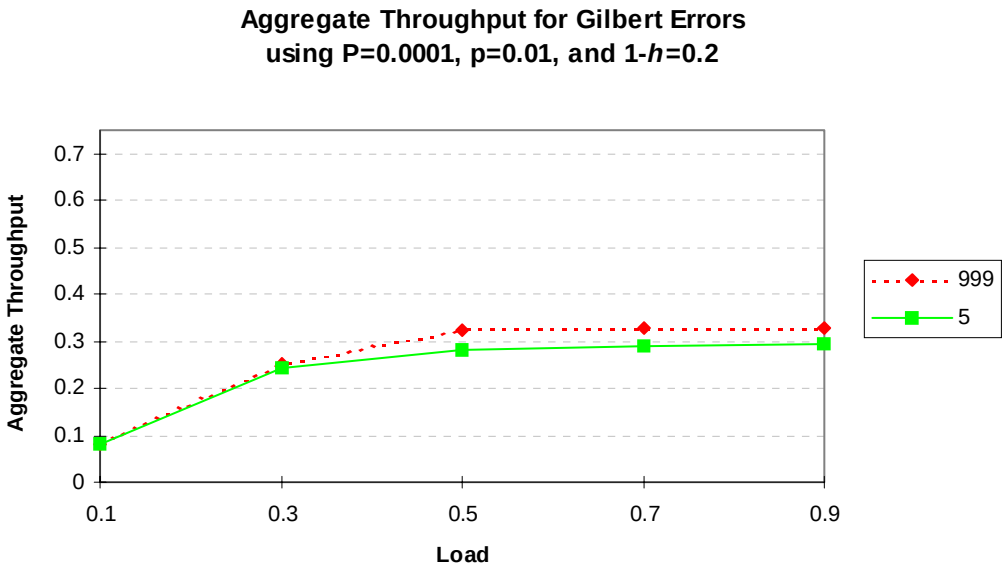

Figure 6-10. Aggregate Throughput for Gilbert Errors (P=0.0001, p=0.01, and 1- *h*=0.2)

Table 6-7. Comparison of Aggregate Throughput for 802.11 and CATER for Gilbert Errors using P=0.0001, p=0.01, and 1- *h*=0.2

	Start		
Load	999	5	Percent Improvement
0.1	0.079719	0.079625	-0.1176
0.3	0.253281	0.241531	-4.63911
0.5	0.325438	0.279625	-14.0772
0.7	0.327813	0.29175	-11.001
0.9	0.328031	0.295406	-9.9457

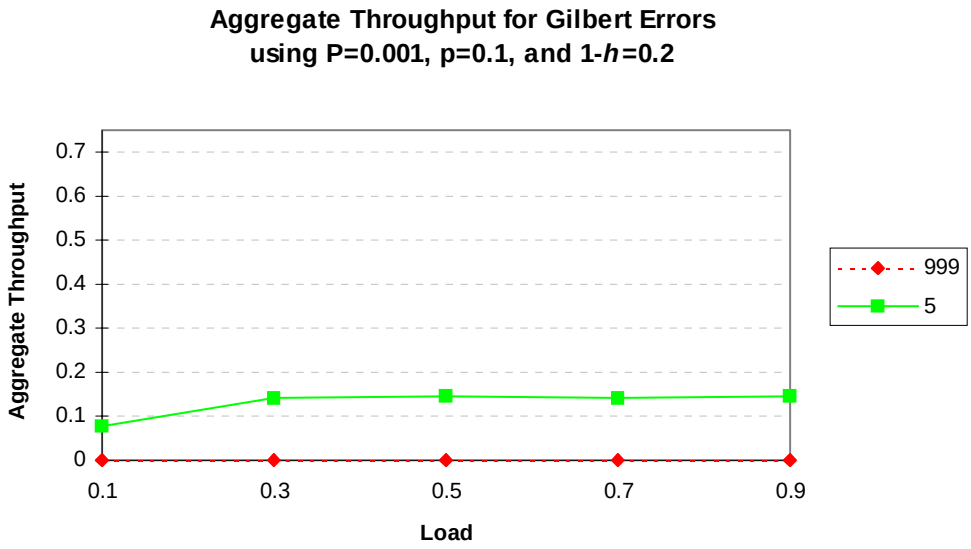

Figure 6-11. Aggregate Throughput for Gilbert Errors (P=0.001, p=0.1, and 1-*h*=0.2)

Table 6-8. Comparison of Aggregate Throughput for 802.11 and CATER for Gilbert Errors using P=0.001, p=0.1, and 1-*h*=0.2

	Start		
Load	999	5	Percent Improvement
0.1	0.001063	0.079469	7379.412
0.3	0.00125	0.139625	11070
0.5	0.00125	0.145094	11507.5
0.7	0.001656	0.142844	8524.528
0.9	0.001594	0.14425	8950.98

These three comparison plots provide an indication of protocol performance in the face of three different channels. It is seen that CATER performs best when the channel is degraded to the point that 802.11 is unable to operate. In all scenarios, aggregate throughput behaves similar to when static BERs are used as the error sources.

6.2.2 Confidence Intervals

The average relative confidence interval half widths are shown in Table 6-9. It is seen that these averages are relatively low indicating high confidence that the simulation closely approximates the actual population mean. However, two exceptions are noted when P is set to 0.001. The high values for the average when *Start* is set to 999 is a result of the aggregate throughput means being nearly 0, which, in turn, cause the relative half width to become large. These higher values are not an indication of a poor simulation. Relative confidence interval half widths are provided in Appendix E.

Table 6-9. Average Relative Confidence Interval Half Widths for Gilbert Errors

Gilbert Parameters			*Start*	
P	p	1-*h*	999	5
0.000001	0.001	0.2	2.4	2.6
0.000001	0.001	0.8	2.4	2.4
0.000001	0.01	0.2	2.3	2.6
0.000001	0.01	0.8	2.4	2.4
0.000001	0.1	0.2	2.5	2.6
0.000001	0.1	0.8	2.6	2.5
0.0001	0.01	0.2	2.0	6.1
0.0001	0.01	0.8	2.2	6.7
0.0001	0.1	0.2	2.0	7.4
0.0001	0.1	0.8	2.0	6.2
0.001	0.1	0.2	34.3	4.6
0.001	0.1	0.8	97.0	4.1

6.3 Fairness

Measuring aggregate throughput for the system does not provide insight into how individual stations are behaving. It is conceivable that one station could monopolize the channel to the detriment of all others such that aggregate throughput would still remain high due to the manner in which aggregate throughput is measured. Therefore, this section addresses the fairness of the protocols.

This section presents the results of simulations run to demonstrate fairness. A total of 24 simulations are run using a combination of the following settings: Load set to 0.1, 0.5, or 0.9; *Start* set to 999 (802.11) or 5 with *Max* set to 6 (CATER); and BER set to 0.01, 0.001, 0.0001, or 0.00001. Only one replication is run for each setting combination since the variable of interest is packet number versus time. Each simulation is run for 500 seconds. Although 24 simulations were run, only 18 produced data since throughput for a native 802.11 is 0 for degraded channels (BER > 0.0001) as can be verified in earlier sections in this chapter.

Fairness is measured by observing the number of frames successfully sent for each station. A fair protocol will ensure all stations are allowed equitable time to transmit on the channel. This is shown by plotting the identification number of each frame from each station sent across the channel versus time. An equitable protocol is evident by inspecting the plot—all stations send approximately the same number of frames in a given time span resulting in a linear relationship. Figure 6-12 and Figure 6-13 illustrate this relationship and are representative of the simulation results. All fairness plots are found in Appendix F.

Table 6-10 summarizes the fairness of the protocols via a comparison. Standard 802.11 (*Start* set to 999) is compared to CATER (*Start* set to 5 and *Max* set to 6) using the coefficient of variance (COV) as the metric. The number of successful frames for each station is recorded during the simulations. The mean of the 10 stations is shown in the table as well as the standard deviation. Defined as the ratio of the standard deviation to the mean, COV provides a unitless indication of system variance and is also shown in the table. Since the values in the COV column are small, variance amongst the stations is small. Thus, all stations operate fairly using both protocols. In addition, CATER and 802.11 are compared against each other. Although the COV is slightly larger when using CATER with a BER of 0.01 and 0.001, it is noted that 802.11 fails and the COV is undefined.

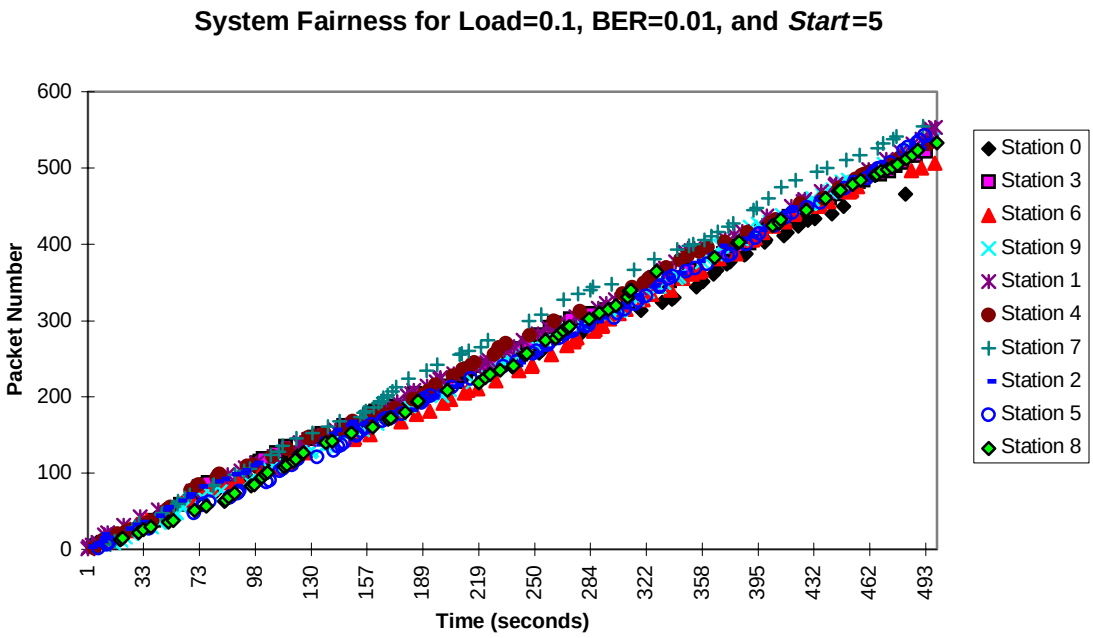

Figure 6-12. System Fairness for Load=0.1, BER=0.01, and *Start*=5

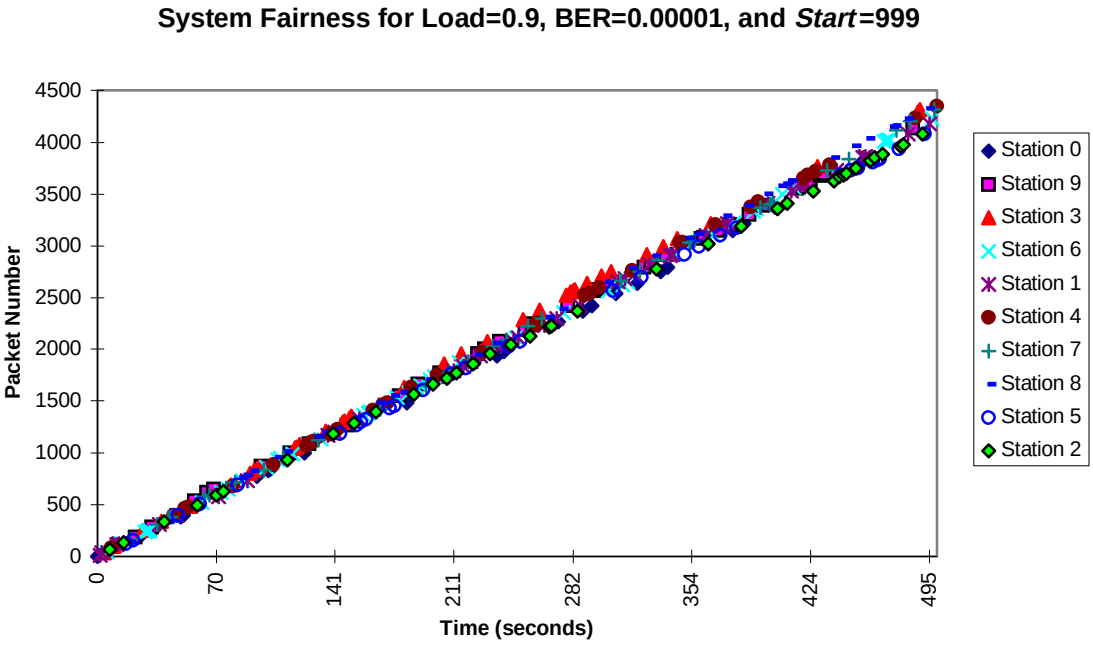

Figure 6-13. System Fairness for Load=0.9, BER=0.00001, and *Start*=999

Table 6-10. Fairness Comparison of CATER and 802.11 using Coefficient of Variance

Start	Load	BER	Mean	St.Dev.	COV
5	0.1	0.01	151.9	13.4	0.09
5	0.1	0.001	531.1	25.6	0.05
5	0.1	0.0001	532.2	25.3	0.05
5	0.1	0.00001	532.1	25.2	0.05
5	0.5	0.01	473	72	0.15
5	0.5	0.001	909.2	52.4	0.06
5	0.5	0.0001	1526.9	44.6	0.03
5	0.5	0.00001	2730.5	73.7	0.03
5	0.9	0.01	482.9	90.6	0.19
5	0.9	0.001	884.2	27.1	0.03
5	0.9	0.0001	1525.6	35.2	0.02
5	0.9	0.00001	4037.6	82.3	0.02
999	0.1	0.01			
999	0.1	0.001			
999	0.1	0.0001	532	25.3	0.05
999	0.1	0.00001	532.1	25.2	0.05
999	0.5	0.01			
999	0.5	0.001			
999	0.5	0.0001	2042.2	52.9	0.03
999	0.5	0.00001	2730.5	73.7	0.03
999	0.9	0.01			
999	0.9	0.001			
999	0.9	0.0001	2044.2	63.6	0.03
999	0.9	0.00001	4275.5	94.9	0.02

To provide further insight into the operation of CATER, a microscopic view of system fairness is provided. The macroscopic view provided by the fairness figures shown above show a near linear relationship between packet number and time. Zooming in on a smaller time scale illustrates the effect *Start* and *Max* have on CATER's performance. Figure 6-14 and Figure 6-15 provide this insight when *Start* is set to 5 and 999, respectively. It is seen that with *Max* set to 6 as in Figure 6-14 there are typically 7 frames transmitted over a reconfigured link yielding the bursty nature of frame traffic. Recall that *Max* is actually the parameter that controls the maximum number of additional frames to send over a

reconfigured link; thus, a total of 7 frames are sent in a burst. Figure 6-15 depicts system behavior when *Start* is set to 999 thereby disabling CATER. In this case, frame traffic is less bursty but still exhibits bursty periods.

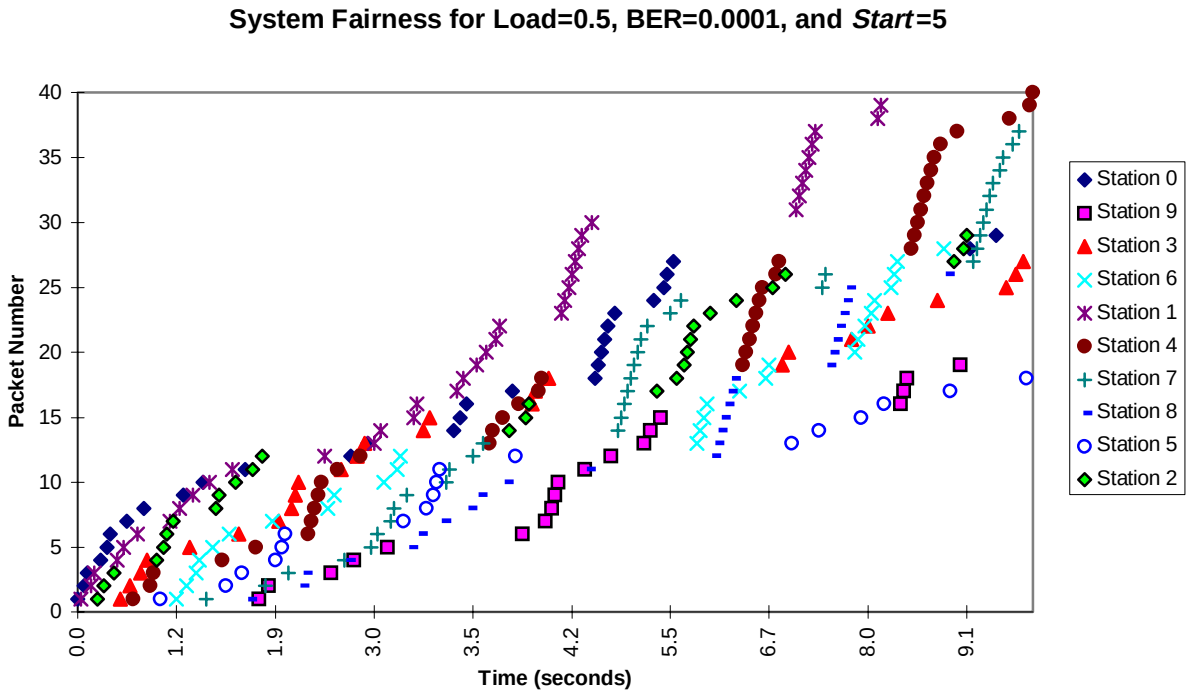

Figure 6-14. Microscopic View of System Fairness (Load=0.5, BER=0.0001, and *Start*=5)

A graphic illustration of CATER's performance is seen during high BER (Figure 6-16). The effect of *Max* is clear in this plot.

System fairness is illustrated in this section by means of plotting packet numbers against time. The near linear relationship between the two demonstrates the system behaves fairly across all stations. An unfair protocol would allow one or more stations to dominate the channel resulting in a graph where the unfair stations would have a significantly larger slope than the remaining stations.

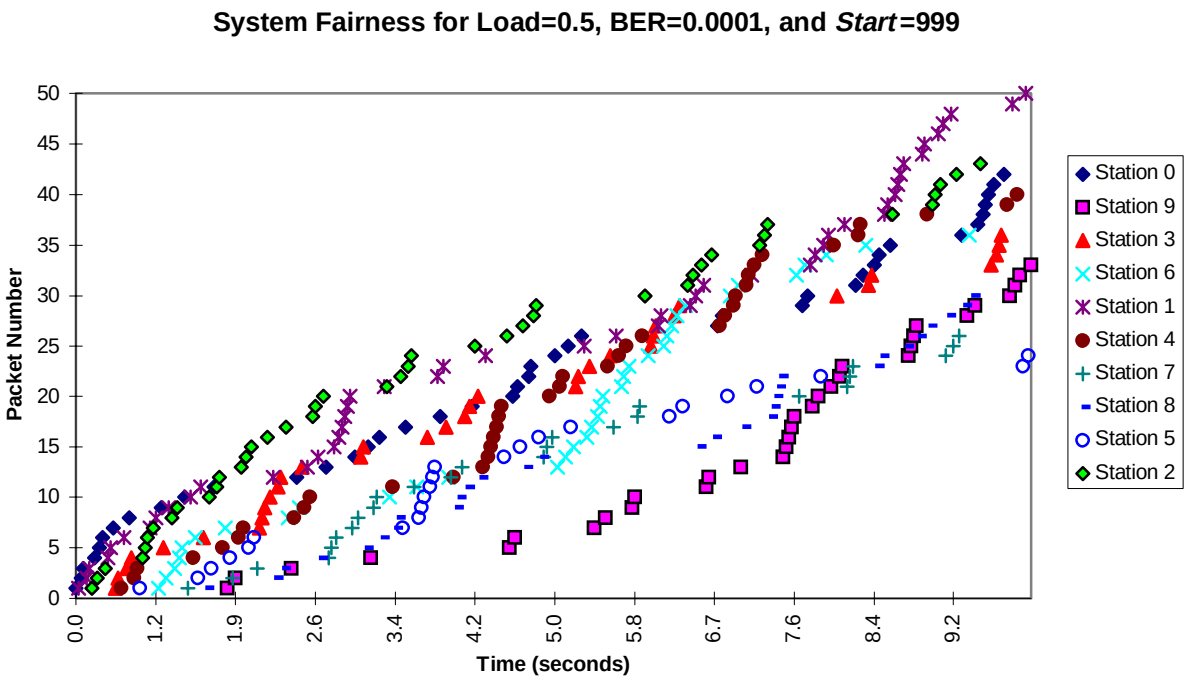

Figure 6-15. Microscopic View of System Fairness (Load=0.5, BER=0.0001, and *Start*=999)

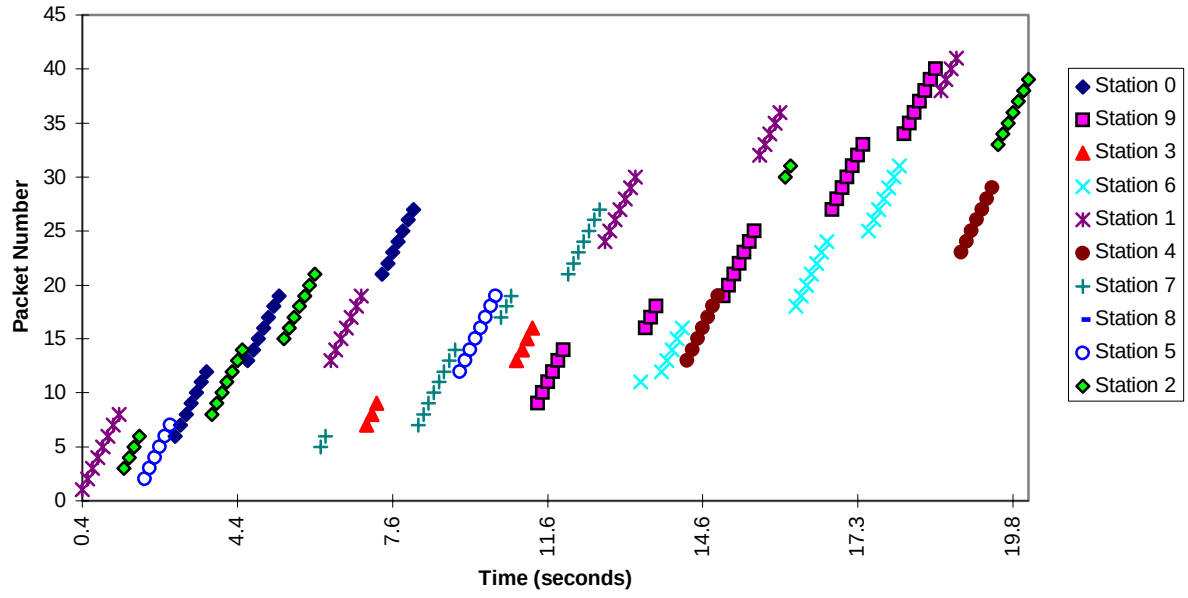

Figure 6-16. Microscopic View of System Fairness (Load=0.5, BER=0.01, and *Start*=5)

6.4 TCP Delay

As discussed in Chapter 2, TCP is responsible for the reliable end-to-end transport of data, error recovery and flow control by providing an idealized, full-duplex connection between two end systems [DiL92]. This section presents the results of simulations run to demonstrate TCP's performance over a range of operating conditions. A total of 24 simulations are run using a combination of the following settings: Load set to 0.1, 0.5, or 0.9; *Start* set to 999 (802.11) or 5 with *Max* set to 6 (CATER); and BER set to 0.01, 0.001, 0.0001, or 0.00001. Only one replication is run for each setting combination since the variable of interest is TCP delay time for individual packets. TCP data are collected from station 0; since system fairness is shown and each station uses the same traffic model, TCP data collected from station 0 is representative of all stations. Each simulation is run for 500 seconds. As seen in the fairness simulations, although 24 simulations are run, only 18 produce data. As can be verified in earlier sections in this chapter, throughput for a native 802.11 is 0 for degraded channels (BER > 0.0001).

The desirable outcome for these simulations is to minimize the number of TCP timeouts, since a timeout causes the retransmission of the packet in question. If the packet is simply delayed and not lost, this retransmission generates more network traffic further exacerbating the situation. Figure 6-17 illustrates a case where TCP does not timeout. The two lines on the graph are the retransmission timeout and the actual round-trip time for each packet. In addition, a timeout calculation marker is displayed to indicate the times when a TCP timeout calculation is made. Since TCP utilizes one timer for timeouts, not all packets are timed [WrS95]. A timeout marker displayed above the retransmission timeout line indicates an actual TCP timeout. A marker displayed at TCP Delay = -1 indicates a calculation without a timeout. In Figure 6-17, the timeout markers are so close together they almost appear as a thick solid line. Figure 6-18 demonstrates a simulation where TCP would timeout. All TCP timeout plots are found in Appendix G.

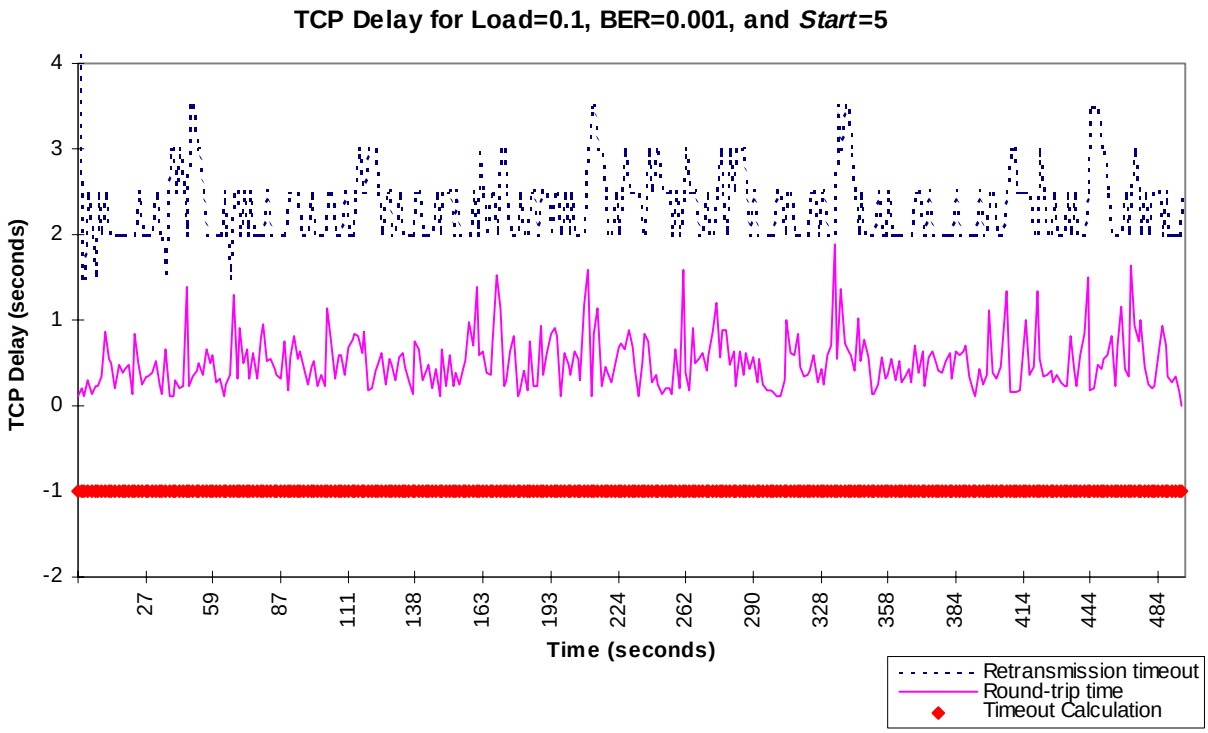

Figure 6-17. TCP Delay for Load=0.1, BER=0.001, and *Start*=5

Table 6-11 summaries TCP performance over the various settings. The number of TCP timeouts is shown for standard 802.11 (*Start* set to 999) and CATER (*Start* set to 5 and *Max* set to 6). In addition, to provide proper perspective on the relative number of times TCP times out, the total number of TCP timeout calculations are also shown. It is noted that 802.11 fails for all cases when BER is set to 0.01 or 0.001. In all cases except one (Load set to 0.9 and BER set to 0.00001) CATER experiences fewer TCP timeouts. In all cases, however, the ratio of the number of TCP timeouts to the total number of calculations is less than three percent.

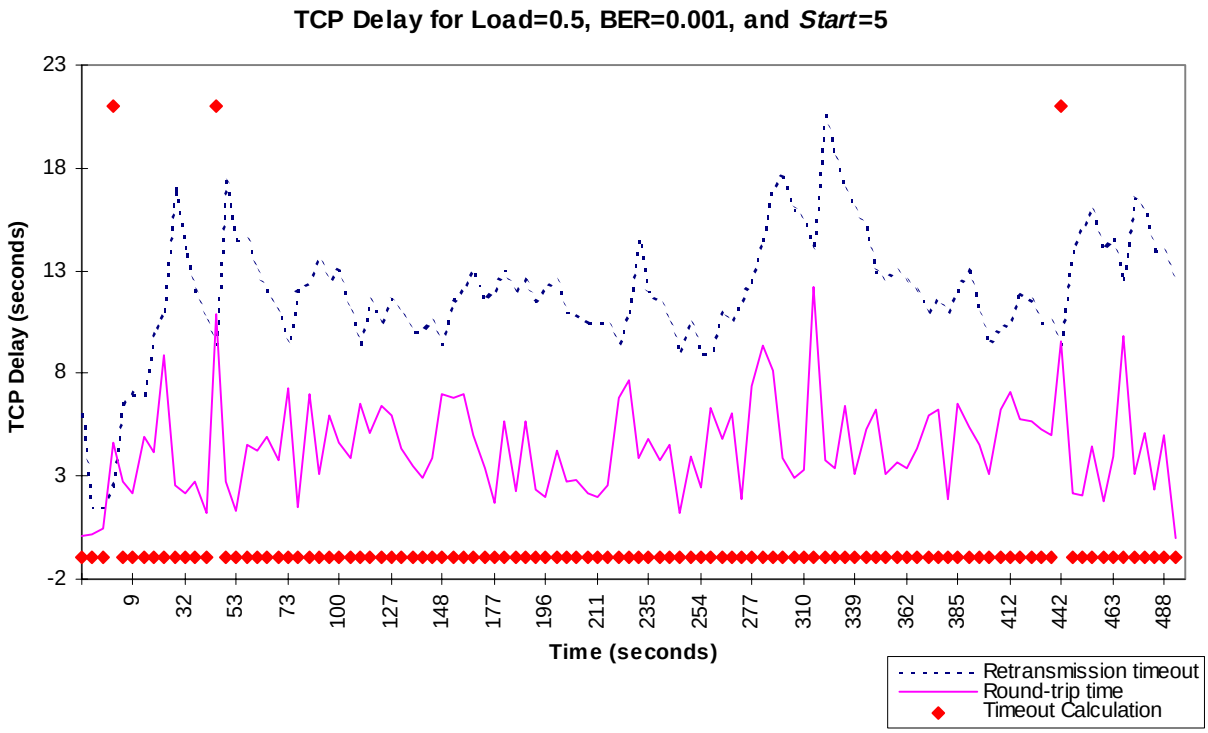

Figure 6-18. TCP Delay for Load=0.5, BER=0.001, and *Start*=5

Table 6-11. TCP Timeouts

Load	BER	802.11 Timeouts	CATER Timeouts	Number of 802.11 TCP Calculations	Number of CATER TCP Calculations
0.1	0.01	N/A	0	N/A	106
0.1	0.001	N/A	0	N/A	327
0.1	0.0001	0	0	463	458
0.1	0.00001	0	0	470	470
0.5	0.01	N/A	1	N/A	69
0.5	0.001	N/A	3	N/A	106
0.5	0.0001	5	2	468	229
0.5	0.00001	0	0	2397	2397
0.9	0.01	N/A	0	N/A	67
0.9	0.001	N/A	2	N/A	92
0.9	0.0001	0	0	247	171
0.9	0.00001	5	8	1160	1013

6.5 Summary

This chapter presented the results of the simulation phase of this research effort. It was shown that CATER performed better than standard IEEE 802.11 when considering the dynamic nature of wireless links. In general, aggregate throughput for a wireless LAN was improved substantially when using the CATER protocol. This improvement was demonstrated clearly using both static BERs and dynamic (Gilbert) BERs. It was shown that the impact of *Start* and *Max* were statistically significant! Also, when CATER was optimally configured (*Start* set to 5 and *Max* set to 6), it supported up to 14 percent aggregate throughput when standard 802.11 failed which represented a 273 percent increase for high BERs. Moreover, CATER's protocol overhead degraded performance of the network a slight 6.6 percent for low BERs.

As noted earlier, the static and dynamic BER models resulted in similar results in both CATER and IEEE 802.11; thus, only static BERs were used to demonstrate fairness and how TCP performed. CATER did not allow asymmetric use of the channel; all stations were allowed a fair share of the medium. In addition, the behavior of TCP was shown to be acceptable when using CATER; TCP retransmissions were kept to a minimum (less than 3 percent of the packets for worst case).

Simulation results indicate CATER offers a substantial improvement in aggregate throughput during periods of high BER while suffering only a slight decrease in throughput during periods of low BER. The CATER protocol offers an alternative to the rigid IEEE 802.11 standard and promises a more robust protocol and, consequently, more reliable wireless connectivity, which is the ultimate goal of any wireless LAN.

Chapter 7. Emulation

One machine can do the work of fifty ordinary men.
No machine can do the work of one extraordinary man.
—Elbert Hubbard.

Emulation can be defined as a simulation using hardware or firmware [Jai91]. In this research effort, the emulation is an implementation of the CATER protocol and IEEE 802.11 on a two station LAN (hereafter called the testbed). The ultimate goal of the emulation is to validate the simulation as described in Chapters 5 and 6. An ancillary goal is to verify the protocol's performance in the presence of real hardware limitations.

This chapter discusses the emulation of standard IEEE 802.11 and CATER on the testbed. The chapter begins with a discussion of the emulation environment in Section 7.1 followed by an explanation of the emulation model in Section 7.2. The next three sections present the parameters and variables unique to the emulation—Section 7.3 presents the emulation parameters that differ from the simulation parameters seen in Chapter 5; Section 7.4 discusses the emulation factors and their assigned values; and Section 7.5 discusses the response variable for the emulation. Emulation verification is presented in Section 7.6 followed by emulation results in Section 7.7. Validation of the simulation and emulation is the topic of Section 7.8. The chapter concludes with a summary in Section 7.9.

7.1 Emulation Environment

The emulation environment is composed of two computers connected via an Ethernet LAN. Both computers are identically equipped with a 100 MHz Pentium processor, 16 MB of RAM and a 3Com EtherLink III card. The computers use twisted pair cables connected through a hub for connectivity. During emulation runs, all other computers are disconnected from the hub to remove outside influences that could adversely affect results. Additionally, processes other than the emulation are not allowed on the stations during emulation once again to avoid contention of system resources that could skew emulation results.

Each computer is running Windows NT 4.0 and a complete implementation of the CATER protocol and standard IEEE 802.11. The emulation is written in the C programming language using Visual C++ 4.0. Designed for modularity, the 1100 lines of source code span six files. Two files, Client.cpp and Server.cpp, are the controlling software for their respective stations. These two files rely upon the services of the remaining four files to complete operations since the supporting files are designed for reuse. Thus, each station is in essence running code derived from five source files. The source listing is not contained within this document, but it may be obtained from the dissertation chairman.

7.2 Emulation Model

The basic emulation model is that of one source station and one destination station. The source station is the only station allowed to initiate data frame transmissions. Consequently, the destination station is concerned with receiving the data frames and sending an acknowledgment back to the source. Similarly, only the source station is allowed to transmit a reconfigure request frame, and the destination station acknowledges this by sending a reconfigure acknowledgment back to the source.

7.2.1 UDP Transport Layer

UDP (User Datagram Protocol) is used as the transport layer mechanism on top of IP. Included in the TCP/IP protocol suite, UDP is an unreliable, send-once-and-forget protocol that generates exactly one datagram for every output operation of the application program [Ste94]. UDP's unreliable nature provides an efficient means of removing packets (frames) from the network to emulate frame errors with no consequences such as retransmissions at the transport layer. Thus, each time a station sends a frame, it is actually sending a UDP packet across the Ethernet connection to the other station.

7.2.2 Two Station Rationale

The underlying reason for implementing two stations instead of the ten stations as seen in the simulation is that of sensing the medium. As discussed in Chapter 2, IEEE 802.11 must sense the medium before attempting to send a frame. The use of an Ethernet connection on the testbed computers precludes sensing the medium such that the emulation is cognizant. Therefore, only one station is allowed to act as the source thereby removing the medium sensing limitation. With just one source station, it is always assured that the medium is clear, since it is aware of the status of the destination station.

At first glance, this does not seem to be an equitable, fair validation technique for the simulation described in Chapter 5. However, since the response variable of interest is aggregate throughput and all stations within the simulation generate data frames at the same rate, all simulation stations exhibit the same properties with regard to aggregate throughput. The total number of successful frames traversing the network is not a function of the number of stations within the network but instead is a function of system load. Therefore, the validation of the simulation using the two station emulation is equitable.

A consequence of using the aforementioned two station environment is that collisions will not occur within the emulation. Therefore, the medium is used more efficiently since collisions waste bandwidth. However, collisions do occur in the simulation and must be accounted for when comparing the simulation and emulation results. The provisions made to account for this difference is discussed further in subsequent sections.

7.2.3 Frame Errors

As frames cross the network, they are subjected to bit errors as controlled by the BER factor discussed later in this chapter. Similar to the simulation, the stations within the emulation reject any frame that experiences any bit errors. In other words, one bit error renders the entire frame useless, and it is discarded at the receiving station. A binomial distribution is used within the emulation to determine if an error has occurred within a frame; this approach is also used within the simulation module provided within the Designer™ simulation package. However, instead of determining the status of every bit, a function is used to determine the status of each frame. Generating a bit-by-bit error pattern for each frame is prohibitively time consuming and does not offer additional accuracy given that the number or pattern of the bit errors is irrelevant. The following procedure is used by each station to determine if the frame is to be accepted. The station generates a random number and compares it against (BER * Number of bits in frame)/adjustment where adjustment is a scaling factor derived during the development of the emulation. If the random number is less than the result of the equation, the station declares the frame in error. With

only one calculation, the adjustment allows the equation to provide an extremely close approximation to the effects of the binomial distribution on a frame given that any bits errors corrupt the entire frame.

The adjustment value varies and is based on the BER and the number of bits to be tested; as such, there are a total of 14 adjustment values corresponding to seven BERs and two frame lengths (long data frames as well as short acknowledgment and reconfigure frames). The adjustment values are derived by comparing the number of frames in error when subjected to a complete bit-by-bit test versus the number of frames in error using the aforementioned formula. By running 10,000 frames through the bit-by-bit model and the emulation model, the total number of corrupt frames for each technique is observed. The adjustment value within the emulation model is then modified to more closely match the bit-by-bit model. This process is iterated until the two models agree within 5 percent of each other.

7.2.4 Data Structures

The emulation uses the data structure shown in Table 7-1 to represent frames traversing the network. The total number of bits within a frame including all overhead is contained in the Number of Bits field. The PN code length field contains the code length used to transmit a particular frame; it can either be 11 or 63. Frame ID is a unique number assigned to each new frame as it is transmitted and is used to verify the receipt of corresponding acknowledgments from the destination. BER is the bit error rate that this frame experiences as it crosses the network. The frame type field is used by the receiving station to determine what to do with the frame. The number of backoffs is used to keep track of the number of times a frame is sent to backoff. The number of additional frames field is used to coordinate the exchange of additional frames during a reconfigured period.

Table 7-1. Emulation Data Structure

Field	Data Type
Number of Bits	Integer
PN Code Length	Integer
Frame ID	Integer
BER	Real
Frame Type	{Data, ACK, Reconfigure, Reconfigure ACK}
Number of Backoffs	Integer
Number of Additional Frames	Integer

7.2.5 Program Operation

This section describes the operation of the emulation. It does not describe the operation of the protocols; for an explanation of IEEE 802.11 or CATER, see Chapters 2 and 4, respectively.

When a station sends a frame to the other, it performs several operations to ensure an accurate emulation of the network is rendered. First, the frame (data structure) is generated with appropriate values set for each of the fields. The sending station then delays or sleeps a prescribed amount of time, which is a function of the type of frame and PN code length. After this time delay, the station sends the frame to Windows NT for immediate transmission to the Ethernet connection via a UDP packet. The UDP packet is addressed specifically to the other station using its IP address.

Once the receiving station receives the frame, it first compares the frame's PN code length against the station's PN code length. If the two are not the same, the frame is discarded; however, if the two match, the frame is checked for bit errors using the aforementioned technique. The station accepts the frame if it is error free, but discards it otherwise.

Aggregate throughput is measured by counting the number of acknowledgment frames that successfully arrive at the source station assuming the frame ID for the acknowledgment frame matches the frame ID of the last sent data frame. This technique ensures multiple transmissions of one data frame are not all counted as distinct successful data frames thereby corrupting the aggregate throughput count.

7.3 Emulation Parameters

The parameters presented in this section are either unique to the emulation or are slightly different than the same parameter defined in the simulation. In both cases, the emphasis is on emulating a real wireless LAN testbed instead of making the emulation match the simulation or vice versa. If a parameter is not listed in this section but is present in the simulation explanation, the parameter and its value are the same for both the emulation and the simulation.

7.3.1 Number of Stations

As previously mentioned, the number of stations within the network is two—one source and one destination.

7.3.2 Timing Parameters

The times used in the emulation differ slightly than those in the simulation for two reasons. First, the resolution of the system timer available to the emulation is limited to 1 mS by Windows NT, whereas the simulation provides virtually infinite resolution. Therefore, it is impossible to provide an absolute

match between all simulation and emulation times. Second, the processing overhead of the emulation software extends parameter values. For all timing parameters, every effort is made to accurately model a real wireless LAN system within the emulation and simulation. To this end, some timing parameters are adjusted slightly to accommodate these two situations.

7.3.2.1 Data Frame Transmission Time

The time required to transmit a data frame is controlled by two parameters within the emulation; they are DATA_TIME_11 and DATA_TIME_63 corresponding to the transmission using 11 chip and 63 chips respectively. DATA_TIME_11 is set to 8 mS in the emulation and 8.578 mS in the simulation. DATA_TIME_63 is set to 48 mS in the emulation, whereas the simulation uses a value of 49.129 mS.

7.3.2.2 Reconfigure Frames Transmission Time

The time required to send a reconfigure frame and corresponding reconfigure acknowledgment is controlled by the parameter called RECONFIGURE_TIME within the emulation. Although the simulation uses a value of 2.2 mS, the emulation uses 1 mS for this parameter once again because the processing overhead extends the actual time to approximately 2 mS.

7.3.2.3 ACK Timeout

The time period allowed to elapse before a data frame is considered lost because an acknowledgment was not received for it is called the ACK timeout. ACK_TIMEOUT_11 controls how long the source station waits before it declares the data frame a failure when the data frame is sent using 11 chips. This parameter is set to 1 mS within the emulation and 0.4 mS in the simulation. ACK_TIMEOUT_63 performs the same operation but is used when the data frame is sent using 63 chips. ACK_TIMEOUT_63 is set to 2 mS in the emulation and 1.8 mS in the simulation.

7.3.2.4 Reconfigure ACK Timeout Period

RECONFIGURE_ACK_TIMEOUT is the parameter that controls the timeout period for a reconfigure request frame sent from the source to the destination. This parameter is set to 2 mS in the emulation, whereas the simulation uses a value of 1.8 mS.

7.3.2.5 Data Not Received Timeout Period

As discussed in Chapter 4, the destination station must maintain a timer to prevent station lockout if a data frame is unsuccessful over a reconfigured link. This timer, called DATA_TIMEOUT within the emulation, is the only timer found at the destination station and is set to 50 mS. The simulation uses 49.22

mS. It is important to note that this timeout period is longer than the DATA_TIME_63 time of 48 mS. If this was not the case the destination would continually timeout.

7.4 Factors

The emulation follows the same experimental design as the simulation. Therefore, the emulation uses the same factors and factors levels which are duplicated below inTable 7-2.

Table 7-2. Emulation Factors

Factor	Levels
BER	0.00001, 0.00002, 0.0001, 0.0002, 0.001, 0.002, 0.01
System Load	0.1, 0.3, 0.5, 0.7, 0.9
Start	3, 4, 5, 6, 7, 8, 9, 999 (disable reconfigure protocol)
Max	0, 2, 4, 6, 8, 10

7.5 Response Variable

The response variable is the same for both the emulation and simulation—aggregate throughput. The total number of frames that successfully traverse the network is recorded during the emulation. The data are collected after the transient removal time of 1 second has elapsed and before the end of the emulation at 51 seconds. As in the case of the simulation, the ultimate goal of the network is to maximize the aggregate throughput over all factor levels.

7.6 Model Verification

A variety of techniques are used to verify the correctness of the software starting with designing the foundation of the program itself using top-down modular design. Modular programming and top-down design are used throughout the development effort to facilitate development and testing of manageable pieces of functional code. This design philosophy enhances system verification by building small, easily-verifiable functions into a working system.

A technique known as antibugging [Jai91] is used to verify system operation. This technique involves monitoring system operation during runtime; variables such as frames sent and frames received are counted during the emulation and written to a file for post-emulation analysis. Fifteen difference variables are monitored and used in the post-emulation analysis.

A structured walk-through is used for some modules. To ensure proper operation of critical modules, a walk-through is accomplished with a fellow graduate student well versed in Windows NT socket programming and the C programming language.

Stepping through the code using the debug capabilities of the C programming tool surfaces several programming errors during initial development. This technique is also used to verify the system. In addition, trace information is written to a file in real time and later analyzed for proper sequencing and timing.

7.7 Emulation Results and Validation

This section discusses the results of the emulation as well as the validation of both the emulation and simulation. Since the emulation is implemented using two stations with one source, a comparison against the ten station simulation is not appropriate in all cases. The emulation will never experience a collision with another source station, whereas the ten station simulation does experience collisions. Moreover, when collisions occur in the ten station network where *Start* is set to a lower value (e.g., *Start* = 3), the adaptive protocol will engage resulting in more collisions. A comparison of Figure 7-1 and Figure 7-2 illustrates this phenomena. Contrasting the two graphs at lower BERs (i.e., 0.00001 and 0.00002) immediately illustrates this difference—throughput suffers due to collisions within the simulation but not within the emulation. Therefore, a comparison of the two station emulation and the ten station simulation is not always fitting. It is noteworthy to observe in Appendix B that as *Start* is increased, this degradation in throughput at lower BERs is not seen at nearly the same magnitude implying validation is possible for higher values of *Start*. The intent of this research, however, is not to provide a fragmented validation; consequently, the ten station simulation is not used. Instead, a three station simulation is used since this configuration experiences fewer collisions which more accurately models the emulation.

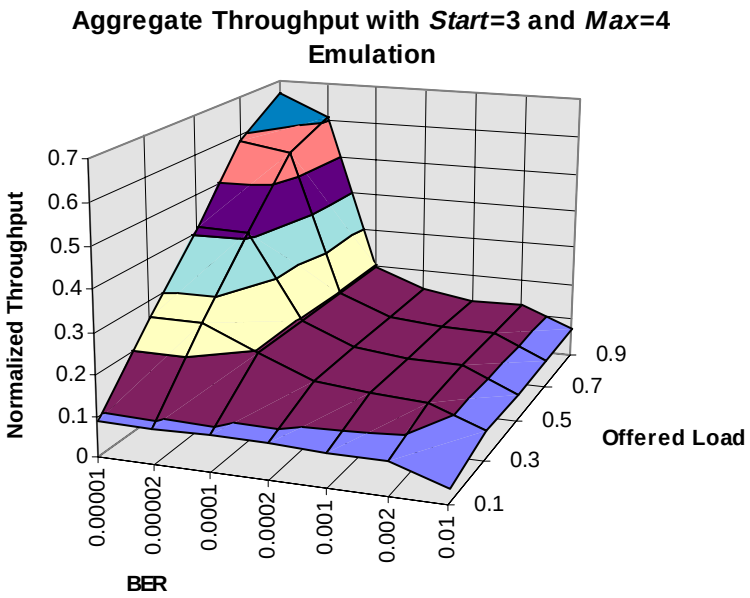

Figure 7-1. Emulation Results

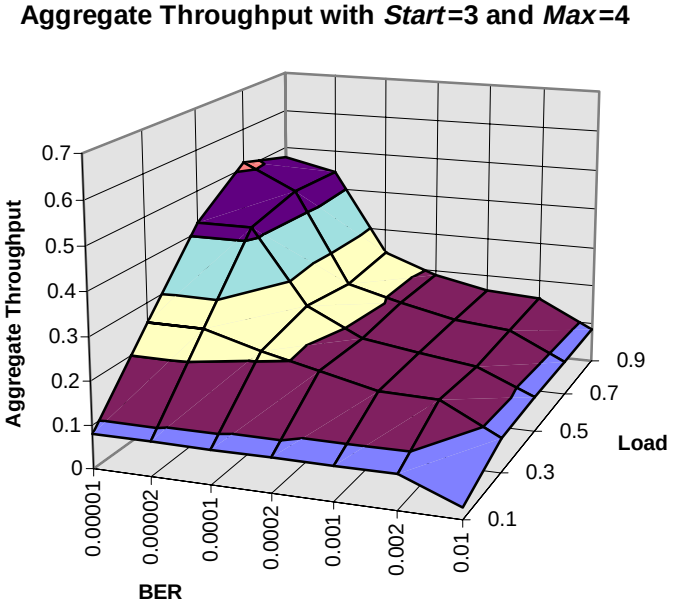

Figure 7-2. Ten Station Simulation Results

Since collisions prohibit a direct comparison of the emulation across all levels of the ten station simulation, the emulation is compared and validated against a three station simulation. This simulation configuration experiences far fewer collisions than the ten station simulation thereby providing a more suitable validation baseline. This is seen clearly by investigating the similarities between Figure 7-1 and Figure 7-3. The throughput does not drastically decrease for lower BERs. Therefore, this research effort conducts validation by comparing the emulation results with the three station simulation results.

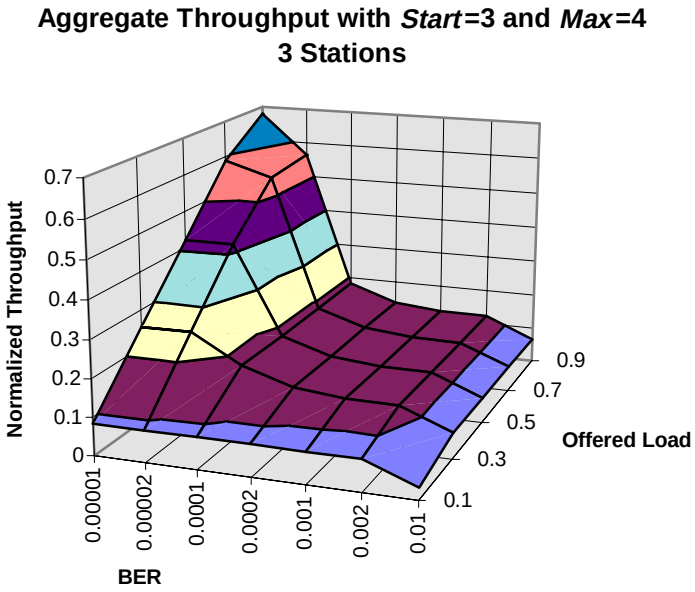

Figure 7-3. Three Station Simulation Results

7.7.1 Experimental Setup

The emulation follows the same experimental setup as the ten station simulation—a full factorial experiment with replications. Emulations are run for all combinations of factor levels with one exception. When the reconfigure protocol is disabled (*Start* = 999), *Max* is meaningless. Thus, the following emulation runs are made: all levels of BER (7), all levels of system load (5), all levels of *Max* (6), the first 7 levels (i.e., *Start* = 3 through 9) of *Start* (7), and 5 different random number seeds. This yields 7,350 emulation runs.

In addition, emulations are run with the reconfigure protocol disabled. This requires 7 levels of BER, 5 levels of system loads, and 5 different random number seeds, resulting in another 175 runs. Therefore, a total of 7,525 emulations are run which takes approximately 109 hours.

The three station simulation experimental setup is similar to the emulation. However, *Start* only ranges from 3 to 6, and *Max* only takes on the values of 0, 2, 4, and 6. A comparison of the emulation against this subset of a full complement of simulations still serves to validate both the simulation and the emulation results nicely as shown in subsequent sections.

7.7.2 Standard IEEE 802.11

Figure 7-4 illustrates the behavior of the emulation for native 802.11. As was seen with the simulation results in Chapter 6, 802.11 fails to function in the presence of high BERs. A visual comparison of the three station simulation results shown in Figure 7-5 and the emulation results in Figure 7-4 clearly supports the notion that the two systems exhibit the same behavior over the factor levels chosen for this investigation. Statistical analysis is used in a subsequent section to quantify the similarities between the emulation and the three station simulation.

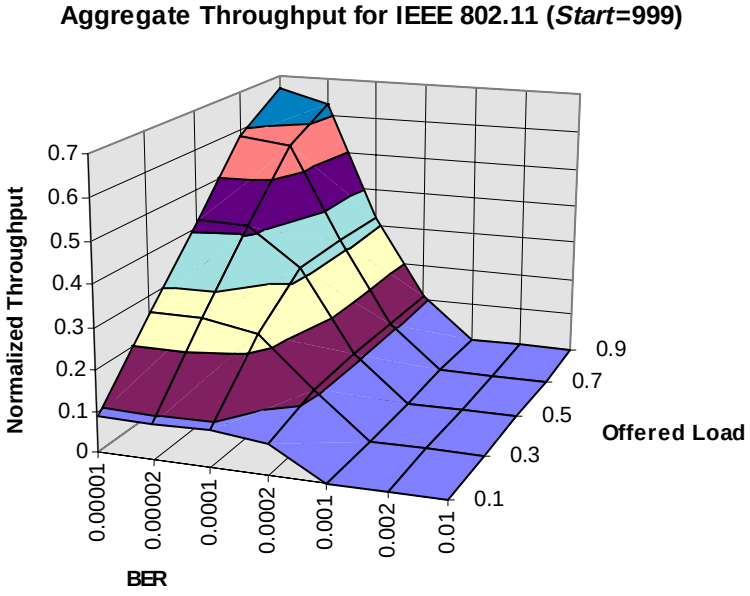

Figure 7-4. Emulation Results for IEEE 802.11

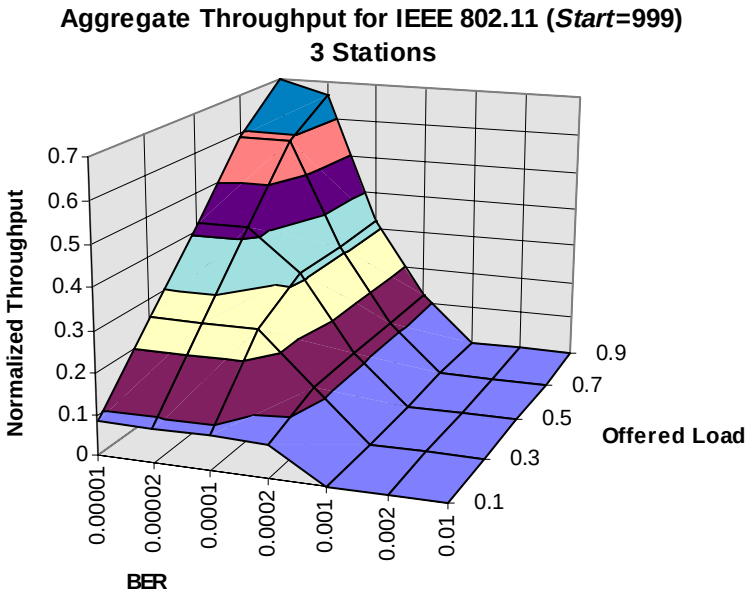

Figure 7-5. Three Station Simulation Results for IEEE 802.11

7.7.3 CATER

This section provides a glimpse into how the emulation and three station simulation function the same when running the CATER protocol. Two configurations are compared here; all configuration comparisons are found in Appendix H. Figure 7-6 depicts the emulation results when *Start* is set to 4 and *Max* is set to 0. Comparing this figure with the three station simulation results in Figure 7-7 clearly illustrates the similarities between the two.

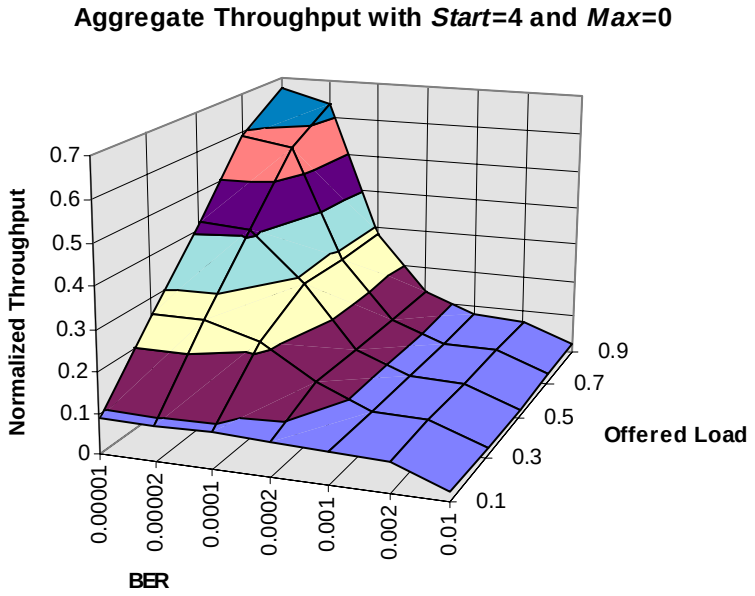

Figure 7-6. CATER Emulation Results when *Start*=4 and *Max*=0

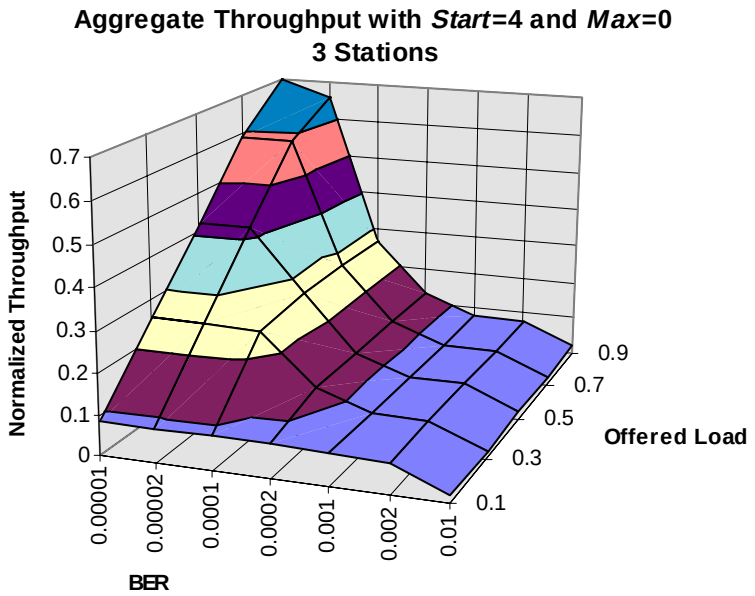

Figure 7-7. Three Station Simulation Results when *Start*=4 and *Max*=0

Another comparison is shown below when *Start* is set to 5 and *Max* is set to 6. Figure 7-8 illustrates the emulation results, whereas Figure 7-9 shows the three station simulation results. Once again, the similarities are striking!

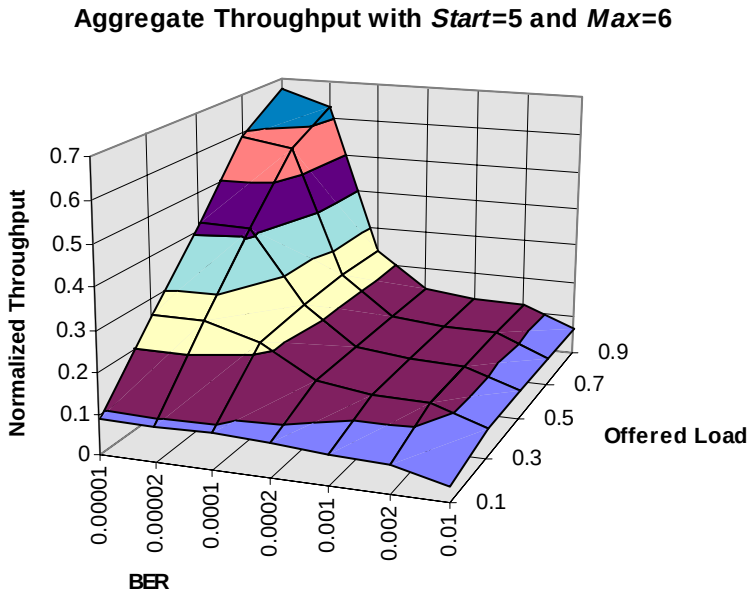

Figure 7-8. CATER Emulation Results when *Start*=5 and *Max*=6

7.7.4 Confidence Intervals

Confidence intervals for the emulation data and three station simulation data are discussed in this section. The same technique as was used in Chapter 6 is used here; the relative 90 percent confidence interval half width (relative H) is derived for each datum point (e.g., Load = 0.3 and BER = 0.0002) on all surface plots.

7.7.4.1 Emulation Data

Table 7-3 contains the relative H values for the emulation when *Start* is set to 5 and *Max* is set to 6. Appendix I contains all relative confidence interval half width tables for the emulation.

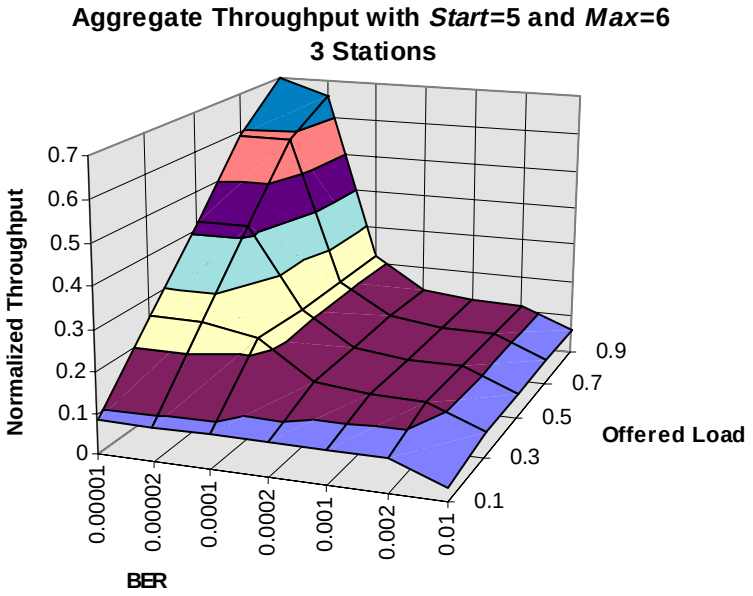

Figure 7-9. Three Station Simulation Results when *Start*=5 and *Max*=6

Table 7-3. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.233687	0.37276	0.679094	1.298326	7.538875
0.3	0.032384	0.032409	1.98629	1.959581	0.748078	1.666562	6.345448
0.5	0.110457	0.188325	4.310451	2.744284	1.364531	1.873054	3.454533
0.7	0.252788	0.356571	4.60577	1.494533	0.577421	0.855319	9.274696
0.9	0.34723	0.540598	3.016414	2.485182	1.68529	1.781526	6.933268

The average relative confidence interval half width is found by averaging all 35 datum points in Table 7-3. This is the same technique used in Chapter 6 and represents a figure of merit for the goodness of the emulation data when *Start* is 5 and *Max* is 6. As was seen in the ten station simulation results in Chapter 6, the lower values seen in the table are an indication that the mean values calculated from the 5 replications closely approximate the true mean implying the emulation duration is sufficient. The averaged relative half widths are shown inTable 7-4.

Table 7-4. Average Relative Confidence Interval Half Width for Emulation Data

	Start						
Max	3	4	5	6	7	8	9
0	1.3	1.6	1.8	1.4	1.6	1.7	1.7
2	1.6	1.8	1.5	1.9	1.9	2.0	2.3
4	1.7	1.4	2.0	1.9	2.0	2.3	2.1
6	1.6	1.5	2.0	1.8	1.9	2.1	2.4
8	1.4	1.7	1.4	2.0	1.9	1.9	2.2
10	1.4	1.6	1.6	2.0	1.8	2.1	2.1
IEEE 802.11 average relative half width = 1.1							

7.7.4.2 Three Station Simulation Data

The average relative confidence interval half width for the three station simulation data is found using the same technique discussed above and are displayed in Table 7-5. Once again, the lower values seen in the table are an indication that the mean values calculated from the 5 replications closely approximate the true mean implying the emulation duration is sufficient. Appendix J contains the relative half width tables for the three station simulation data.

Table 7-5. Average Relative Confidence Interval Half Width for Three Station Simulation Data

	Start			
Max	3	4	5	6
0	2.3	2.2	2.6	3.1
2	2.4	2.5	2.6	2.7
4	2.4	2.2	2.4	2.6
6	2.4	2.3	2.3	2.5
IEEE 802.11 average relative half width = 2.2				

7.7.5 Statistical Analysis

A visual comparison of the emulation and the three station simulation data highlights the similarities between the two systems. This visual comparison is a significant step in validating the emulation and the simulation, but it must be quantified. A paired-difference Student's t-test is performed

to test the hypothesis that the mean value of the differences of normalized aggregate throughput between the emulation data and the three station simulation data is zero.

The aggregate throughput from all levels of *Start* and *Max* available from the three station simulations is used. Table 7-5 lists the values of *Start* and *Max* used for the statistical analysis. A total of 595 pairs of aggregate throughput values are used in the analysis; this encompasses 7 levels of BER, 5 levels of Load, 4 levels of *Start*, and 4 levels of *Max* when CATER is used and an additional 7 levels of BER and 5 levels of Load for standard 802.11. A 90 percent confidence interval with 594 degrees of freedom is used for the Student's t-test resulting in a critical two-tailed value of ±1.647. Evaluating the t-test yields a t ratio of 0.712. Since this t value falls within the acceptance region of the t-distribution, the hypothesis is definitely upheld. There is no significant difference between the emulation run on the testbed and the three station simulation. Thus, the simulation is validated using a real-time hardware-in-the-loop implementation of the CATER protocol and standard IEEE 802.11. Consequently, the emulation is validated completely using the simulation results.

7.8 Summary

This chapter presented the emulation environment as well as the results of emulating the CATER and IEEE 802.11 protocols on the testbed. The emulation model was discussed and included the use of UDP packets to allow the loss of frames without retransmissions. The reasoning behind the two station emulation environment was then discussed followed by the implementation details of frame errors and the data structure. The general operation of the emulation program was then presented. Emulation parameters and factors were presented along with the response variable—normalized aggregate throughput. Several model verification techniques and how they were used to verify the model's code were discussed.

The remainder of the chapter was dedicated to emulation results and validation efforts. An explanation of why a ten station comparison with the emulation results would not be completely appropriate was presented. This was followed by a discussion on why a three station simulation could be used to validate the emulation. The relatively few collisions experienced in the three station simulation more closely matched the emulation which experiences no collisions. The experimental setup for the emulation was slightly different than the ten station simulation, and this was explained. A few emulation results were contrasted against the corresponding three station simulation results with amazing agreement; it was shown that both exhibit the same behavior over the factor levels used in the experiment. In fact, a paired-difference Student's t-test clearly supported the hypothesis that both systems are the same. This means the CATER and IEEE 802.11 models have been validated, and, therefore, the detailed simulation

described in Chapters 5 and 6 is an accurate representation of an actual wireless LAN running CATER and IEEE 802.11!

Chapter 8. Conclusions and Recommendations

To infinity … and beyond!!
—Buzz Lightyear, 1995, Walt Disney Pictures "Toy Story"

Everything that can be invented has been invented.
—Charles H. Duell, Commissioner, U.S. Office of Patents, 1899

This research effort examined the development and evaluation of a wireless local area network medium access control protocol called CATER. Designed for robustness, CATER improves throughput of an *ad hoc* LAN via adaptively varying the PN code length used for frame transmissions. Simulation models have been developed and simulated to verify performance improvements as well as provide a means to perform trade-off analyses. CATER was also emulated on a wireless LAN testbed to validate simulation results. This chapter chronicles the work performed during this dissertation as well as highlights significant achievements and extensions to the existing body of knowledge.

8.1 Summary of Research

Chapter 1 provided an introduction to the problem under investigation. Specifically, the importance of providing a reliable link between communicating wireless LAN stations was discussed. With the move towards more ubiquitous computing and mobility, wireless stations will be required to operate in often hostile environments suffering from various interfering phenomena such as multipath. The development of smaller computing devices as well as the increase in their applications drives the requirement for wireless LANs that offer the same quality of service experienced by wired LANs. To date, this quality of service is not available. This research extended the body of knowledge within this

field by advancing wireless networking to a level of reliability historically reserved for their wired counterparts.

Background information was provided in Chapter 2. Information pertinent to the development of wireless LAN was presented. A complete understanding of wireless LANs first requires insight into the operation and organization of wired LANs; this chapter provided this insight by presenting the OSI layered model of LANs, LAN topologies, and medium access control techniques. Performance of the medium access control techniques was presented and compared. It was seen that using a carrier sense mechanism significantly improves network throughput. Moreover, adding a collision detection capability further increases throughput. Collision detection is not always feasible; implementing a network using radios renders the collision detection capability impractical. Therefore, another technique known as collision avoidance is proposed by IEEE 802.11 to enhance throughput. Chapter 2 discussed this technique as implemented by IEEE 802.11 as well as other nuances of its implementation. An enabling technology for wireless LANs is spread spectrum communication. Chapter 2 presented a brief overview of the specific spread spectrum techniques used in this research effort. Specifically, characteristics of direct sequence spread spectrum systems that enable them to perform in adverse conditions was discussed. Chapter 2 concluded with a discussion of related research efforts. It was found that although literature existed for LANs, a scarcity existed for wireless LANs and their accompanying protocols.

The objectives of this research as well as the methods used to attain these objectives were the topic of Chapter 3. Following sound experimental procedures presented by Jain [Jai91], this chapter presented ten steps to successfully complete an experimental evaluation of a computer system. Beginning with a problem definition that included bounding assumptions for the network, this chapter delineated the scope of the research. Research methodology was presented as well.

Chapter 4 provided the first glance at CATER. Its operation was described in detail in addition to examples of the protocol in action. It was seen that CATER attempts to transmit frames using native IEEE 802.11 until the connection between two stations becomes suspect. After a prescribed number of failed transmission attempts, CATER reconfigures the link between the two stations and retransmits using a longer PN code. The longer PN code increases the process gain at the receiver thereby increasing the probability of successfully receiving the frame in question. It was shown that the use of the longer PN code resulted in a decrease in the data rate available to the two stations and the indiscriminate use of the longer PN code could actually exacerbate the situation. Chapter 4 postulated that by reconfiguring the link between the two stations only after a prescribed number of retransmissions had already occurred, network performance can improve.

The simulation environment was presented in Chapter 5. This chapter discussed the simulation of a wireless LAN using native IEEE 802.11 and CATER as the underlying MAC protocols. Designer™ was identified as the simulation tool used to construct, simulate and analyze the two protocols. It was shown that Designer™ used the propagation of data structures to mimic network traffic; these data structures were presented and explained in this chapter. The actual wireless LAN simulation model was then presented and explained in detail. The upper modules of the model's hierarchy were then presented along with an explanation of their general operation. Gilbert's two-state Markov chain model was introduced as the mechanism used to simulate bursty errors on a wireless channel. Since reliable bit error characteristics were not available for the channel in question, a sensitivity analysis was performed over the three Gilbert model parameters. The TCP model was then presented. Round trip times were collected for each successful frame and later analyzed using a spreadsheet designed to duplicate the timeout mechanism within TCP with the intent of determining how many times TCP would timeout. Simulation parameters were discussed and assigned values followed by system factors. Aggregate throughput was identified as the response variable to be maximized. Model verification and validation efforts were then discussed.

Chapter 6 presented the results of the simulation phase of this research effort. It was shown that CATER performed better than standard IEEE 802.11 when considering the dynamic nature of wireless links. Aggregate throughput for a wireless LAN was improved when using the CATER protocol. This was demonstrated using both static BERs and dynamic BERs. In addition, this chapter demonstrated that CATER does not allow asymmetric use of the channel; all stations were allowed a fair share of the medium. The behavior of TCP was shown to be acceptable when using CATER; TCP retransmissions were kept to a minimum (less than 3 percent of the time for worst case). Simulation results indicated that CATER offers a substantial improvement in aggregate throughput during periods of high BER while suffering only a slight decrease in throughput during periods of low BER. It was shown that CATER offers an alternative to the rigid IEEE 802.11 standard and promises a more robust protocol. When optimally configured, CATER offered normalized aggregate throughput of 14 percent when IEEE 802.11 failed!

Chapter 7 discussed the emulation used to validate the simulation results. This chapter presented the emulation environment as well as the results of emulating the CATER and IEEE 802.11 protocols on the testbed. The emulation model was discussed which included the use of UDP packets. The reasoning behind the two station emulation environment was then discussed followed by the implementation details of frame errors and the frame data structure. The general operation of the emulation program was then

presented. Emulation parameters and factors were presented along with the response variable—normalized aggregate throughput. Several model verification techniques and how they were used to verify the model's code were discussed. An explanation of validation configurations was explained. Emulation results were contrasted against the corresponding three station simulation results with amazing agreement; it was shown that both exhibit the same behavior over the factor levels used in the experiment. In fact, a paired-difference Student's t-test clearly supported the hypothesis that both systems are the same thereby validating the simulation!

8.2 Research Results

This research effort has developed and evaluated a novel medium access control protocol for wireless LANs designed to enhance the proposed IEEE 802.11 standard. Two evaluation techniques were used to determine the merit of CATER—simulation and emulation. This section presents the conclusions gathered from the two techniques.

By way of simulation, it is shown that CATER does, in fact, provide a more robust protocol than standard IEEE 802.11. Executing extensive simulations over all possible combinations of enumerated factors yielded a complete sensitivity analysis. Before a comparison was made between 802.11 and CATER, optimal values for *Start* and *Max* were derived using the sensitivity analysis. It was found that setting *Start* to 5 and *Max* to 6 yields near optimum performance for the network under investigation. These two values were used throughout when a comparison was made between the two protocols. Given the network configuration described in Chapters 3 and 5, it was shown that CATER increased aggregate throughput by 273 percent when operating at high BERs (between 0.0002 and 0.01 inclusive) while experiencing only a slight decrease (6.6 percent) in aggregate throughput for low BERs (between 0.00001 and 0.0001 inclusive). In fact, instead of no throughput using standard 802.11, CATER supported up to 14 percent of channel capacity—a substantial improvement for BERs up to 10^{-3}.

The preceding analysis was performed using static BERs. This research effort also considered how the protocols would perform under a more bursty source of bit errors as generated by a Gilbert model of a wireless channel. It was shown that both protocols behaved similar to the static BER situation.

System fairness was also demonstrated via simulations. By monitoring the packet numbers sent to the channel by each station, it was shown that CATER exhibited fairness amongst all stations. The coefficient of variance was found to be small for all simulations implying the number of packets transmitted by each station over time was approximately equal.

Retransmissions at the transport layer were investigated. It has been shown that retransmissions by TCP were not significantly increased when using CATER instead of standard 802.11; in fact, it was shown that CATER experienced the same or *fewer* TCP timeouts and retransmissions in all cases except one. It was shown that the ratio of the number of TCP timeouts to the total number of TCP calculations is less than three percent!

Emulations were used to validate the simulation with excellent results. Implemented on a real testbed and transmitting real frames, the emulation demonstrated a remarkable correlation with the simulation results.

8.3 Recommendations for Future Research

This research effort has extended the knowledge base of wireless local area networks and the protocols used to control the same. A novel, adaptive medium access protocol has been developed that significantly improves the robustness of wireless LANs. This work has been the first to examine the opportunistic use of the physical layer and the medium access control layer within a wireless LAN.

Although notable, the work presented in this dissertation is by no means a panacea to all the challenges presented by wireless networking. While this work has heightened the wireless LAN communities understanding of IEEE 802.11 and how to improve it, extensions and adaptations to CATER could prove valuable. During the course of this research effort, additional areas of investigation have presented themselves but have fallen outside the scope of this effort. It is recommended that the following research areas be investigated to advance the understanding of wireless LANs.

1. Develop a fully operational wireless LAN testbed and analyze the performance of CATER using spread spectrum transceivers in various locations within Whittemore Hall. This effort would provide further insight into the performance of the protocol.

2. Develop a demonstration program for use on the wireless LAN testbed to accentuate the advantage of using CATER when the network is experiencing excessive bit errors. The program would provide real time feedback during the course of moving stations around an area.

3. Investigate the impact of forward error correction on IEEE 802.11. As currently proposed 802.11 does not provide for forward error correction. This effort could determine if incorporating an error correction code is feasible and, if so, at what level should it be implemented.

4. Investigate an alternative source of bit errors within the simulation. The dynamic error model in this research effort was based on a two state Markov chain. Other models exist and could be used to

generate the bursty error patterns for the simulation. Also, empirical bit error data could be obtained and incorporated into the simulation.

5. Develop a closed form solution or another technique to determine the values of *Start* and *Max* given a specific network topology.

6. Analyze the behavior of CATER and 802.11 when confronted with a large number of stations. Examine how both protocols handle the increased demand on system resources.

7. Examine the impact of CATER on an infrastructure network configuration. CATER is designed to strengthen 802.11 running over an *ad hoc* network. There are issues needing resolution before CATER will offer an infrastructure network the same improvements as an *ad hoc* network. For example, how a reconfigured station behaves using the point coordination function is not completely known.

8. Investigate the impact of using alternative PN code lengths.

8.4 Conclusions

This dissertation encapsulates a major improvement in the performance of wireless networking by augmenting the existing body of networking knowledge. By providing more robust connectivity, the CATER protocol enhances the proposed IEEE 802.11 to the point that CATER maintains viable wireless links at times when 802.11 fails. Furthermore, CATER does not adversely impact the TCP transport protocol. In fact, CATER's performance with regard to TCP timeouts was superior to 802.11 in 16 percent of the test cases.

The results of this dissertation have been published in two papers. A portion has been published in the proceedings of the *IEEE International Conference on Communications* [MuD97a] and presented at the conference. The selection rate for this conference was approximately 43 percent. In addition, this work has been accepted for publication in the Association for Computing Machinery's journal titled *Mobile Computing and Communication Review* [MuD97b].

Although primarily targeted to LANs, the techniques and protocols developed do not limit themselves to this environment. Other wireless communication systems that utilize a two way direct sequence spread spectrum channel can benefit from this research as well. In this regard, this research has extended the research community's understanding of LANs in general such that wireless LANs can be viewed as a viable alternative to conventional terrestrial-based LANs in their quest to satisfy one of mankind's fundamental needs—communication.

Bibliography

Abr73 N. Abramson, “The ALOHA System,” In C*omputer-Communication Networks, Edited by N. Abramson and F.F. Kuo*, Prentice-Hall, Inc., Englewood Cliffs, NJ, pp. 501-517, 1973.

AnR95 J.B. Anderson, T.S. Rappaport, and S. Yoshida, “Propagation Measurements and Models for Wireless Communications Channels,” *IEEE Communications Magazine*, pp. 42-49, January 1995.

BaB94 D.F. Bantz and F.J. Bauchot, “Wireless LAN Design Alternatives,”*IEEE Network*, pp. 43-53, March/April 1994.

Bal95 R. Baldazo, “Local Air Networks,” *BYTE Magazine*, pp. 201-206, June 1995.

BeG92 D. Bertsekas and R. Gallager, *Data Networks, Second Edition*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1992.

BeM63 J.M. Berger and B. Mandelbrot, “A New Model for Error Clustering in Telephone Circuits,” *IBM Journal of Research and Development*, Vol. 7, pp. 224-236, July 1963.

BhD94 V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, “MACAW: A Medium Access Protocol for Wireless LAN’s,” *Computer Communication Review*, Vol. 24, No. 4, pp. 212-225, October 1994.

Bib92 K. Biba, “A Hybrid Wireless MAC Protocol Supporting Asynchronous and Synchronous MSDU Delivery Services,” *IEEE 802.11 Working Group paper 802.11/91-92* , September 1992.

Bis93 S.K. Biswas, “Performance Modeling of Window Based Flow Control in a Multihop Packet Radio LAN,” *Proceedings of the IEEE INFOCOM*, pp. 10b.3.1-10b.3.10, 1993.

Boy95 P. Boyle, "Wireless LANs No Strings Attached," *PC Magazine*, pp. 215-237, January 10, 1995.

BrB95 E. Brewer, *et al.*, "Design of Wireless Portable Systems," *Proceedings of the IEEE COMPCOM*, pp. 169-176, 1995.

BrP95 L.S. Brakmo and L.L. Peterson, "TCP Vegas: End to End Congestion Avoidance on a Global Internet," *IEEE Journal on Selected Areas in Communications*, Vol. 13, No. 8, pp. 1465-1480, October 1995.

Che92 J.Y.C. Cheah, "A Proposed Access Method for the Wireless LAN Standard," *Proceedings of the 3rd International Symposium on Personal, Indoor, and Mobile Radio Communications*, pp. 5.3.1-5.3.4, 1992.

Che94 K.C. Chen, "Medium Access Control of Wireless LANs for Mobile Computing," *IEEE Network*, Vol. 8, No. 5, pp. 50-63, September/October 1994.

ChH72 R.T. Chien, A.H. Haddad, B. Goldberg, and E. Moyes, "An Analytic Error Model for Real Channels," *Proceedings of the IEEE International Conference on Communications*, pp. 15-7 - 15-12, 1972.

ChK91 D. Choi and B.G. Kim, "The Expected (Not Worst-Case) Throughput of the Ethernet Protocol," *IEEE Transactions on Computers*, Vol. 40, No. 3, pp. 245-252, March 1991.

ChR85 G.L. Choudhury and S.S. Rappaport, "Priority Access Schemes Using CSMA-CD," *IEEE Transactions on Communications*, Vol. 33, No. 7, pp. 620-626, July 1985.

ClP78 D.D. Clark, K.T. Pogram, and D.P. Reed, "An Introduction to Local Area Networks," *Proceedings of the IEEE*, Vol. 66, pp. 1497-1517, November 1978.

Cof95 P. Coffee, "Techniques Spread Wireless Benefits," *PC Week*, pp.72,76, July 10, 1995.

Col90 G.D. Cole, *Implementing OSI Networks*, John Wiley & Sons, Inc., New York, 1990.

CoL83 E.J. Coyle and B. Liu, "Finite Population CSMA/CD Networks," *IEEE Transactions on Communications*, Vol. 31, No. 11, pp. 1247-1251, November 1983.

Com93 *BONeS® Designer™ Introductory Overview*, Comdisco Systems Incorporated, June 1993.

Com95 D.E. Comer, *Internetworking with TCP/IP, Volume 1, Principles, Protocols, and Architecture, Third Edition*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1995.

CoM83 C.E. Cook and H.S. Marsh, "An Introduction to Spread Spectrum," *IEEE Communications Magazine*, Vol. 21, No. 2, pp. 8-16, March 1983.

Con83 J.W. Conard, "Services and Protocols of the Data-Link Layer," *Proceedings of the IEEE*, Vol. 71, No. 12, pp. 1378-1383, December 1983.

Cou93 L.W. Couch II, *Digital and Analog Communication Systems, Fourth Edition*, Macmillan Publishing Company, New York, 1993.

CrW96 B. Crow, *et al*., "Performance of IEEE 802.11 Wireless Local Area Networks," *Proceedings of the SPIE - The International Society of Optical Engineering*, Vol. 2917, pp. 480-491, 1996.

DaE93 S. Dastangoo, R. Eftekari, and H. Tran, "Wireless LAN Technologies and Applications," *Proceedings of the IEEE MILCOM*, pp. 497-501, 1993.

DaJ91 P.B. Danzig and S. Jamin, "tcplib: A Library of TCP Internetwork Traffic Characteristics," Technical Report USC-CS-91-495, anonymous ftp from *jerico.usc.edu/pub/jamin/tcplib/tcplibtr.ps.Z* Computer Science Department, University of Southern California, 1991.

DaZ83 J.D. Day and H. Zimmermann, The OSI Reference Model," *Proceedings of the IEEE*, Vol. 71, No. 12, pp. 1334-1340, December 1983.

DiL92 G. Dickson and A. Lloyd, *Open Systems Interconnection*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1992.

Dix94 R.C. Dixon, *Spread Spectrum Systems with Commercial Applications, Third Edition*, John Wiley & Sons, Inc., New York, 1994.

DrM88 S. Dravida, M.J. Master, and C.H. Morton, "A Method to Analyze Performance of Digital Connections," *IEEE Transactions on Communications*, Vol. 36, No. 3, pp. 298-305, March 1988.

DuR96 R. Dube, C.D. Rais, and S.K. Tripathi, "Improving NFS Performance over Wireless Links," Technical Report CS-TR-3614, anonymous ftp from *ftp.cs.umd.edu/pub/papers/papers/3583/3583.ps.Z* Department of Computer Science, University of Maryland, 1996.

Ell63 E.O. Elliott, "Estimates of Error Rates for codes on Burst-Noise Channels," *Bell System Technical Journal*, Vol. 42, pp. 1977-1997, September 1963.

Fla94 A. Flatman, "Wireless LANs: Developments in Technology and Standards,"*Computing & Control Engineering Journal*, pp. 219-224, October 1994.

FrE80 P. Freret, R. Eschenbach, D. Crawford and P. Braisted, "Applications of Spread-Spectrum Radio to Wireless Terminal Communications," *Proceedings of the IEEE National Telecommunications Conference*, pp. 69.7.1-69.7.4, December 1980.

Fre91 T.A. Freeburg, "Enabling Technologies for Wireless In-Building Network Communications—Four Technical Challenges, Four Solutions,"*IEEE Communication Magazine*, pp. 58-64, April 1991.

Fri67 B.D. Fritchman, "A Binary Channel Characterization Using Partitioned Markov Chains," *IEEE Transactions on Information Theory*, Vol. IT-13, No. 2, pp. 221-227, April 1967.

GfB79 F. Gfeller and U. Bapst, "Wireless In-house Data Communication Via Diffused Radiation," *Proceedings of the IEEE*, Vol. 67, pp. 1474-1486, Nov. 1979.

Gil60 E.N. Gilbert, "Capacity of a Burst-Noise Channel," *Bell Systems Technical Journal*, Vol. 39, pp. 1253-1265, 1960.

Gol94 L. Goldberg, "MAC Protocols: The Key to Robust Wireless Systems," *Electronic Design*, Vol. 42, No. 12, pp. 63-74, June 13, 1994.

Gon85 T.A. Gonsalves, "Measured Performance of the Ethernet," In *Advances in Local Area Networks*, Edited by K. Kummerle, F.A. Tobagi, and J.O. Limb, IEEE Press, Institute of Electrical and Electronics Engineers, Inc., New York, pp. 383-410, 1987.

GuM94 N. Guo and S.D. Morgera, "Frequency-Hopped ARQ for Wireless Network Data Services," *IEEE Journal on Selected Areas in Communications*, Vol. 12, No. 8, pp. 1324-1337, October 1994.

HaK89 I.M.I. Habbab, M. Kavehrad, and C.W. Sundberg, "ALOHA with Capture Over Slow and Fast Fading Radio Channels with Coding and Diversity," *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 1, pp. 79-88, January 1989.

Hay91 V. Hayes, "Standardization Efforts for Wireless LANs," *IEEE Network Magazine*, pp. 19-20, November 1991.

IEE85 ANSI/IEEE Std. 802.3-1985, *Carrier Sense Multiple Access with Collision Detection (CSMA/CD)*, 1985.

IEE94 Editors of IEEE 802.11, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Draft Standard IEEE 802.11, P802.11/D1*, Institute of Electrical and Electronics Engineers, Inc., New York, December 1, 1994.

IEE96 Editors of IEEE 802.11, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Draft Standard IEEE 802.11, P802.11/D4.0*, Institute of Electrical and Electronics Engineers, Inc., New York, May 20, 1996.

Jai91 R. Jain, *The Art of Computer Systems Performance Analysis*, John Wiley & Sons, Inc., New York, 1991.

Kar90 P. Karn, "MACA - A New Channel Access Method for Packet Radio," *In Proceedings of ARRL/CRRL Amateur Radio 9th Computer Networking Conference*, September 22, 1990.

KaS78 L.N. Kanal and A.R.K. Sastry, "Models for Channels with Memory and Their Applications to Error Control," *Proceedings of the IEEE*, Vol. 66, No. 7, pp. 724-744, July 1978.

Kei89 G.E. Keiser, *Local Area Networks*, McGraw-Hill Book Company, New York, 1989.

Kle76 L. Kleinrock, *Queueing Systems Volume 2: Computer Applications*, John Wiley & Sons, Inc., New York, 1976.

KlT75 L. Kleinrock and F.A. Tobagi, "Packet Switching in Radio Channels: Part 1—Carrier Sense Multiple-Access Modes and Their Throughput-Delay Characteristics,"*IEEE Transactions on Communications*, Vol. 23, No. 12, pp. 1400-1416, December 1975.

KoN92 J. Konorski, K. Nowicki, and J. Wozniak, "The Cooperative Random Token Protocol for High-Speed Radio LAN's," *Proceedings of the IEEE GLOBECOM*, pp. 974-978, 1992.

KoY77 H. Kobayashi, Y. Onozato, and D. Huynh, "An Approximate Method for Design and Analysis of an ALOHA System," *IEEE Transactions on Communications*, Vol. COM-25, No. 1, pp. 148-157, January 1977.

Kru92 J. Kruys, HIPERLAN, Applications and Requirements, *Proceedings of the 3rd International Symposium on Personal, Indoor, and Mobile Radio Communications* , pp. 5.1.1-5.1.6, 1992.

LaK94 R.O. LaMaire, A. Krishna, and H. Ahmadi, "Analysis of a Wireless MAC Protocol with Client-Server Traffic and Capture," *IEEE Journal on Selected Areas in Communications*, Vol. 12, No. 8, pp.1299-1313, October 1994.

LiD94 C. Links, W. Diepstraten, and V. Hayes, "Universal Wireless LANs," *Byte Magazine*, pp. 99-108, May 1994.

Mac92 M.H. MacDougall, *Simulating Computer Systems Techniques and Tools*, The MIT Press, Cambridge, MA, 1992.

MaN94 D.T. Magill, F.D. Natali, and G.P. Edwards, "Spread-Spectrum Technology for Commercial Applications," *Proceedings of the IEEE*, Vol. 82, No. 4, pp. 572-584, April 1994.

MeB76 R.M. Metcalfe and D.R. Boggs, "Ethernet: Distributed Packet Switching for Local Computer Networks," *Communications of the ACM*, Vol. 19, pp. 395-404, July 1976.

MeL83 J.S. Meditch and C.A. Lea, "Stability and Optimization of the CSMA and CSMA/CD Channels," *IEEE Transactions on Communications*, Vol. 31, pp. 763-774, June 1983.

Met76 J.J. Metzner, "On Improving Utilization in ALOHA Networks," *IEEE Transactions on Communications*, Vol. 24, pp. 447-448, April 1976.

Mit91 J.E. Mitzlaff, "Radio Propagation and Anti-Multipath Techniques in the WIN Environment," *IEEE Network Magazine*, pp. 21-26, November 1991.

MuB89 M. Mushkin and I. Bar-David, "Capacity and Coding for the Gilbert-Elliott Channels," *IEEE Transactions on Information Theory*, Vol. 35, No. 6, pp. 1277-1290, November 1989.

MuD97a B.E. Mullins, N.J. Davis IV, and S.F. Midkiff, "A Wireless Local Area Network Protocol That Improves Throughput Via Adaptive Control," *Proceedings of the IEEE International Conference on Communications*, pp. 1427-1431, June 1997.

MuD97b B.E. Mullins, N.J. Davis IV, and S.F. Midkiff, "An Adaptive Wireless Local Area Network Protocol That Improves Throughput Via Adaptive Control of Direct Sequence Spread Spectrum Parameters," To appear in *ACM Mobile Computing and Communication Review*, Vol. 1, No. 3, 1997.

NeT95 D. Newman and K. Tolly, "Wireless LANs: How Far? How Fast?," *Data Communications*, Vol. 24, No. 4, pp. 77-86, March 21, 1995.

Pah85 K. Pahlavan, "Wireless Communications for Office Information Networks," *IEEE Communications Magazine*, Vol. 23, No. 6, pp. 19-27, June 1985.

PaP95 K. Pahlavan, T.H. Probert, and M.E. Chase, "Trends in Local Wireless Networks," *IEEE Communications Magazine*, Vol. 33, No. 3, pp. 88-95, March 1995.

PrB86 T. Pratt and C.W. Bostian, *Satellite Communications*, John Wiley & Sons, Inc., New York, 1986.

Rob75 L.G. Roberts, "ALOHA Packet System with and without Slots and Capture," *ACM SIGCOM Computer Communication Review*, Vol. 5, pp. 28-42, April 1975.

Ryp92 C.A. Rypinski, "Motivation for Centralized Wireless LAN Functions," *Proceedings of the 3rd International Symposium on Personal, Indoor, and Mobile Radio Communications*, pp. 5.5.1-5.5.6, 1992.

SaI95 M.N.O. Sadiku and M. Ilyas, *Simulation of Local Area Networks*, CRC Press, Inc., Boca Raton, FL, 1995.

SAS85a SAS Institute Inc., *SAS® Introductory Guide, Third Edition*, Cary, NC, 1985.

SAS85b SAS Institute Inc., *SAS® User's Guide: Statistics, Version 5 Edition*, Cary, NC, 1985.

Sch82 R.A. Scholtz, "The Origins of Spread-Spectrum Communications," *IEEE Transactions on Communications*, Vol. 30, pp. 822-854, May 1982.

ShH82 N. Shacham and V.B. Hunt, "Performance Evaluation of the CSMA/CD (1-persistent) Channel-Access Protocol in Common-Channel Local Networks," In *Local Computer Networks, Edited by P.C. Ravasio, G. Hopkins, and N. Naffah*, North-Holland Publishing Company, pp. 401-414, 1982.

SiG85 R. Sinha and S.C. Gupta, "Mobile Packet Radio Networks: State-of-the-Art," *IEEE Communications Magazine*, Vol. 23, No. 3, pp. 53-61, March 1985.

SoM87 K. Sohraby, M.L. Molle, and A.N. Venetsanopoulos, "Comments on 'Throughput Analysis for Persistent CSMA Systems'," *IEEE Transactions on Communications*, Vol. 35, No. 2, pp. 240-243, February 1987.

Sta91 W. Stallings, *Data and Computer Communications, Third Edition*, Macmillan Publishing Company, New York. 1991.

Ste94 W.R. Stevens, *TCP/IP Illustrated, Volume 1 The Protocols*, Addison-Wesley Publishing Company, Reading, MA, 1994.

SwF91 F. Swarts and H.C. Ferreira, "Modeling and Performance Evaluation of Mobile VHF Radio Channels Employing FSK, DPSK, QPSK, and 8-ary PSK as Modulation Schemes," *Proceedings of IEEE GLOBECOM*, pp. 1935-1939, 1991.

SwF93 F. Swarts and H.C. Ferreira, "Markov Characterization of Channels with Soft Decision Outputs," *IEEE Transactions on Communications*, Vol. 41, No. 5, pp. 678-682, May 1993.

SwF94 F. Swarts and H.C. Ferreira, "Markov Characterization of Digital Fading Mobile VHF Channels," *IEEE Transactions on Vehicular Technology*, Vol. 43, No. 4, pp. 977-985, November 1994.

TaK85 H. Takagi and L. Kleinrock, "Throughput Analysis for Persistent CSMA Systems," *IEEE Transactions on Communications*, Vol. 33, No. 7, pp. 627-638, July 1985.

TaK87 H. Takagi and L. Kleinrock, "Correction to 'Throughput Analysis for Persistent CSMA Systems'," *IEEE Transactions on Communications*, Vol. 35, No. 2, pp. 243-245, February 1987.

Tan88 A.S. Tanenbaum, *Computer Networks, Second Edition*, Prentice-Hall, Inc., Englewood Cliffs, NJ, 1988.

Tas86 S. Tasaka, "Dynamic Behavior of a CSMA-CD System with a Finite Population of Buffered Users," *IEEE Transactions on Communications*, Vol. 34, No. 6, pp. 576-586, June 1986.

ToH80 F.A. Tobagi and V.B. Hunt, "Performance Analysis of Carrier Sense Multiple Access with Collision Detection," *Computer Networks*, Vol. 4, pp. 245-259, November 1980.

TrL95 W.H. Tranter and O.H. Lee, "On the Use of Signal-to-Noise Ratio Estimation for Establishing Hidden Markov Models," *Proceedings of Virginia Tech's Fifth Symposium on Wireless Personal Communications*, pp. 18-1 - 18-12, June 1995.

Tsa69 S. Tsai, "Markov Characterization of the HF Channel," *IEEE Transactions on Communication Technology*, Vol. COM-17, No. 1, pp. 24-32, February 1969.

TsC95 Y. Tsai and J. Chang, "Using Spread Spectrum Techniques to Combat Multipath Interference in Mobile Random Accessed Networks," *IEEE Transactions on Communications*, Vol. 43, No. 2/3/4, pp. 329-337, February/March/April 1995.

Tur88 W. Turin, "Simulation of Error Sources in Digital Channels," *IEEE Journal on Selected Areas in Communications*, Vol. 6, No. 1, pp. 85-93, January 1988.

Tur90 W. Turin, *Performance Analysis of Digital Transmission Systems*, Computer Science Press, New York, 1990.

TuS93 W. Turin and M.M. Sondhi, "Modeling Error Sources in Digital Channels," *IEEE Journal on Selected Areas in Communications*, Vol. 11, No. 3, pp. 340-347, April 1993.

Vit85 A.J. Viterbi, "When Not to Spread Spectrum—a Sequel," *IEEE Communications Magazine*, Vol. 23, No. 4, pp. 12-17, April 1985.

WaM95 H.S. Wang and N. Moayeri, "Finite-State Markov Channel—A Useful Model for Radio Communication Channels," *IEEE Transactions on Vehicular Technology*, Vol. 44, No. 1, pp. 163-171, February 1995.

WaN92 J. Wang, M. Moeneclaey, and L.B. Milstein, "Predetection Diversity for CDMA Indoor Radio Communications," *Proceedings of Virginia Tech's Second Symposium on Wireless Personal Communications*, pp. 13-1 - 13-10, June 17-19, 1992.

WeC90 J.H. Wen and J.F. Chang, "The Effect of Multipath Interference on the Performance of Packet Radios," *IEEE Transactions on Communications*, Vol. 38, No. 6, pp. 740-743, June 1990.

WeW95 J. Weinmiller, H. Woesner, J. Ebert, and A. Wolisz, "Analyzing the RTS/CTS Mechanism in the DFWMAC Media Access Protocol for Wireless LANs," Presented at *IFIP TC6 Workshop Personal Wireless Communications (Wireless Local Access)*, Prague, Czech Rep, April 1995.

WoW95 H. Woesner, J. Weinmiller, J. Ebert, and A. Wolisz, "Modified Backoff Algorithms for DFWMAC's Distributed Coordination Function," 2. *ITG Fachtagung Mobile Kommunikation '95*, Neu-Ulm, Germany, September 1995.

WrS95 G.R. Wright and W.R. Stevens, *TCP/IP Illustrated, Volume 2 The Implementation*, Addison-Wesley Publishing Company, Reading, MA, 1995.

Appendix A. Simulation Modules

This appendix is a compilation of simulation modules that are not shown in Chapter 5. The modules are presented in alphabetical order with the first figure providing a hierarchical breakdown of the modules. Chapter 5 describes the modules.

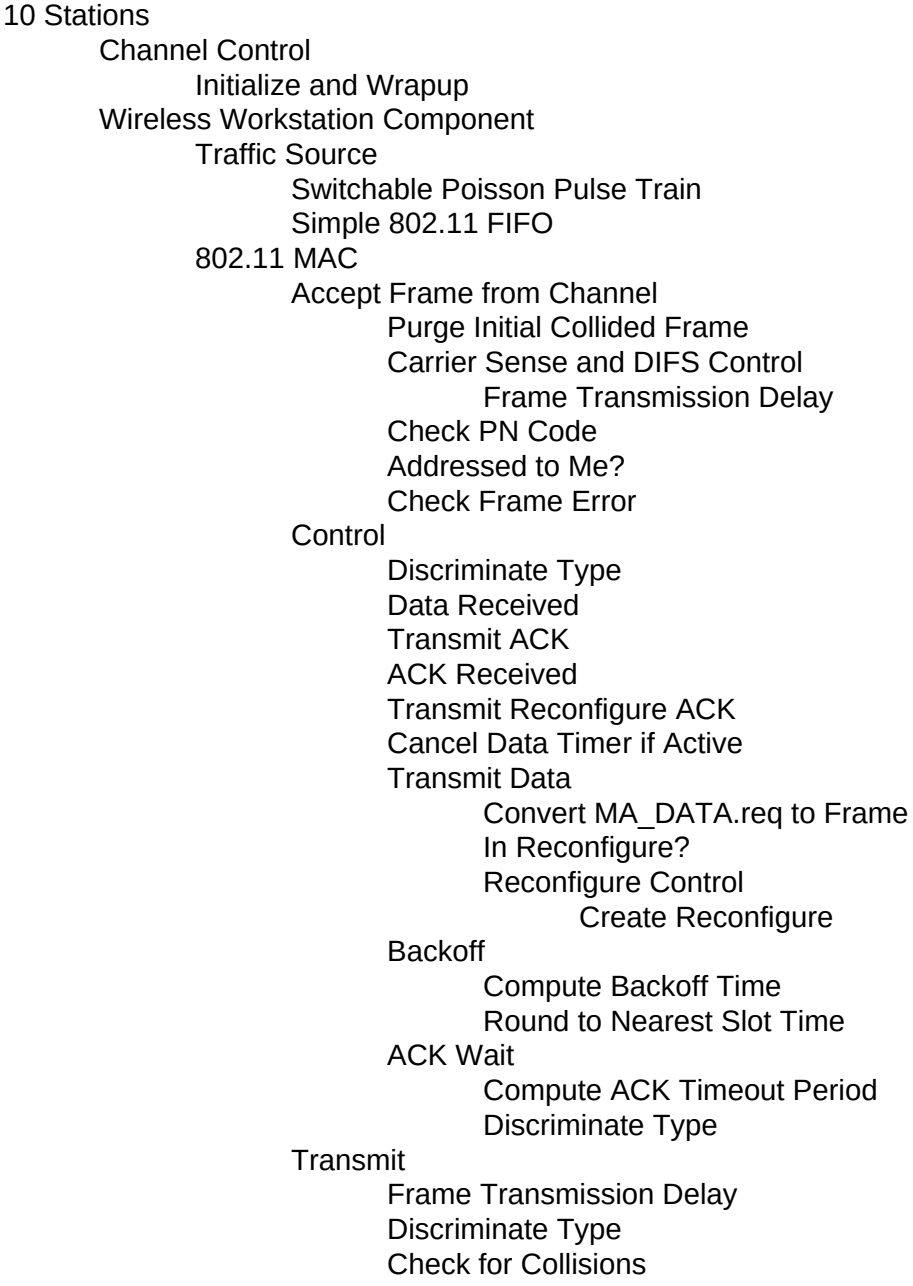

Figure A-1. Simulation Module Hierarchy

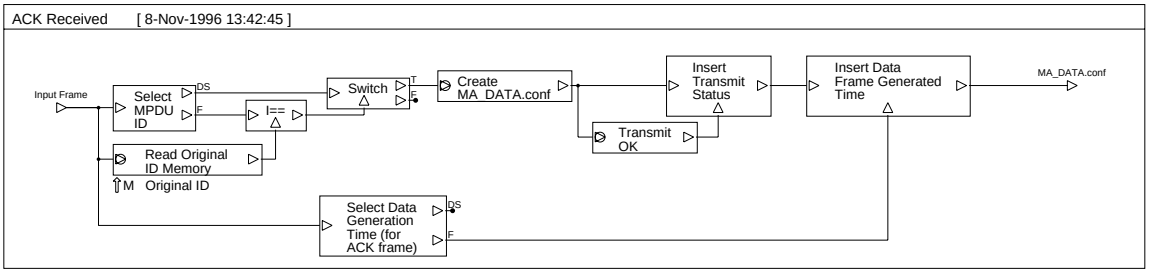

Figure A-2. *ACK Received* Simulation Module

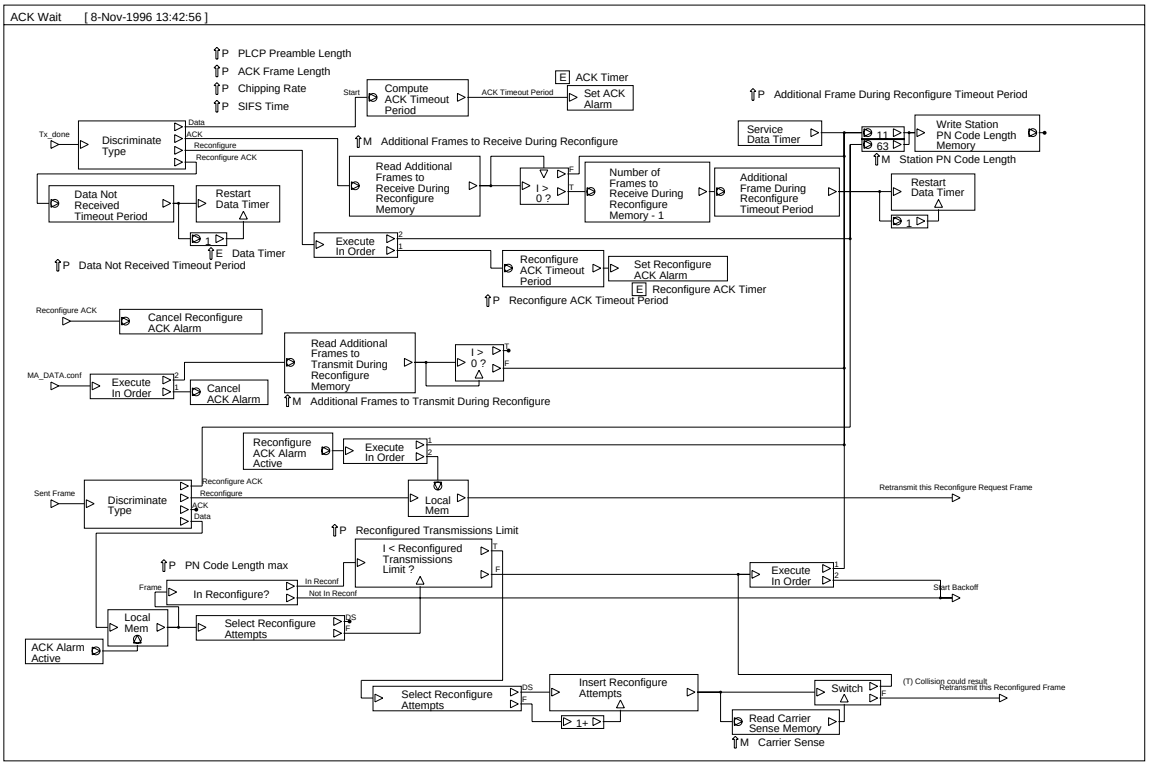

Figure A-3. *ACK Wait* Simulation Module

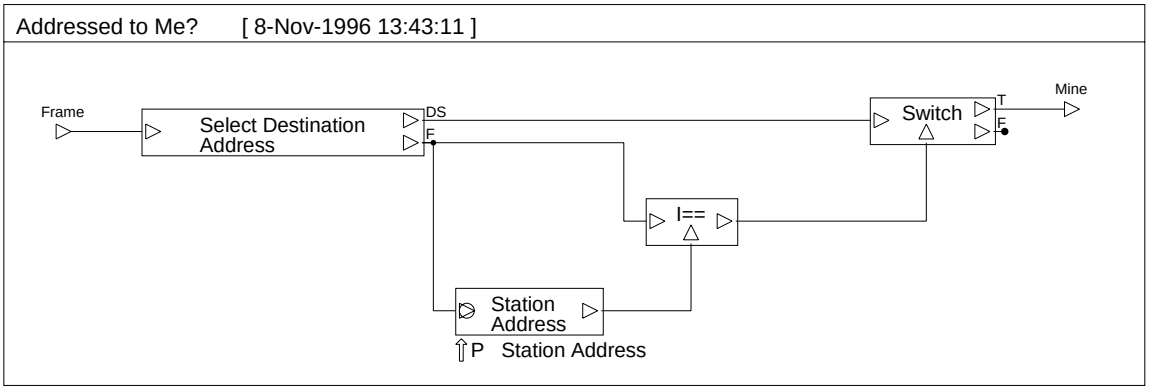

Figure A-4. *Addressed to Me?* Simulation Module

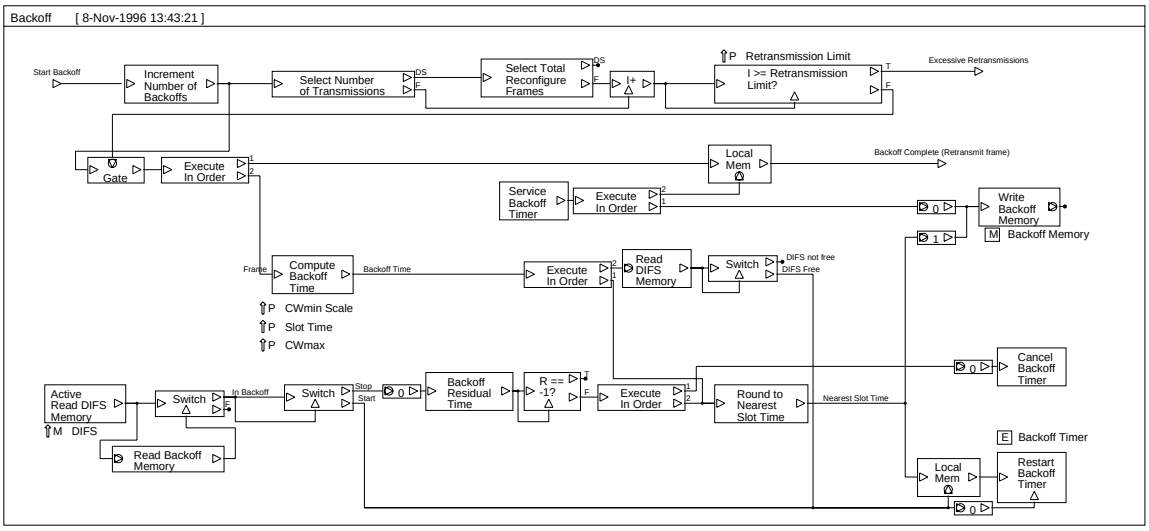

Figure A-5. *Backoff* Simulation Module

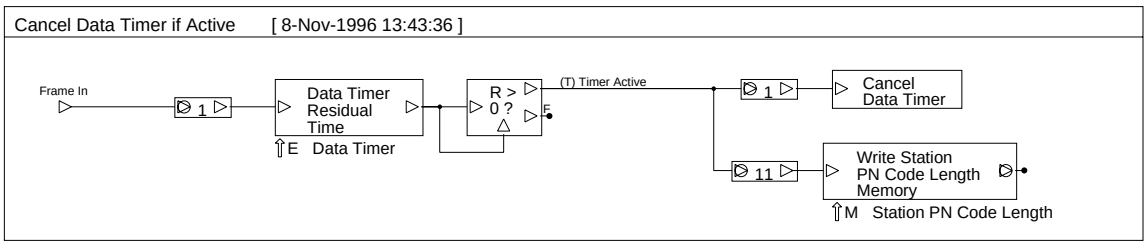

Figure A-6. *Cancel Data Timer if Active* Simulation Module

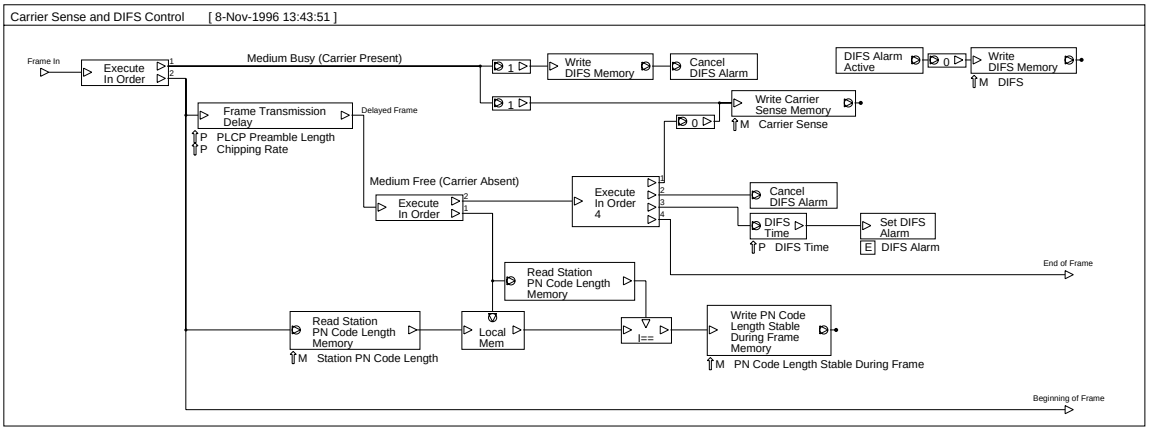

Figure A-7. *Carrier Sense and DIFS Control* Simulation Module

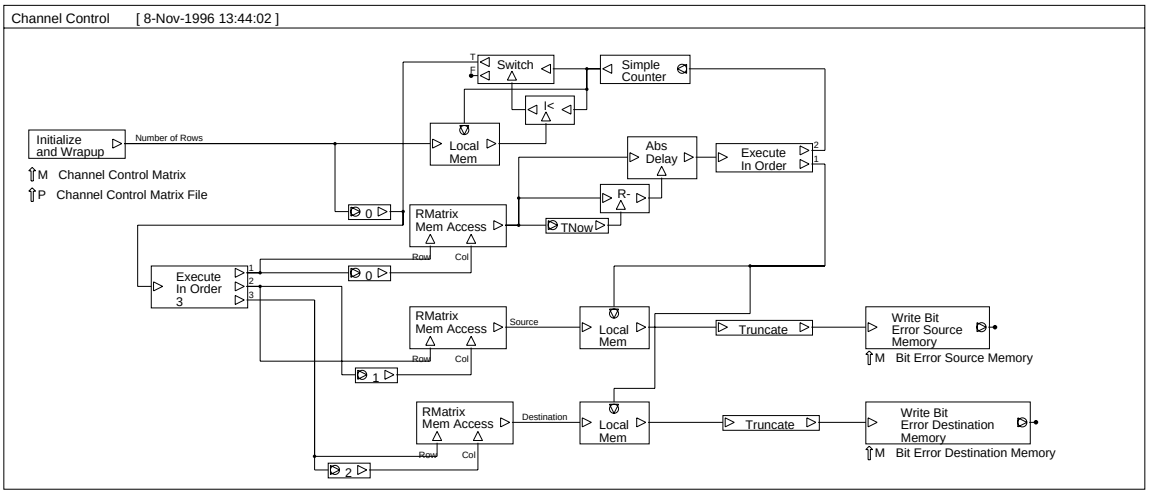

Figure A-8. *Channel Control* Simulation Module

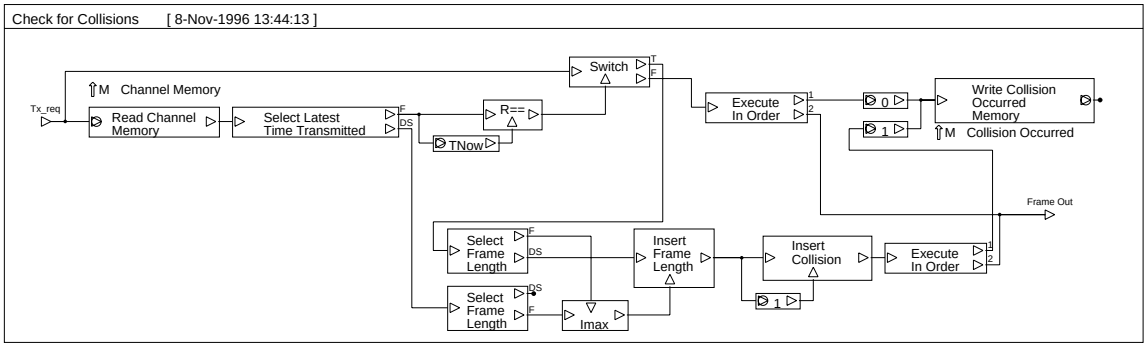

Figure A-9. *Check for Collisions* Simulation Module

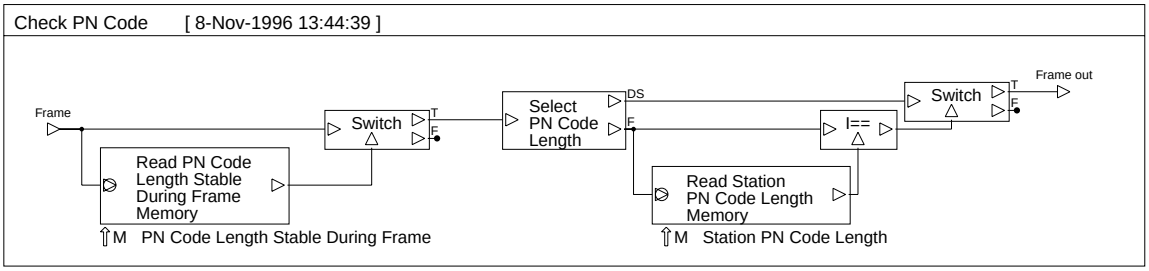

Figure A-10. *Check PN Code* Simulation Module

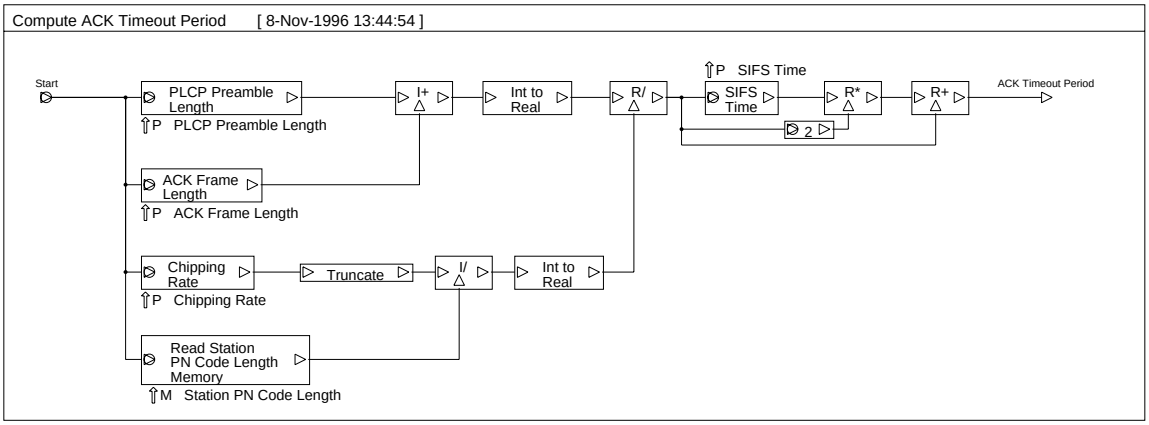

Figure A-11. *Compute ACK Timeout Period* Simulation Module

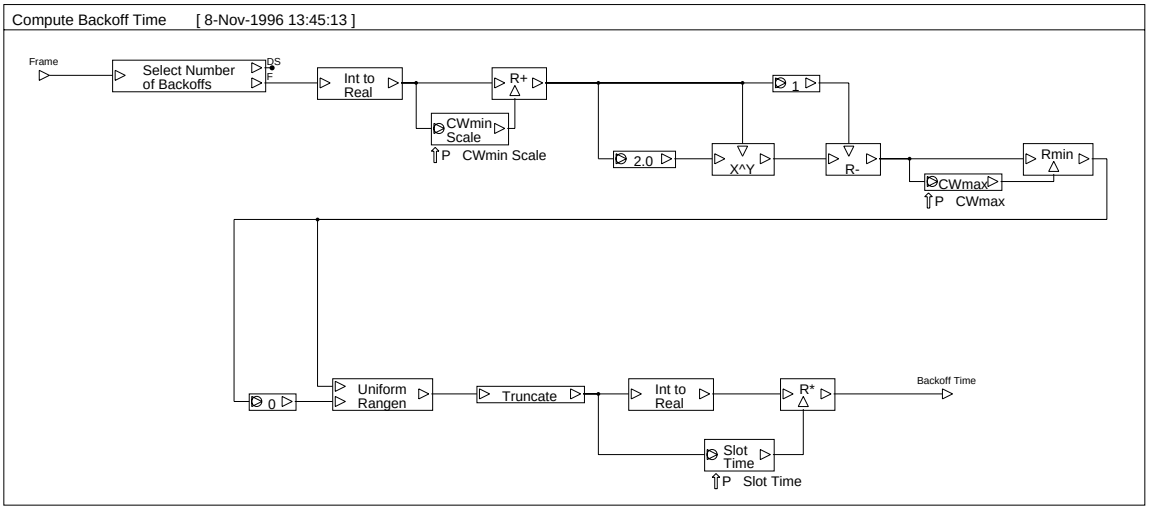

Figure A-12. *Compute Backoff Time* Simulation Module

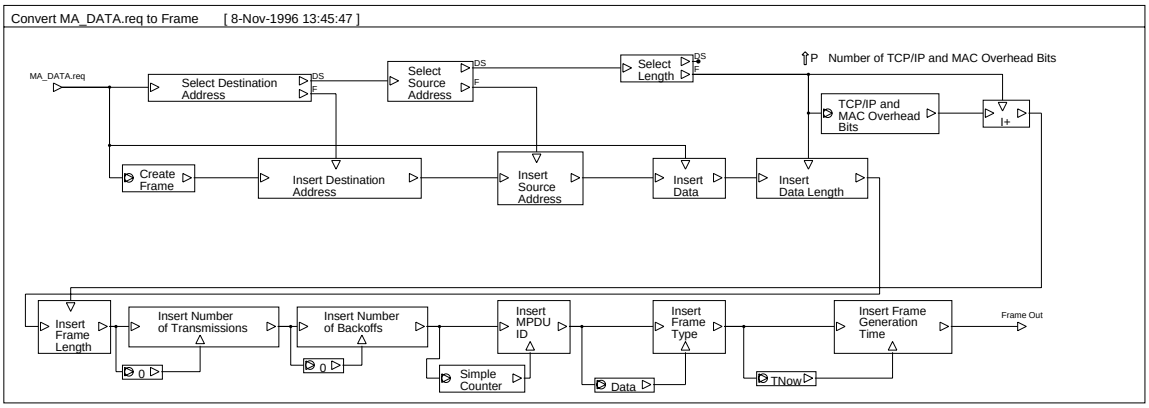

Figure A-13. *Convert MA_DATA.req to Frame* Simulation Module

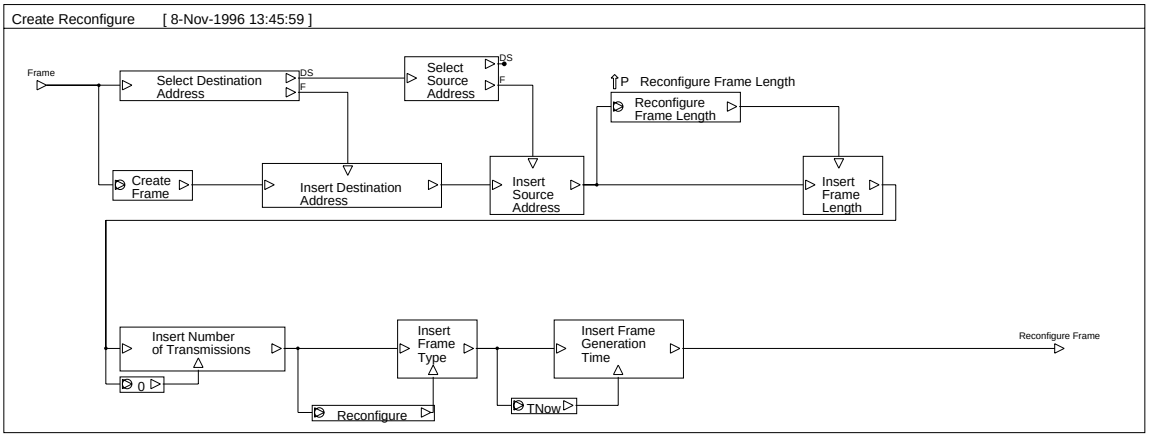

Figure A-14. *Create Reconfigure* Simulation Module

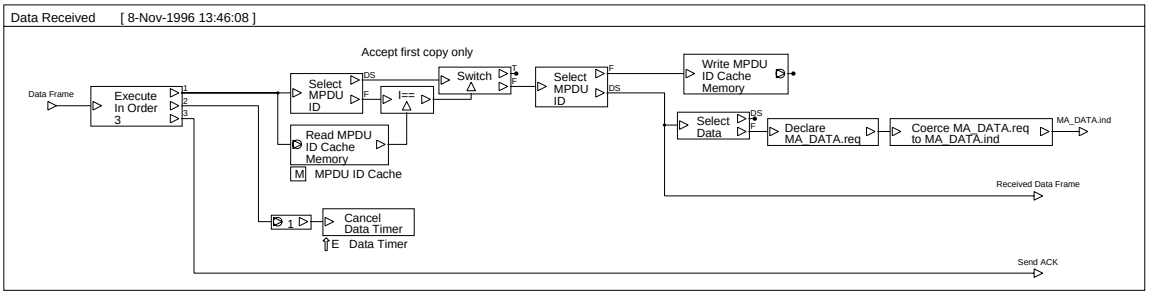

Figure A-15. *Data Received* Simulation Module

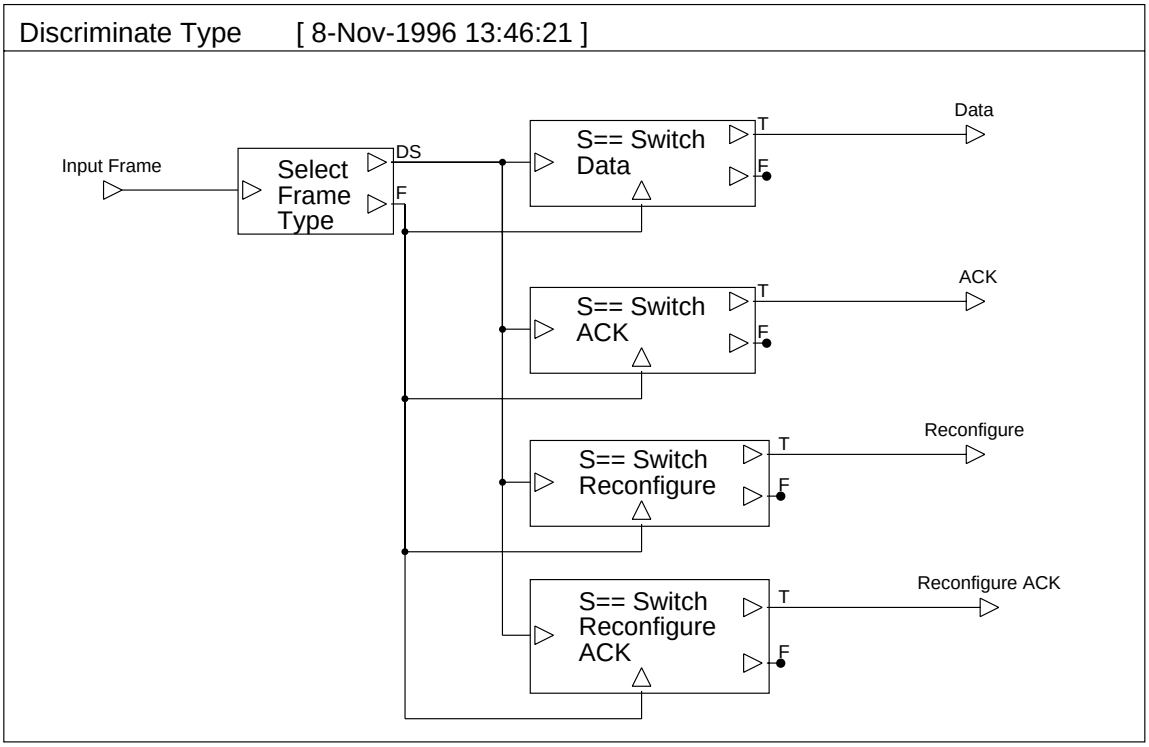

Figure A-16. *Discriminate Type* Simulation Module

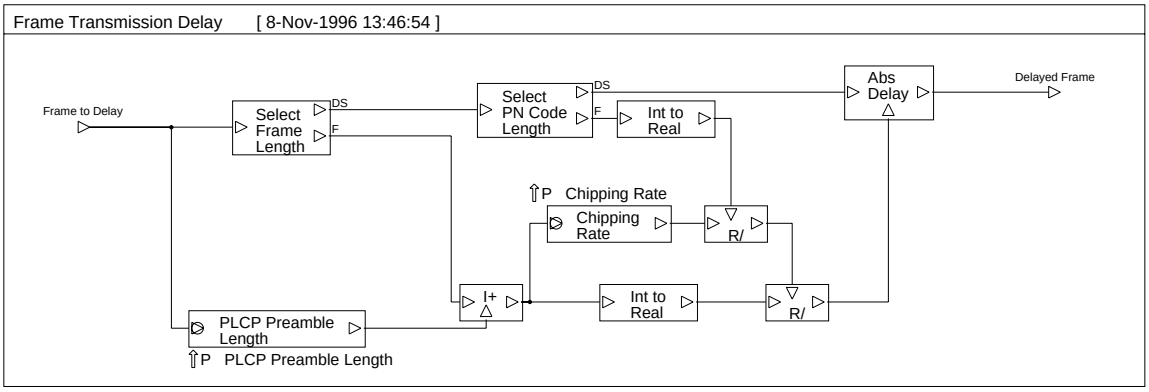

Figure A-17. *Frame Transmission Delay* Simulation Module

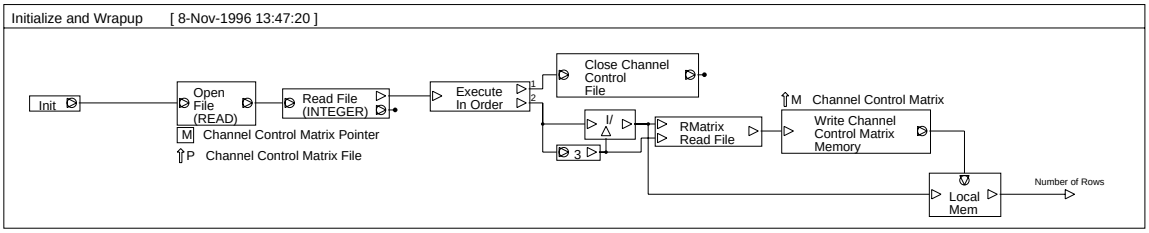

Figure A-18. ***Initialize and Wrapup*** **Simulation Module**

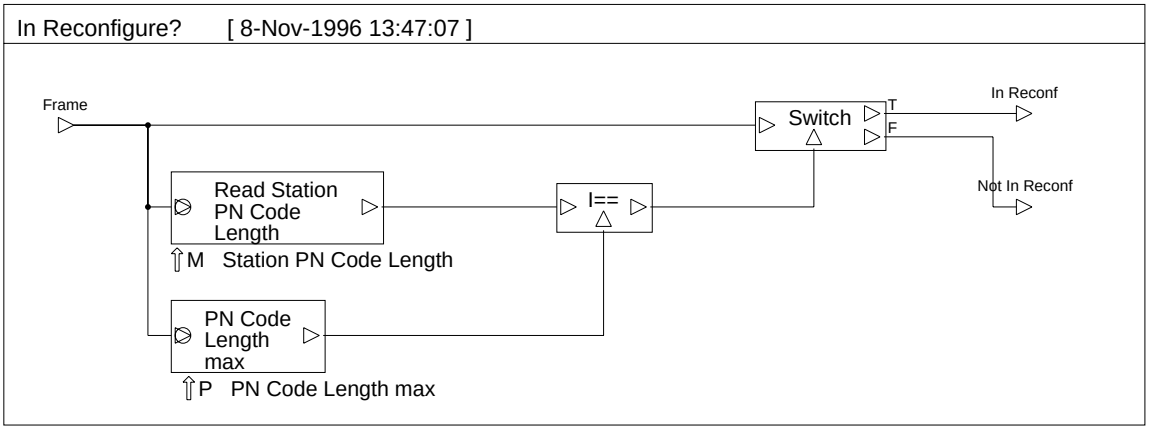

Figure A-19. ***In Reconfigure?*** **Simulation Module**

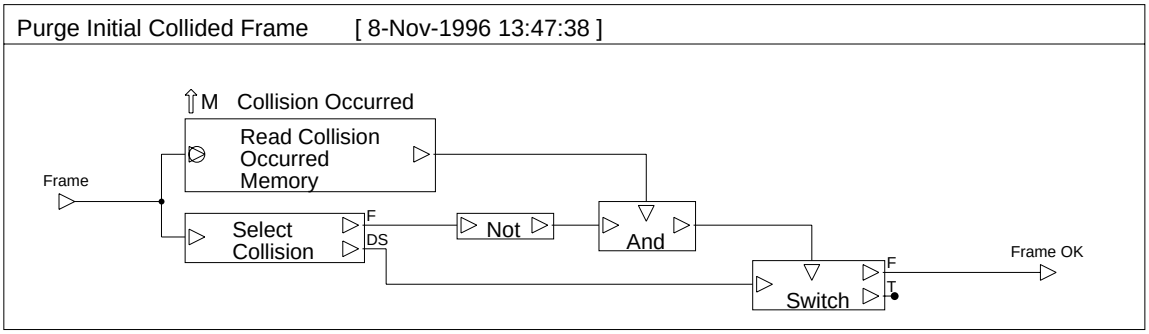

Figure A-20. ***Purge Initial Collided Frame*** **Simulation Module**

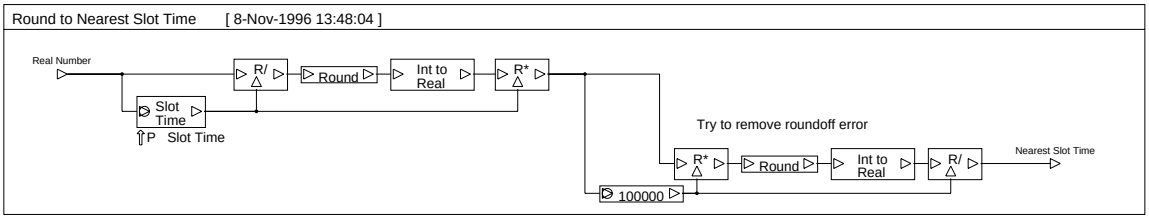

Figure A-21. ***Round to Nearest Slot Time*** **Simulation Module**

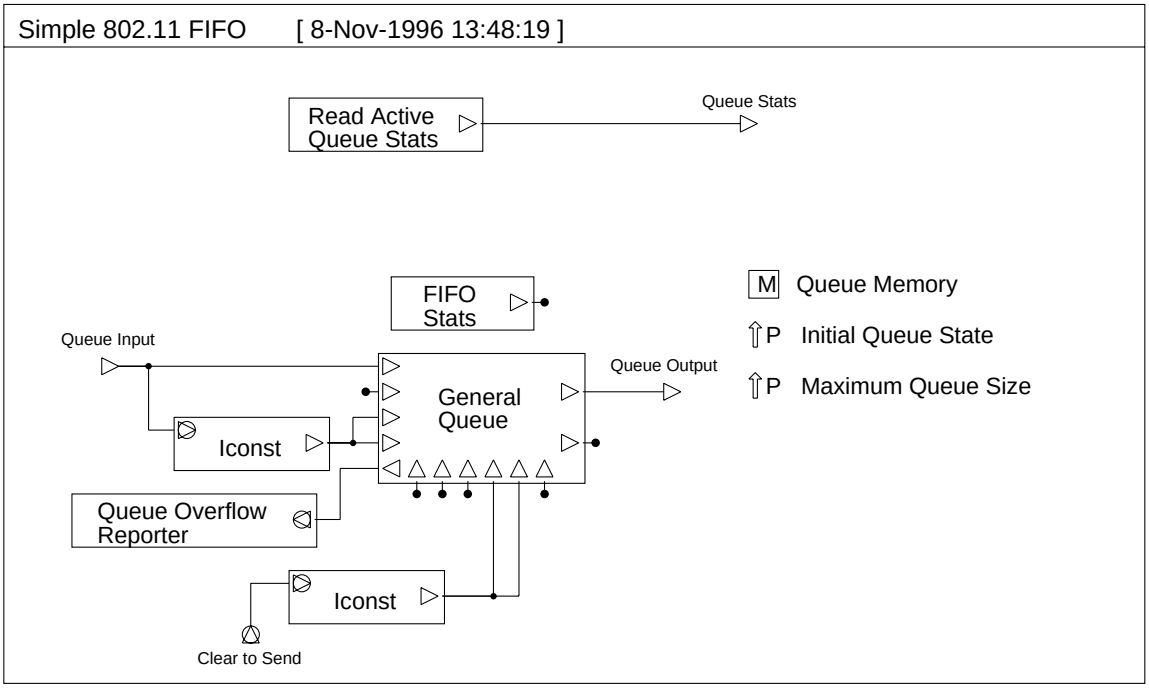

Figure A-22. ***Simple 802.11 FIFO*** **Simulation Module**

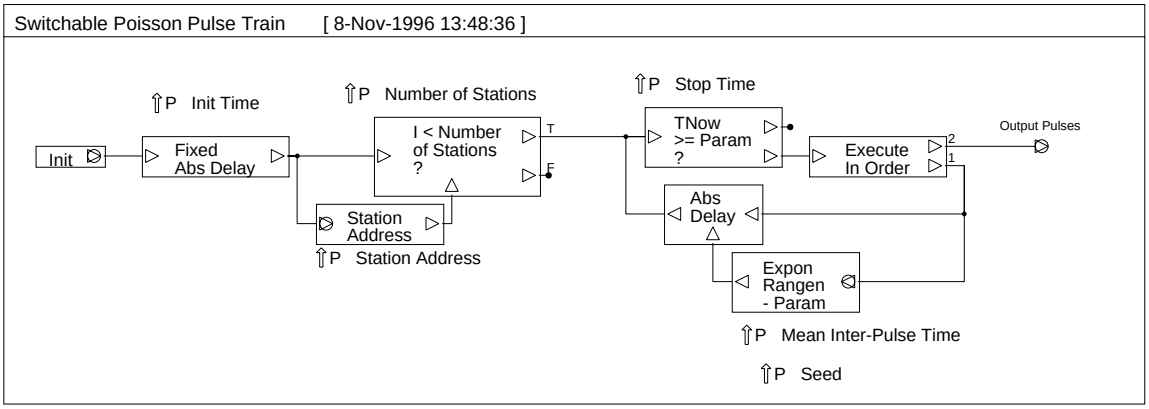

Figure A-23. ***Switchable Poisson Pulse Train*** **Simulation Module**

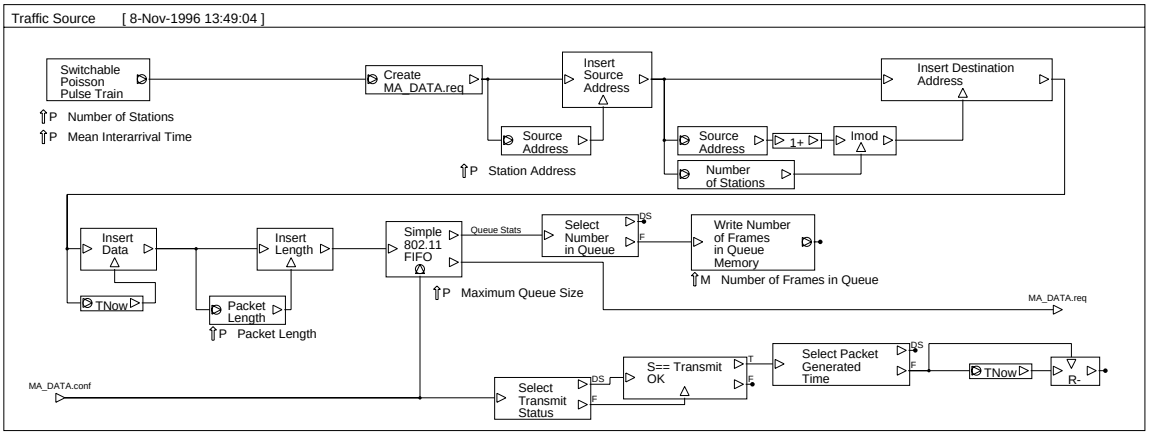

Figure A-24. ***Traffic Source*** **Simulation Module**

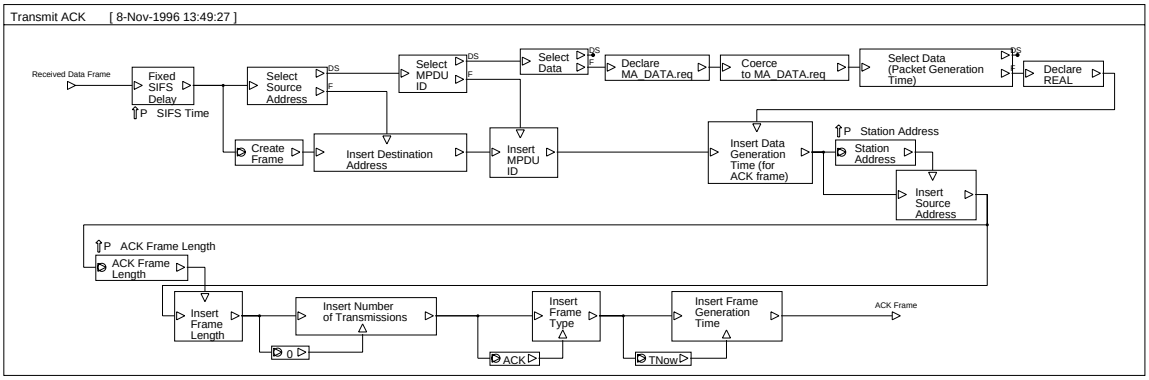

Figure A-25. *Transmit ACK* Simulation Module

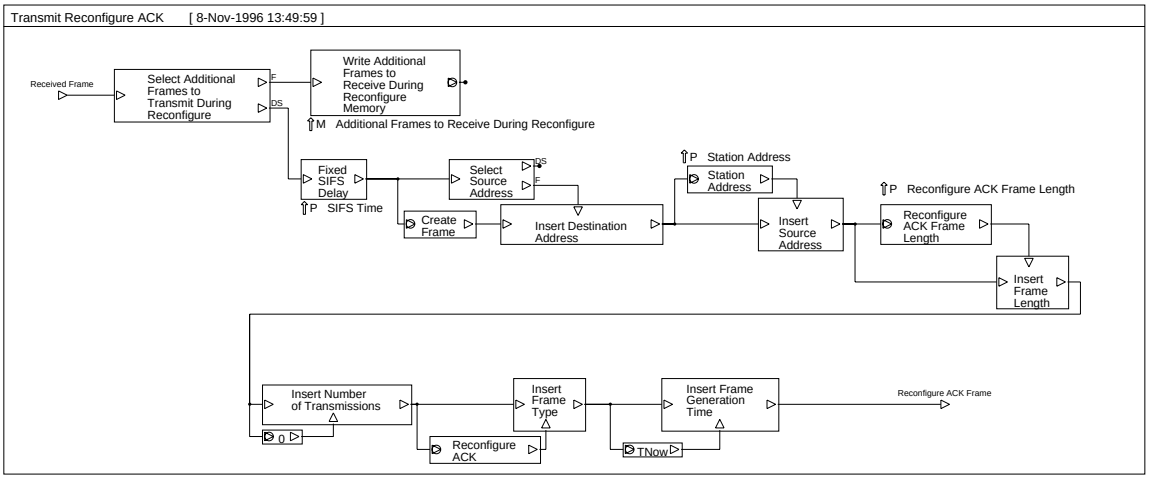

Figure A-26. *Transmit Reconfigure ACK* Simulation Module

Appendix B. Aggregate Throughput Surface Plots for Simulation Data

This appendix contains the aggregate throughput surface plots for data gathered on the 10 station simulation. An explanation of these plots is found in Chapter 6.

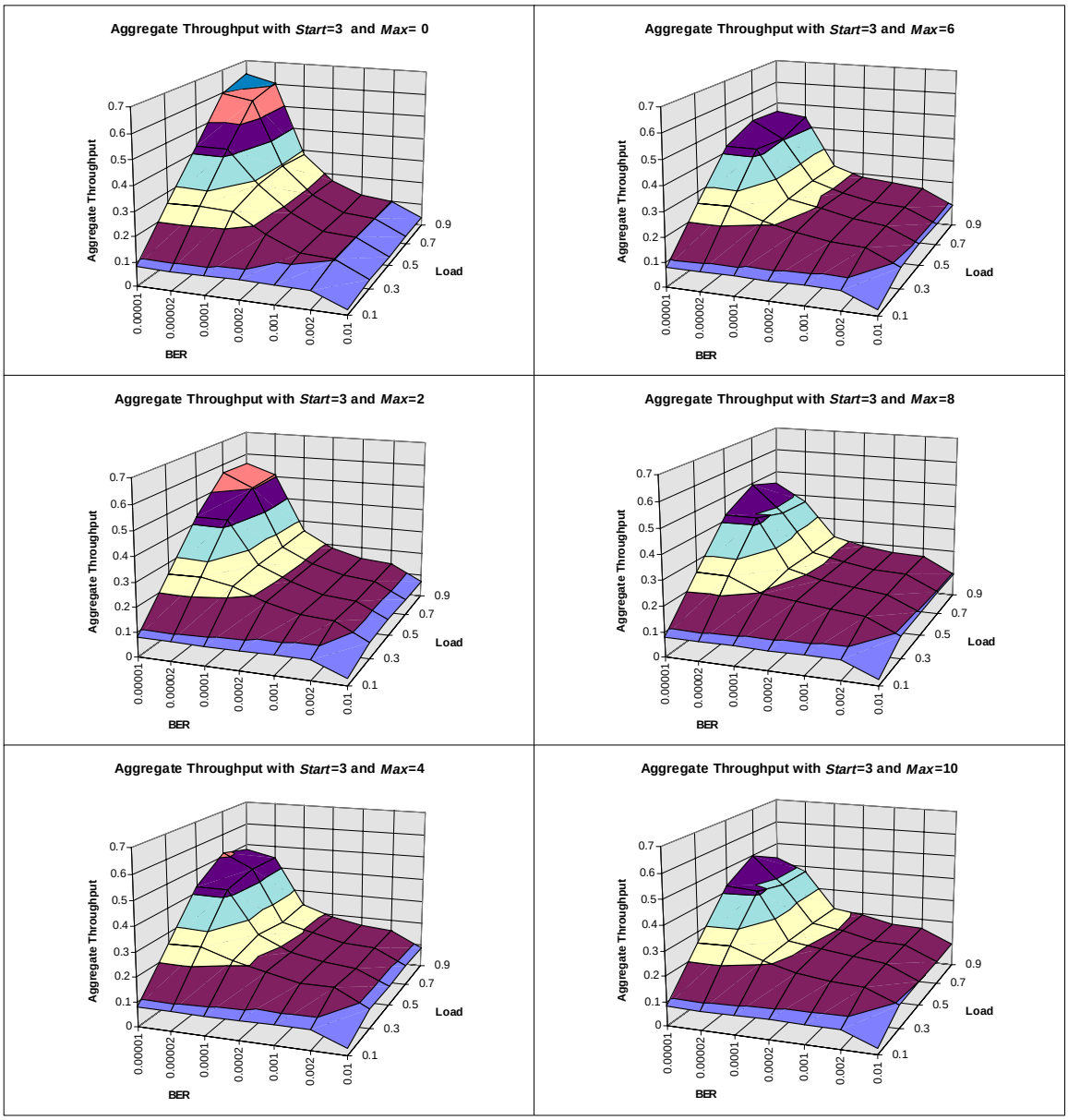

Figure B-1. Aggregate Throughput with *Start*=3 for 10 Station Simulation

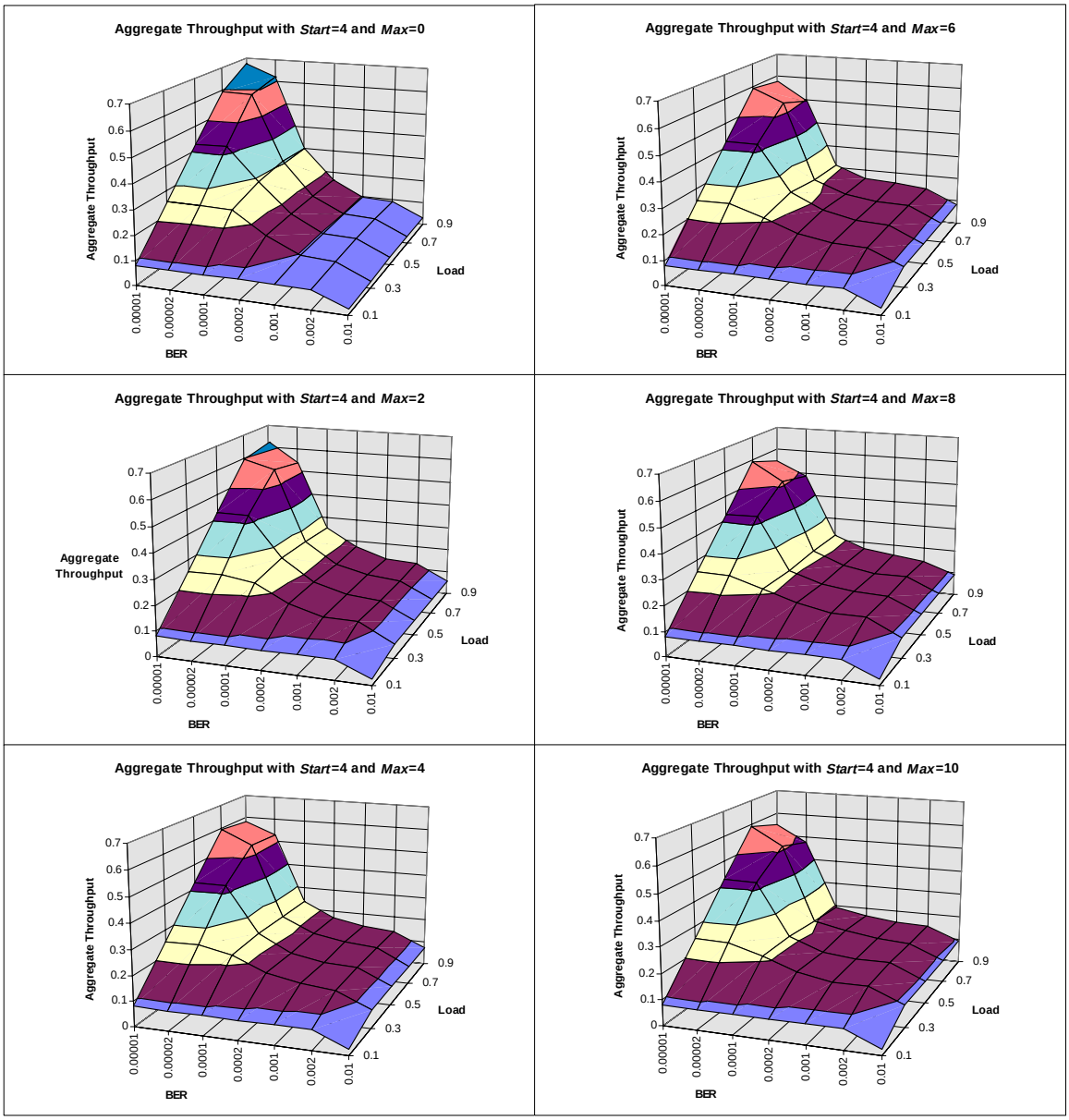

Figure B-2. Aggregate Throughput with *Start*=4 for 10 Station Simulation

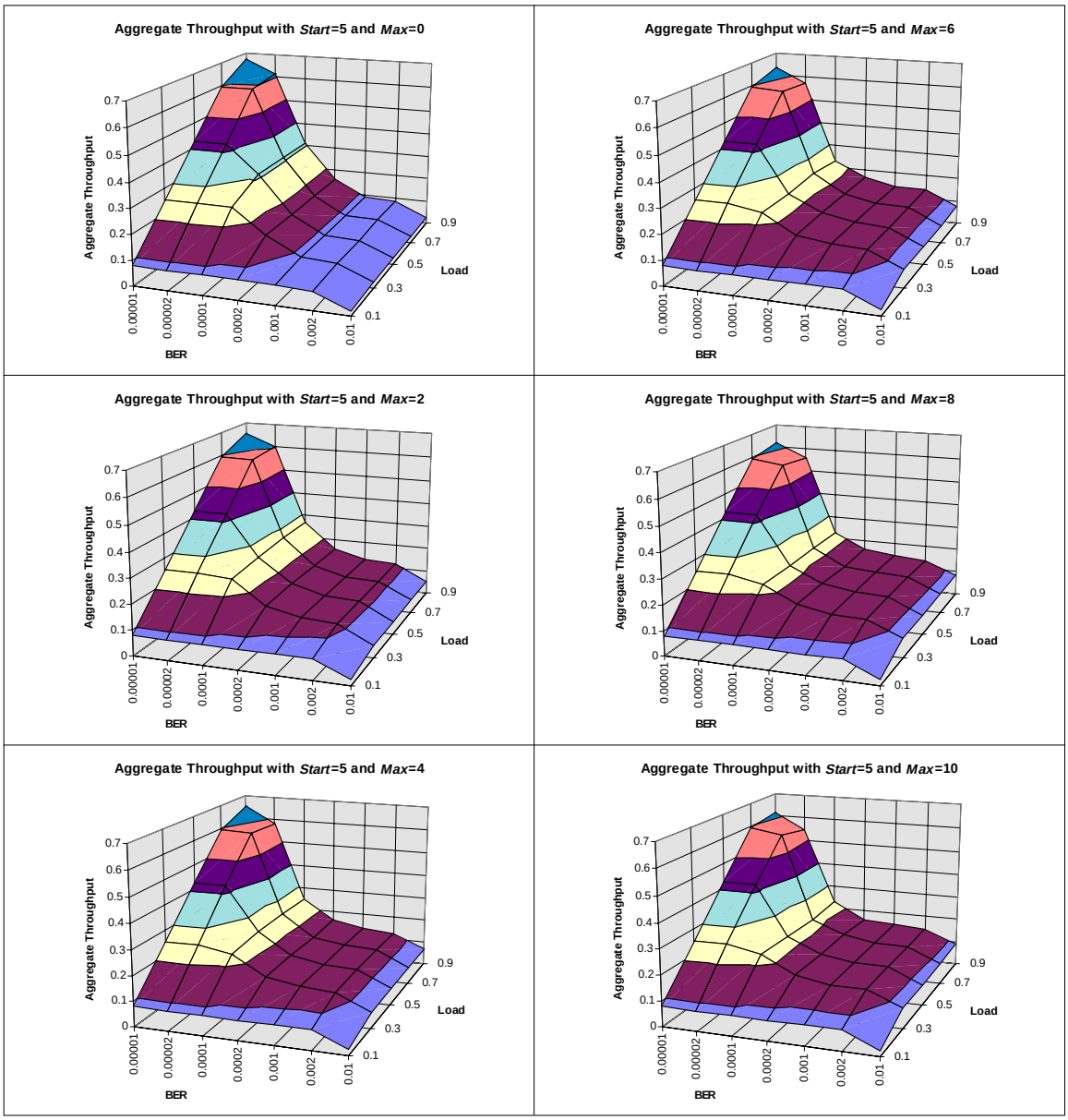

Figure B-3. Aggregate Throughput with *Start*=5 for 10 Station Simulation

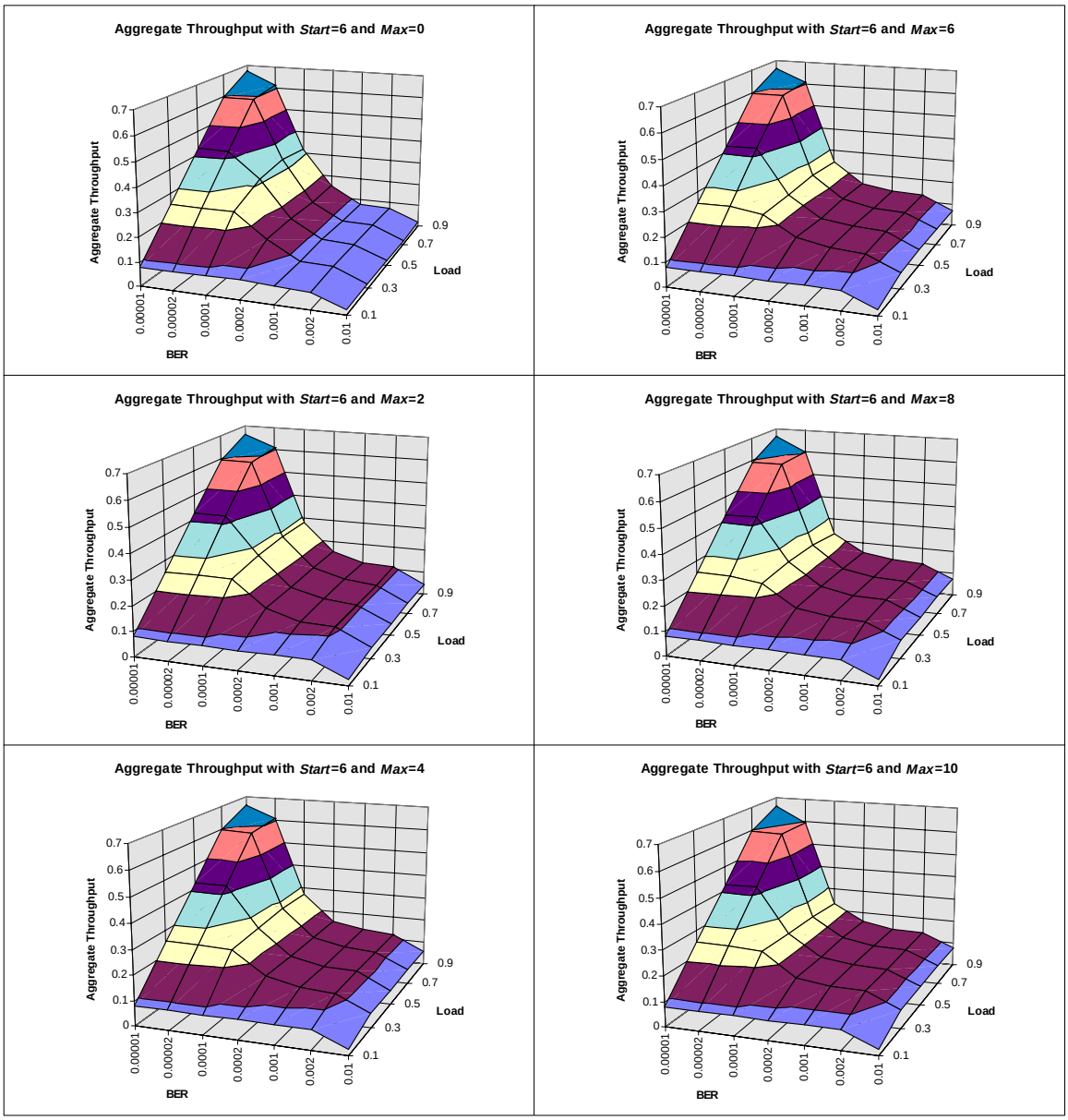

Figure B-4. Aggregate Throughput with *Start*=6 for 10 Station Simulation

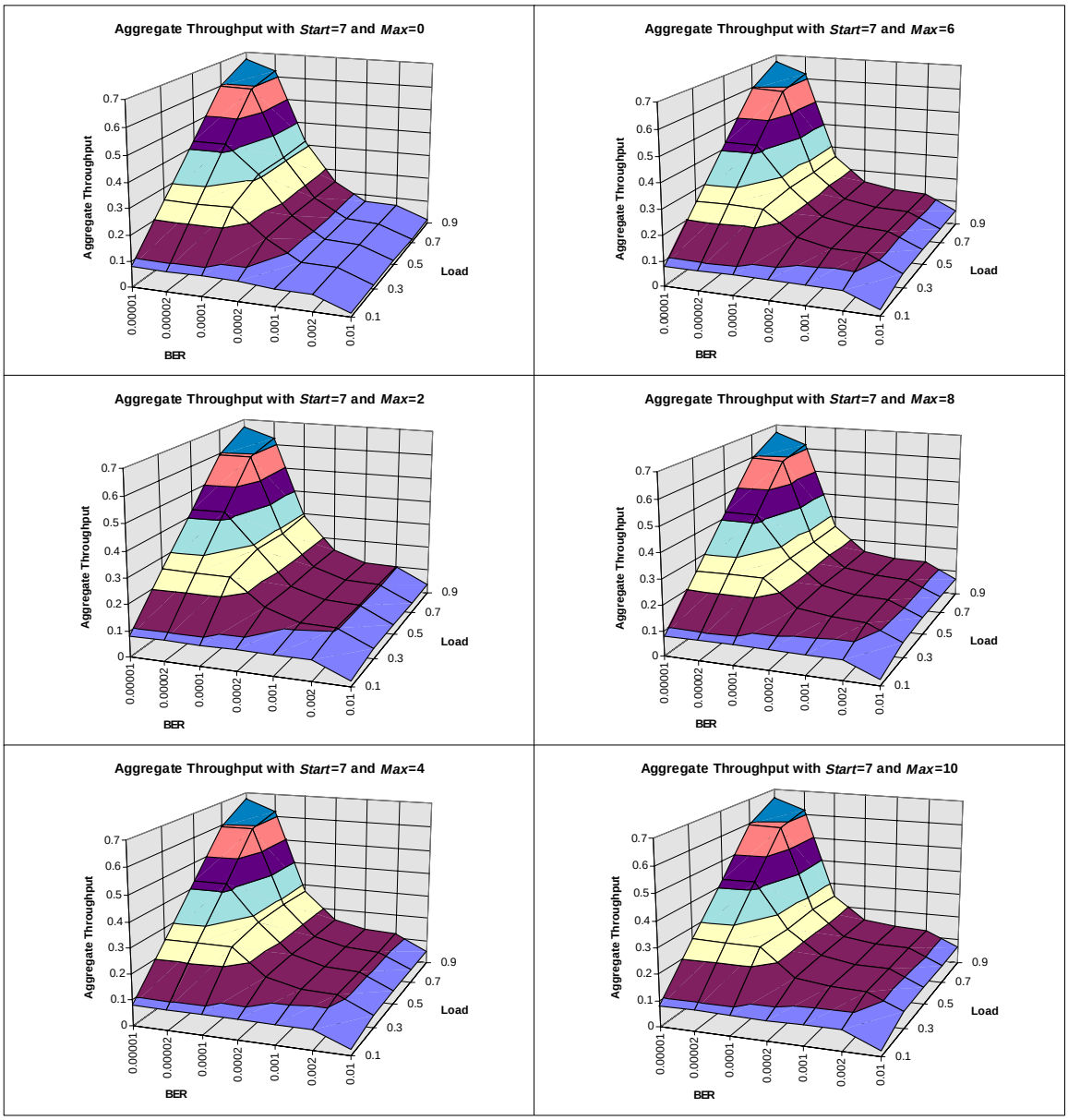

Figure B-5. Aggregate Throughput with *Start*=7 for 10 Station Simulation

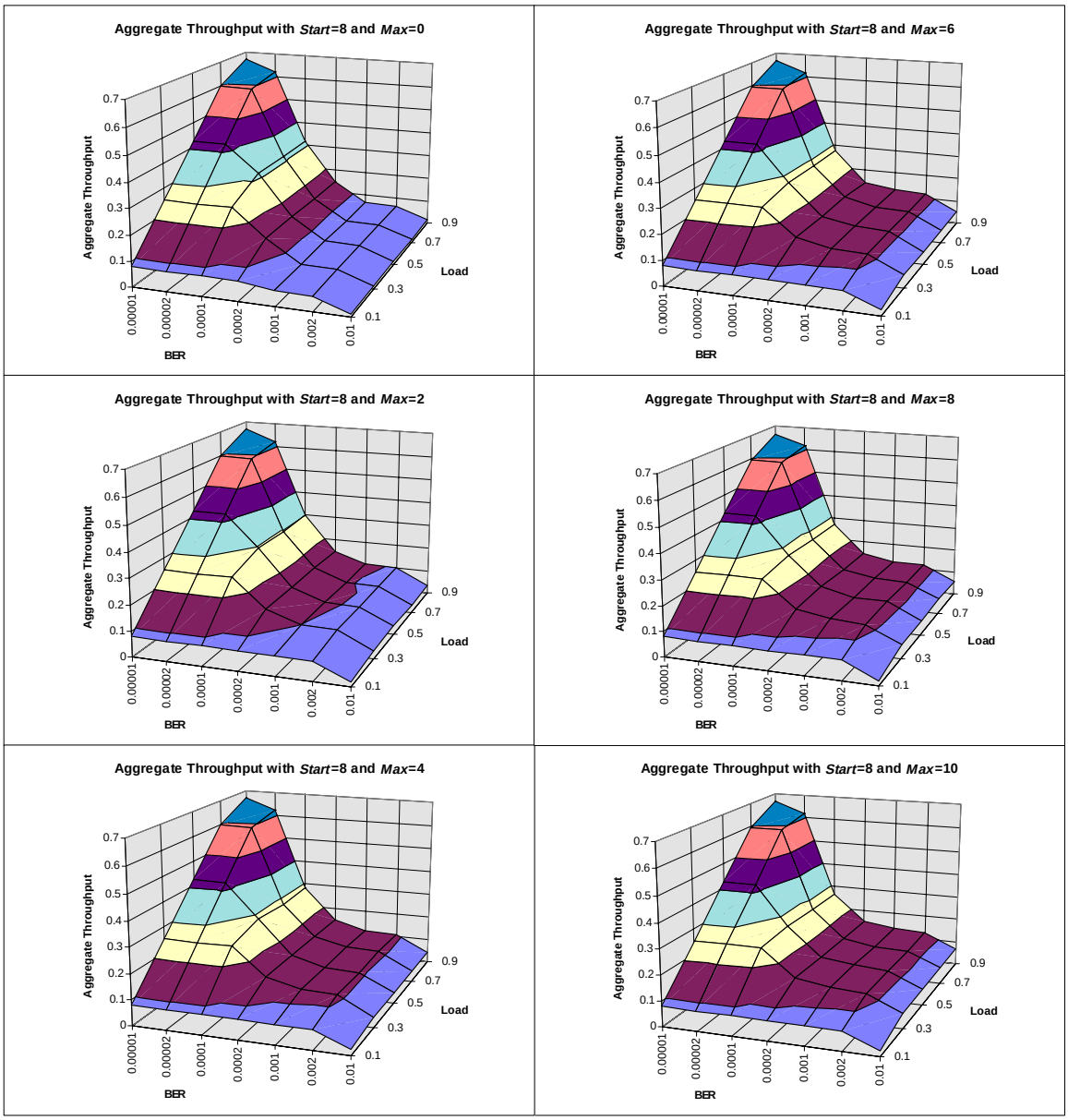

Figure B-6. Aggregate Throughput with *Start*=8 for 10 Station Simulation

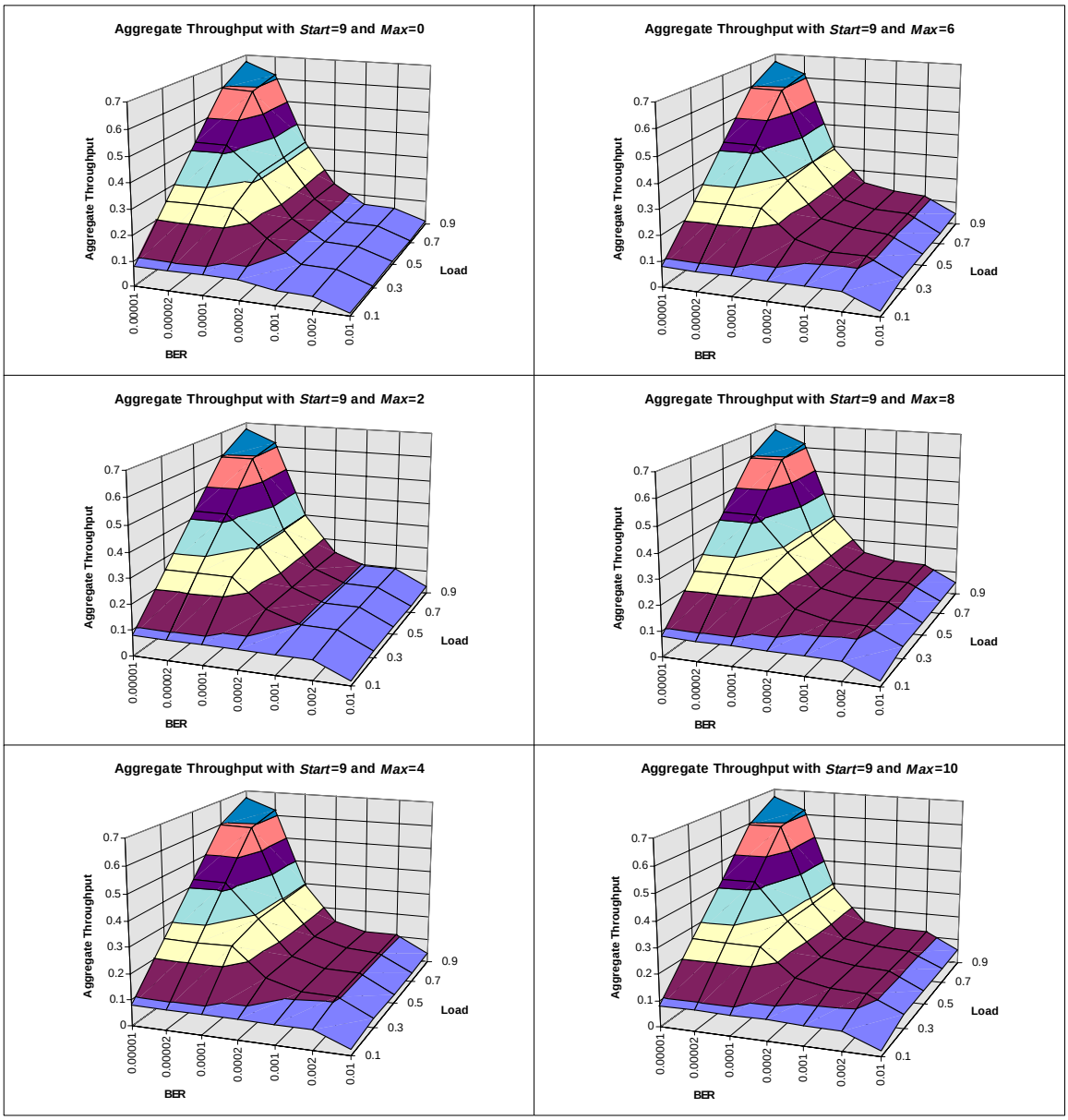

Figure B-7. Aggregate Throughput with *Start*=9 for 10 Station Simulation

Appendix C. Relative Confidence Interval Half Widths for 10 Station Simulation Data

This appendix contains all relative confidence interval half width tables for the 10 station simulation. An explanation of the these tables is found in Chapter 6.

Relative Confidence Interval Half Widths for *Start* = 3

Table C-1. Relative Confidence Interval Half Widths for *Start*=3 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.759136	2.814589	2.874153	2.757621	4.980026
0.3	3.859965	3.859965	2.482929	4.702612	2.619931	0.649541	1.904545
0.5	2.340103	2.365847	3.948412	3.816846	1.731569	1.257758	2.847065
0.7	2.118679	1.969644	2.725683	3.198647	1.514357	1.548119	3.103756
0.9	2.147969	2.336103	4.555127	3.928625	1.510161	1.349706	4.000359

Table C-2. Relative Confidence Interval Half Widths for *Start*=3 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.778577	3.01407	2.965898	2.9316	6.805012
0.3	3.859965	3.859965	2.108535	8.035825	6.072892	3.866819	3.919979
0.5	2.340103	2.374573	7.297989	5.461886	7.020377	2.078392	4.042493
0.7	3.717091	3.147277	7.204689	6.056361	7.021947	2.803749	4.413003
0.9	3.748645	7.145951	6.499947	5.358524	1.703721	3.323922	3.905573

Table C-3. Relative Confidence Interval Half Widths for *Start*=3 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.778577	2.917026	2.984793	3.043343	5.621137
0.3	3.859965	3.859965	5.114376	5.823101	2.589955	4.273885	3.648554
0.5	2.340103	2.659036	5.729706	4.651264	6.620524	5.037009	4.27904
0.7	3.158425	3.791254	7.121188	8.092392	2.10686	1.613114	6.683406
0.9	4.568712	4.99712	5.692519	6.490756	5.411549	7.407358	4.554035

Table C-4. Relative Confidence Interval Half Widths for *Start*=3 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.778577	2.917026	2.984793	3.043343	5.621137
0.3	3.859965	3.859965	6.498055	4.323437	2.510408	3.008262	5.327997
0.5	2.340103	2.330751	8.013547	5.234106	9.556902	3.633685	5.129038
0.7	6.78636	7.086877	14.45224	10.20967	6.429866	2.423073	5.532521
0.9	7.428794	10.67188	9.537314	5.187972	7.370083	7.820658	4.175959

Table C-5. Relative Confidence Interval Half Widths for *Start*=3 and *Max*=8

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.778577	2.917026	2.984793	3.043343	5.621137
0.3	3.859965	3.859965	4.554571	4.83327	8.459072	3.669317	4.152527
0.5	2.340103	2.586226	13.07463	8.460131	10.16414	4.568443	5.694352
0.7	13.39003	7.679799	5.270072	8.96046	7.90125	3.722904	4.611987
0.9	8.778862	8.070395	12.50739	8.596709	8.217704	5.942973	6.446516

Table C-6. Relative Confidence Interval Half Widths for *Start*=3 and *Max*=10

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.778577	2.917026	2.984793	3.043343	5.621137
0.3	3.859965	3.859965	5.756024	5.094023	4.779064	2.26519	5.194811
0.5	2.340103	4.053421	6.791783	12.22245	13.4744	2.733453	3.830938
0.7	10.64967	5.773842	14.35508	14.74743	3.472562	6.471935	5.145223
0.9	10.30831	10.79939	10.58084	12.5065	8.803811	5.476803	4.11463

Relative Confidence Interval Half Widths for *Start* = 4

Table C-7. Relative Confidence Interval Half Widths for *Start*=4 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.814589	2.88384	2.75566	2.352283	6.288714
0.3	3.859965	3.859965	3.316869	4.275399	3.161521	1.217793	1.72667
0.5	2.321731	2.376656	4.535007	3.908991	2.861081	1.299837	3.175531
0.7	2.125526	1.414776	2.11414	4.456779	1.608708	0.737008	4.000934
0.9	1.568329	2.163898	2.709647	2.325972	2.404937	1.350242	3.956896

Table C-8. Relative Confidence Interval Half Widths for *Start*=4 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.801175	2.978896	3.017628	3.049353	4.540981
0.3	3.859965	3.864639	2.597543	2.710608	2.875773	3.666044	4.547579
0.5	2.321731	2.330036	2.913813	6.652986	2.980044	3.703629	7.225539
0.7	2.131992	2.782787	8.529635	5.382488	5.164806	1.806763	4.434473
0.9	1.251374	2.155907	2.757469	6.157392	1.869468	3.761592	6.166803

Table C-9. Relative Confidence Interval Half Widths for *Start*=4 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.801175	2.965898	3.032174	2.959709	4.809023
0.3	3.859965	3.864639	5.333088	6.365509	5.259227	1.287761	7.399607
0.5	2.321731	2.330036	6.703417	4.218395	5.328704	1.24632	6.285699
0.7	1.967189	5.859956	2.969669	3.362613	4.991838	2.831798	5.689876
0.9	3.638196	3.740624	13.99717	4.791446	3.168552	3.055798	5.849728

Table C-10. Relative Confidence Interval Half Widths for *Start*=4 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.801175	2.965898	3.0293	2.959709	4.809023
0.3	3.859965	3.864639	3.423726	5.988258	5.11769	2.603499	7.046067
0.5	2.321731	2.330036	8.949959	8.57686	3.919367	1.786812	5.77857
0.7	1.611373	4.185778	10.44405	9.463276	3.727013	4.083838	8.067411
0.9	4.727507	4.732849	4.435926	5.492143	2.179822	3.970911	4.800825

Table C-11. Relative Confidence Interval Half Widths for *Start*=4 and *Max*=8

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.801175	2.965898	3.0293	2.959709	4.809023
0.3	3.859965	3.864639	5.575443	7.163171	7.278189	3.889704	8.441391
0.5	2.321731	2.330036	7.480881	2.340939	3.720401	4.060817	5.827667
0.7	1.505227	5.758222	6.972248	6.480989	8.616833	4.471792	5.989141
0.9	1.461226	5.307732	10.45536	10.3542	8.064581	2.756253	5.240807

Table C-12. Relative Confidence Interval Half Widths for *Start*=4 and *Max*=10

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.801175	2.965898	3.0293	2.959709	4.809023
0.3	3.859965	3.864639	9.312923	4.243154	6.778111	4.862808	6.193547
0.5	2.321731	2.330036	8.197338	5.23582	4.396652	3.94735	5.220606
0.7	1.345051	5.803168	3.479506	9.534504	7.291608	6.157577	4.500679
0.9	7.683147	3.907211	5.331803	4.55522	5.89905	4.47111	4.491771

Relative Confidence Interval Half Widths for *Start* = 5

Table C-13. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.830094	2.862384	2.326261	0.541654	6.738629
0.3	3.859965	3.864639	3.772167	3.533444	2.840249	1.72463	2.532407
0.5	2.321731	2.345041	1.896652	3.4447	3.776904	0.874944	2.820532
0.7	2.114586	1.748725	5.007351	3.268819	1.397826	1.720047	3.205117
0.9	0.452653	2.202526	2.955798	2.596573	4.718796	1.244865	1.318272

Table C-14. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.804292	2.972986	3.011807	2.976282	4.766304
0.3	3.859965	3.864639	2.842138	4.246187	1.260641	2.949838	7.473633
0.5	2.321731	2.345041	3.292188	9.76585	4.732108	1.070428	5.24917
0.7	2.113806	1.408239	4.897033	3.586715	0.448833	1.451447	7.69173
0.9	1.059376	2.483507	5.325782	2.257017	3.681462	2.060571	5.048375

Table C-15. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.804292	3.023455	2.994211	3.061205	6.532174
0.3	3.859965	3.864639	1.994793	4.663689	4.621153	2.520571	6.88297
0.5	2.321731	2.345041	5.838745	5.086848	3.408901	2.539437	7.374602
0.7	2.120178	2.147596	3.346965	3.147225	4.58803	4.759958	5.484312
0.9	2.166386	5.803446	6.190537	6.147846	2.563317	3.052493	8.507319

Table C-16. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.804292	3.023455	2.994211	3.045432	6.532174
0.3	3.859965	3.864639	2.910448	3.662455	3.905667	6.522376	7.719049
0.5	2.321731	2.345041	5.033351	8.28736	3.670056	3.05367	8.764167
0.7	2.162536	2.765611	6.389558	5.089841	3.618376	1.815118	7.604177
0.9	3.349329	3.550318	5.797921	5.39132	5.703672	4.414164	6.450789

Table C-17. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=8

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.804292	3.023455	2.994211	3.045432	6.532174
0.3	3.859965	3.864639	4.998148	2.69483	7.3752	2.328029	3.816877
0.5	2.321731	2.345041	7.814582	6.426706	6.12914	3.38587	6.796803
0.7	2.136272	2.298311	8.280093	5.891138	2.942764	5.219747	7.275011
0.9	1.964711	2.076219	7.829724	5.635541	8.563282	4.941764	6.967088

Table C-18. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=10

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.804292	3.023455	2.994211	3.045432	6.532174
0.3	3.859965	3.864639	4.243095	5.310385	5.41321	3.996614	4.850222
0.5	2.321731	2.345041	4.381408	6.842353	4.345282	1.657768	5.127148
0.7	2.136272	3.206433	4.53917	5.470042	4.951184	3.633014	8.870105
0.9	2.363217	4.608188	5.96039	2.860007	5.504583	1.842625	4.411704

Relative Confidence Interval Half Widths for *Start* = 6

Table C-19. Relative Confidence Interval Half Widths for *Start*=6 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.804292	2.901276	1.557484	1.64412	2.791967
0.3	3.859965	3.864639	3.592024	1.288311	2.393757	1.312839	5.626032
0.5	2.321731	2.345041	1.008601	4.983868	1.787611	2.134341	8.596428
0.7	2.114586	1.855919	2.323473	2.00185	2.075088	1.376452	5.16014
0.9	1.034079	0.829925	4.617777	2.130271	2.923624	0.964892	4.496368

Table C-20. Relative Confidence Interval Half Widths for *Start*=6 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.830094	2.949294	2.969885	3.297092	10.55632
0.3	3.859965	3.864639	3.326138	1.591471	3.21338	1.461529	7.133725
0.5	2.321731	2.345041	1.920798	2.031952	3.218891	1.741147	6.825886
0.7	2.114586	1.474489	3.604649	2.584386	3.869429	1.19593	7.739739
0.9	1.254445	1.900957	6.511831	2.333156	2.763749	1.384785	10.8166

Table C-21. Relative Confidence Interval Half Widths for *Start*=6 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.830094	2.949294	2.87604	2.725321	7.885865
0.3	3.859965	3.864639	2.36527	2.134986	3.346165	2.858062	7.653034
0.5	2.321731	2.345041	2.653021	3.886331	4.334245	1.968134	9.309533
0.7	2.114586	1.281324	5.42919	2.754825	2.278793	2.894449	9.899042
0.9	1.60077	2.239438	7.393733	4.859636	5.265545	2.083096	9.226804

Table C-22. Relative Confidence Interval Half Widths for *Start*=6 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.830094	2.949294	2.984766	2.773065	7.885865
0.3	3.859965	3.864639	1.963646	6.5597	3.821529	2.931601	7.495676
0.5	2.321731	2.345041	1.868154	2.799136	3.991319	3.573372	6.873905
0.7	2.114586	1.741566	6.055463	4.605625	1.537442	2.261776	5.95025
0.9	1.605768	2.043987	5.403329	3.545372	2.273053	4.208128	7.541184

Table C-23. Relative Confidence Interval Half Widths for *Start*=6 and *Max*=8

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.830094	2.949294	2.984766	2.773065	7.885865
0.3	3.859965	3.864639	3.099955	2.910921	3.58056	5.850243	5.31549
0.5	2.321731	2.345041	2.8706	3.936263	4.019832	1.219218	8.345962
0.7	2.114586	1.439144	6.690157	2.504727	0.840942	2.839452	4.274318
0.9	1.804708	2.522968	3.611957	4.442239	4.270488	2.001589	5.287042

Table C-24. Relative Confidence Interval Half Widths for *Start*=6 and *Max*=10

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.830094	2.949294	2.984766	2.773065	7.885865
0.3	3.859965	3.864639	2.472834	1.383297	4.301586	5.585779	7.080173
0.5	2.321731	2.345041	7.354159	4.77222	3.185499	1.197545	9.598502
0.7	2.114586	1.439144	3.883803	5.5118	2.267122	3.351212	5.452613
0.9	2.547779	3.862191	3.295967	6.465988	5.704388	4.741999	5.932548

Relative Confidence Interval Half Widths for *Start* = 7

Table C-25. Relative Confidence Interval Half Widths for *Start*=7 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.815971	2.901951	1.44624	1.646673	11.22946
0.3	3.859965	3.864639	3.881582	3.023042	1.17308	0.969081	8.358278
0.5	2.321731	2.345041	2.818698	2.465519	3.174641	0.975027	7.345754
0.7	2.114586	1.954401	4.767097	4.076817	3.818645	1.290811	5.521646
0.9	1.282669	1.635829	3.407894	2.824173	2.619073	0.723551	7.223587

Table C-26. Relative Confidence Interval Half Widths for *Start*=7 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.833184	2.901951	3.061444	3.19866	11.61594
0.3	3.859965	3.864639	3.835067	2.556218	1.650705	2.13567	9.29317
0.5	2.321731	2.345041	4.632703	2.116752	3.298242	2.202192	8.467718
0.7	2.114586	1.820795	2.291606	1.649039	1.347662	2.182531	8.535611
0.9	0.867084	1.919039	5.540824	1.490028	1.38618	1.125078	5.946245

Table C-27. Relative Confidence Interval Half Widths for *Start*=7 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.833184	2.927135	3.00014	3.079761	7.007801
0.3	3.859965	3.864639	3.726165	5.732202	2.125755	2.448131	8.035934
0.5	2.321731	2.345041	4.108406	1.346229	4.49967	1.899819	12.68373
0.7	2.114586	1.877277	5.798122	3.229141	4.281353	2.618214	6.624101
0.9	0.57619	1.997786	7.047399	3.148566	1.667522	1.082589	8.511274

Table C-28. Relative Confidence Interval Half Widths for *Start*=7 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.833184	2.927135	3.05314	2.656414	10.415
0.3	3.859965	3.864639	3.126755	3.635085	4.722646	2.300398	7.779568
0.5	2.321731	2.345041	3.702457	4.190298	3.988205	1.488869	12.62044
0.7	2.114586	1.765572	4.120133	2.428647	2.737039	3.518411	8.130701
0.9	0.668627	2.117731	3.97067	2.723703	2.239962	0.863752	6.763799

Table C-29. Relative Confidence Interval Half Widths for *Start*=7 and *Max*=8

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.833184	2.927135	3.05314	2.688315	10.415
0.3	3.859965	3.864639	2.738153	2.701621	3.563504	1.920799	8.866666
0.5	2.321731	2.345041	2.255112	4.843657	4.155145	2.827	6.250063
0.7	2.114586	1.949623	2.706902	1.863746	4.527476	2.680649	8.002487
0.9	1.312248	2.296101	3.402833	2.393577	7.404466	3.936638	8.379151

Table C-30. Relative Confidence Interval Half Widths for *Start*=7 and *Max*=10

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.833184	2.927135	3.05314	2.688315	10.415
0.3	3.859965	3.864639	3.456861	2.893222	2.466356	3.868723	9.904491
0.5	2.321731	2.345041	5.085412	4.044274	4.147809	1.227108	6.243889
0.7	2.114586	2.448249	6.807552	5.725654	4.304812	3.306499	5.951522
0.9	1.3592	2.367159	2.876819	4.715906	4.482856	3.826387	8.344696

Relative Confidence Interval Half Widths for *Start* = 8

Table C-31. Relative Confidence Interval Half Widths for *Start*=8 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.801175	2.945089	2.570748	0.687372	13.20362
0.3	3.859965	3.864639	3.834401	3.439837	1.869264	0.814304	6.660256
0.5	2.321731	2.345041	2.831402	2.864664	1.545973	1.791515	9.794167
0.7	2.114586	1.668547	2.785444	1.910461	0.400835	1.345065	10.01946
0.9	0.637025	0.786243	1.846864	3.15506	3.108523	1.442334	6.02164

Table C-32. Relative Confidence Interval Half Widths for *Start*=8 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.852777	2.901951	2.788362	2.732949	6.281402
0.3	3.859965	3.864639	3.853161	3.059317	1.138225	1.966254	13.43905
0.5	2.321731	2.345041	0.783579	4.74778	3.129249	2.118801	12.70415
0.7	2.114586	1.668547	3.489567	2.293717	0.780134	1.54057	13.39201
0.9	0.542365	0.906368	3.860387	2.455505	1.159116	0.672584	12.81341

Table C-33. Relative Confidence Interval Half Widths for *Start*=8 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.852777	2.901951	2.854409	3.243449	6.409492
0.3	3.859965	3.864639	3.807304	4.185238	2.326771	2.354027	13.78809
0.5	2.321731	2.345041	3.321611	4.347788	2.881811	2.405431	12.51636
0.7	2.114586	1.668547	2.911451	3.281687	2.239536	2.480284	12.0944
0.9	0.533695	0.912272	3.339639	2.324887	2.843196	1.586101	10.75701

Table C-34. Relative Confidence Interval Half Widths for *Start*=8 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.852777	2.901951	3.032076	3.44241	8.327678
0.3	3.859965	3.864639	3.906522	3.23582	2.263137	2.222478	7.450403
0.5	2.321731	2.345041	2.130649	2.502624	2.415228	4.576925	12.93375
0.7	2.114586	1.668547	2.445604	3.686849	4.068612	2.841715	7.98286
0.9	0.554598	0.912272	3.206331	2.814544	1.999496	4.02657	8.081968

Table C-35. Relative Confidence Interval Half Widths for *Start*=8 and *Max*=8

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.852777	2.901951	3.032076	3.44241	8.072465
0.3	3.859965	3.864639	3.855588	4.621522	6.826545	1.079511	5.143988
0.5	2.321731	2.345041	1.568317	4.773332	3.959643	3.131722	7.445865
0.7	2.114586	1.668547	2.703508	1.602272	2.183353	3.098787	10.93372
0.9	0.925756	0.912272	2.027521	5.873714	2.701596	3.459679	8.376359

Table C-36. Relative Confidence Interval Half Widths for *Start*=8 and *Max*=10

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.852777	2.901951	3.032076	3.44241	8.072465
0.3	3.859965	3.864639	3.855588	4.747444	2.515889	2.437534	8.074701
0.5	2.321731	2.345041	2.4442	4.166141	3.573693	3.695879	12.74547
0.7	2.114586	1.668547	4.632928	2.112712	0.993596	2.795647	3.813422
0.9	0.723217	0.912272	3.111476	3.026412	2.341578	4.516229	10.84331

Relative Confidence Interval Half Widths for *Start* = 9

Table C-37. Relative Confidence Interval Half Widths for *Start*=9 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.833184	2.91239	2.24797	1.637144	7.436636
0.3	3.859965	3.864639	3.867259	2.214051	1.772153	0.42347	9.975892
0.5	2.321731	2.345041	3.100628	1.837819	1.368899	3.530352	11.16654
0.7	2.114586	1.668547	2.407458	3.351634	1.709096	0.81986	4.001072
0.9	0.637219	0.810957	2.389456	2.111025	2.160457	1.582042	12.52525

Table C-38. Relative Confidence Interval Half Widths for *Start*=9 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.833184	2.886351	2.078075	3.113148	14.67923
0.3	3.859965	3.864639	3.768919	2.888342	2.463122	1.896147	11.89903
0.5	2.321731	2.345041	3.147341	4.453335	1.740307	1.632445	10.87428
0.7	2.114586	1.668547	2.996843	3.608576	3.22512	1.920643	7.336702
0.9	0.603696	0.810957	2.604949	2.268025	2.344924	1.890748	7.549382

Table C-39. Relative Confidence Interval Half Widths for *Start*=9 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.833184	2.870012	2.651793	4.339653	12.69312
0.3	3.859965	3.864639	3.880734	3.487587	2.011199	1.511146	12.31885
0.5	2.321731	2.345041	2.218237	0.9729	3.076509	1.574897	12.79015
0.7	2.114586	1.668547	2.609705	1.652065	4.069184	3.434734	11.83923
0.9	0.603696	0.810957	2.586775	1.721645	2.493838	1.495516	13.93682

Table C-40. Relative Confidence Interval Half Widths for *Start*=9 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.833184	2.870012	2.779984	2.619729	8.032077
0.3	3.859965	3.864639	3.732052	1.412312	4.509824	0.993076	9.821632
0.5	2.321731	2.345041	3.614152	1.450047	1.782624	1.800391	10.37132
0.7	2.114586	1.668547	7.301847	2.214342	5.754032	2.763991	9.767805
0.9	0.606867	0.810957	2.934928	1.641057	1.823546	1.184135	14.74728

Table C-41. Relative Confidence Interval Half Widths for *Start*=9 and *Max*=8

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.833184	2.870012	2.618816	3.79232	9.917422
0.3	3.859965	3.864639	3.250397	2.662682	2.639379	2.780571	9.694761
0.5	2.321731	2.345041	2.744078	1.997253	2.871254	2.255928	7.924946
0.7	2.114586	1.668547	3.973876	4.987701	2.620226	4.1771	12.58916
0.9	0.605208	0.810957	2.872323	2.277108	3.708391	3.237045	5.546515

Table C-42. Relative Confidence Interval Half Widths for *Start*=9 and *Max*=10

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.833184	2.870012	2.618816	3.79232	6.377824
0.3	3.859965	3.864639	3.798121	3.170895	1.686901	1.149447	8.202707
0.5	2.321731	2.345041	4.47894	1.159581	2.125156	2.115474	8.135275
0.7	2.114586	1.668547	2.802067	3.684895	2.955989	3.193582	11.1355
0.9	0.604451	0.810957	4.023778	2.388081	3.639007	3.4381	11.71553

Relative Confidence Interval Half Widths for IEEE 802.11

Table C-43. Relative Confidence Interval Half Widths for Standard IEEE 802.11

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	2.788829	2.788829	2.833184	3.878059	N/A	N/A	N/A
0.3	3.859965	3.864639	3.864502	2.428413	N/A	N/A	N/A
0.5	2.321731	2.345041	2.110321	4.157128	N/A	N/A	N/A
0.7	2.114586	1.668547	1.159154	2.967449	N/A	N/A	N/A
0.9	0.616318	0.810957	1.577222	3.446132	N/A	N/A	N/A

Appendix D. Difference Surface Plots for 10 Station Simulation Data

This appendix contains the difference surface plots for the 10 station simulation data. An explanation of these plots is found in Chapter 6.

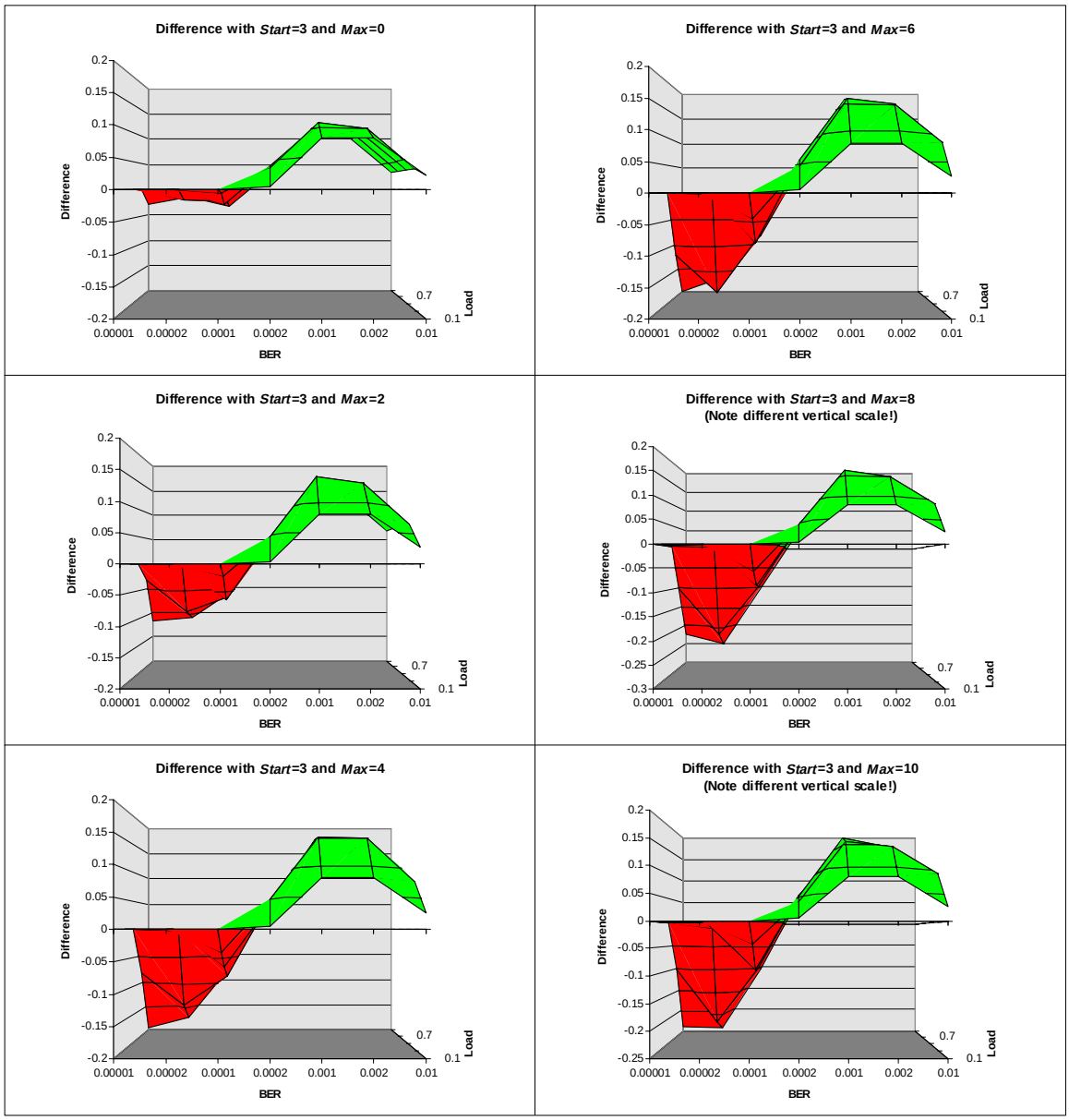

Figure D-1. Difference Plots with *Start*=3

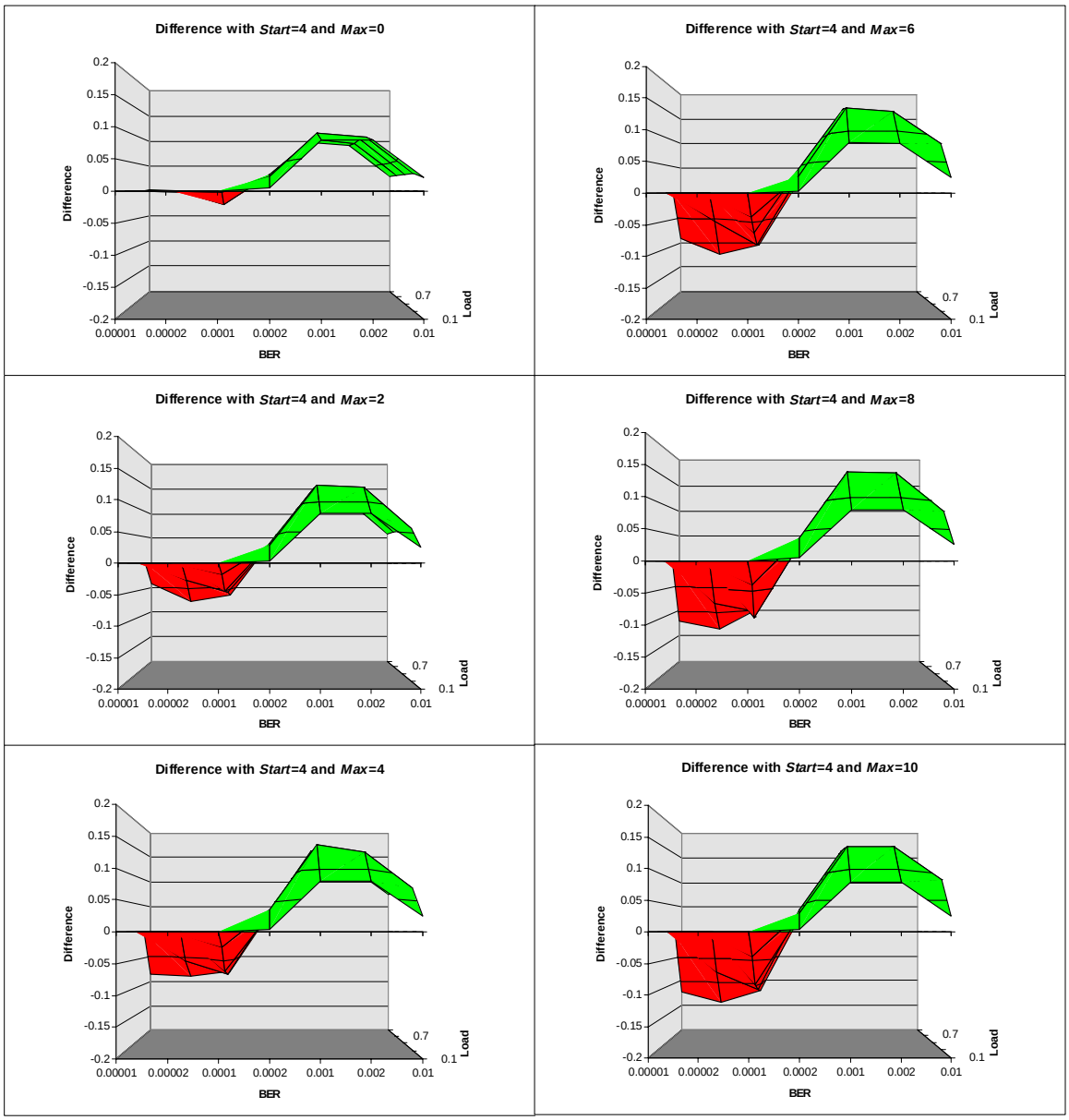

Figure D-2. Difference Plots with *Start*=4

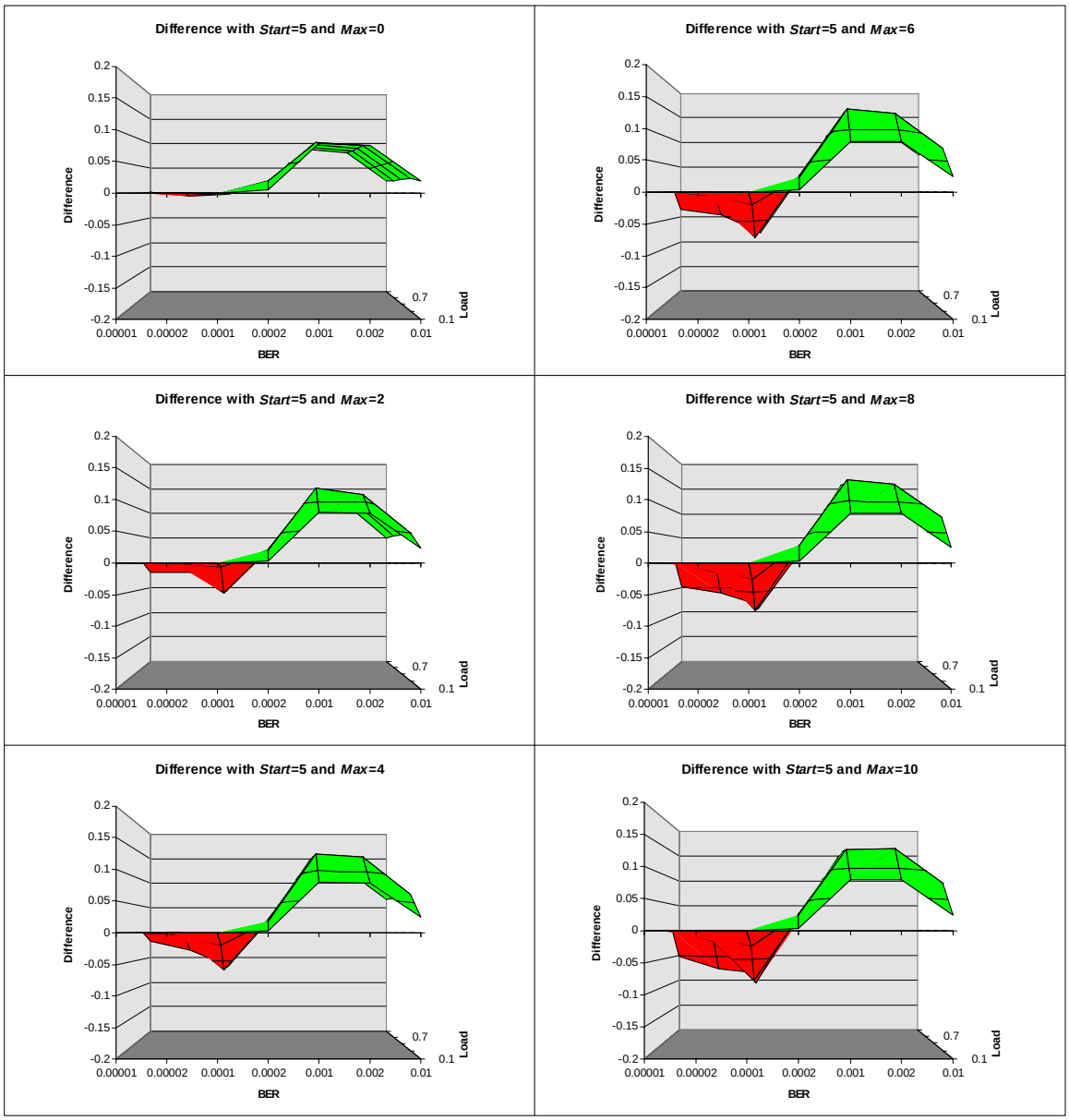

Figure D-3. Difference Plots with *Start*=5

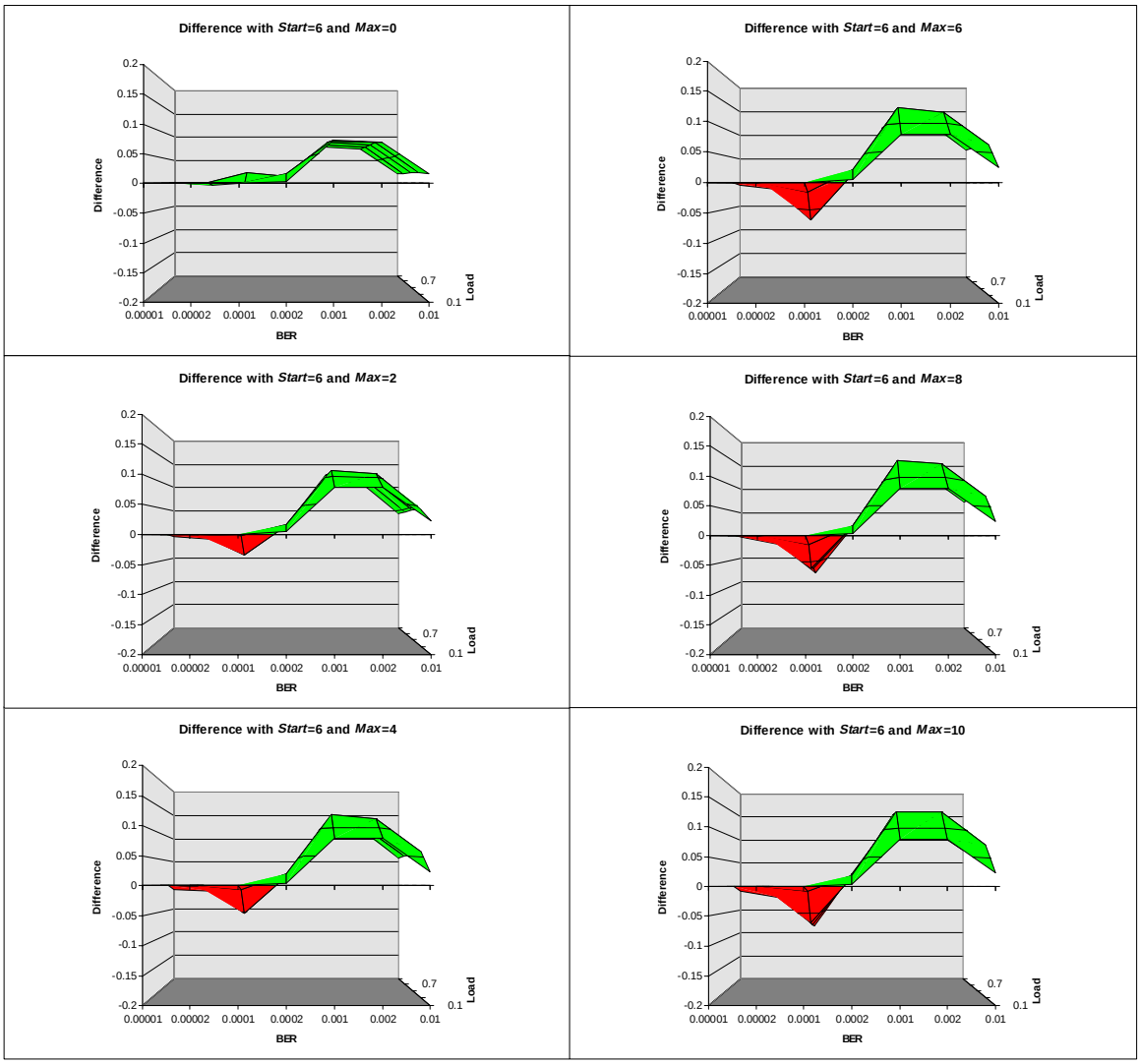

Figure D-4. Difference Plots with *Start*=6

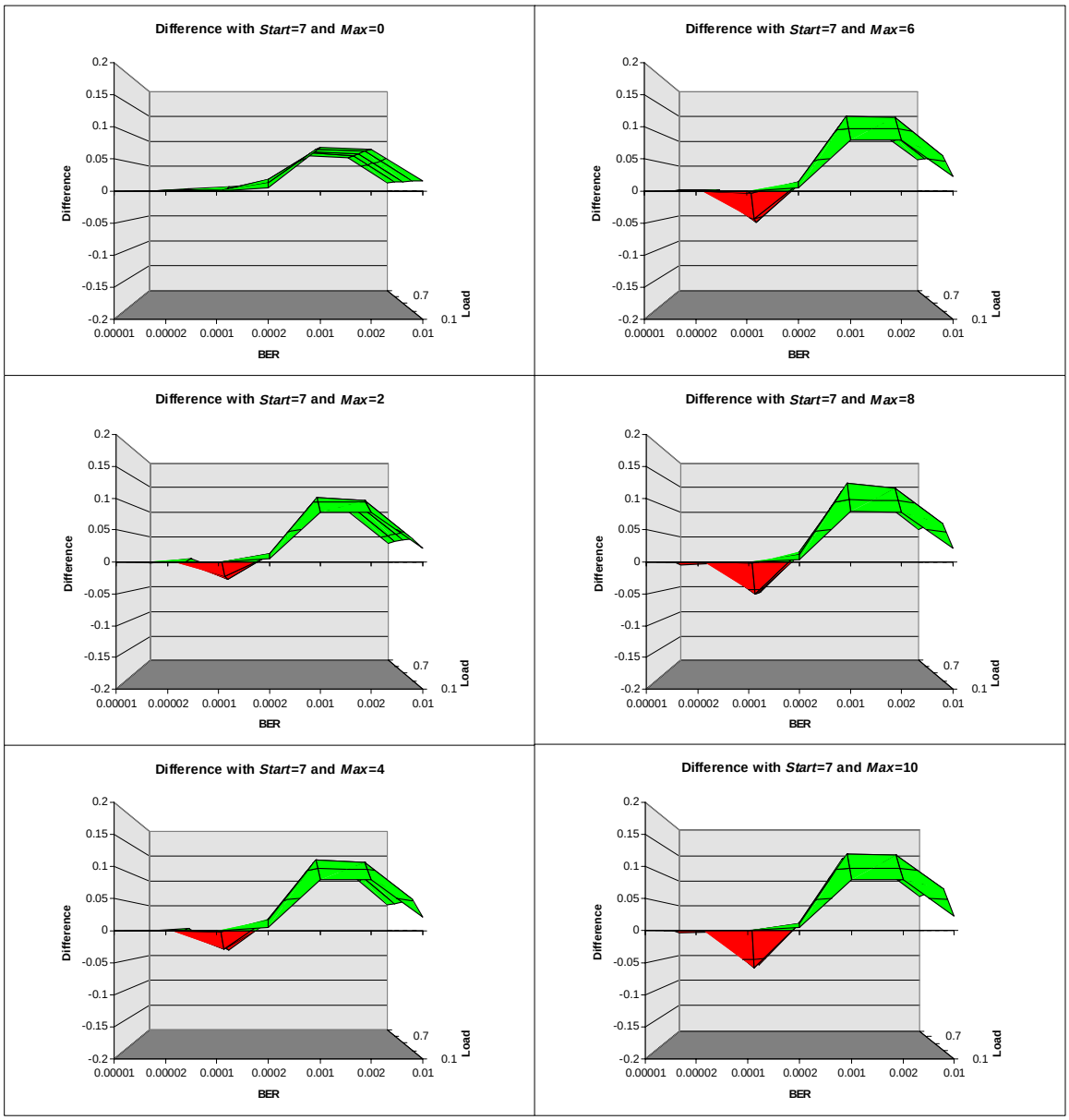

Figure D-5. Difference Plots with *Start*=7

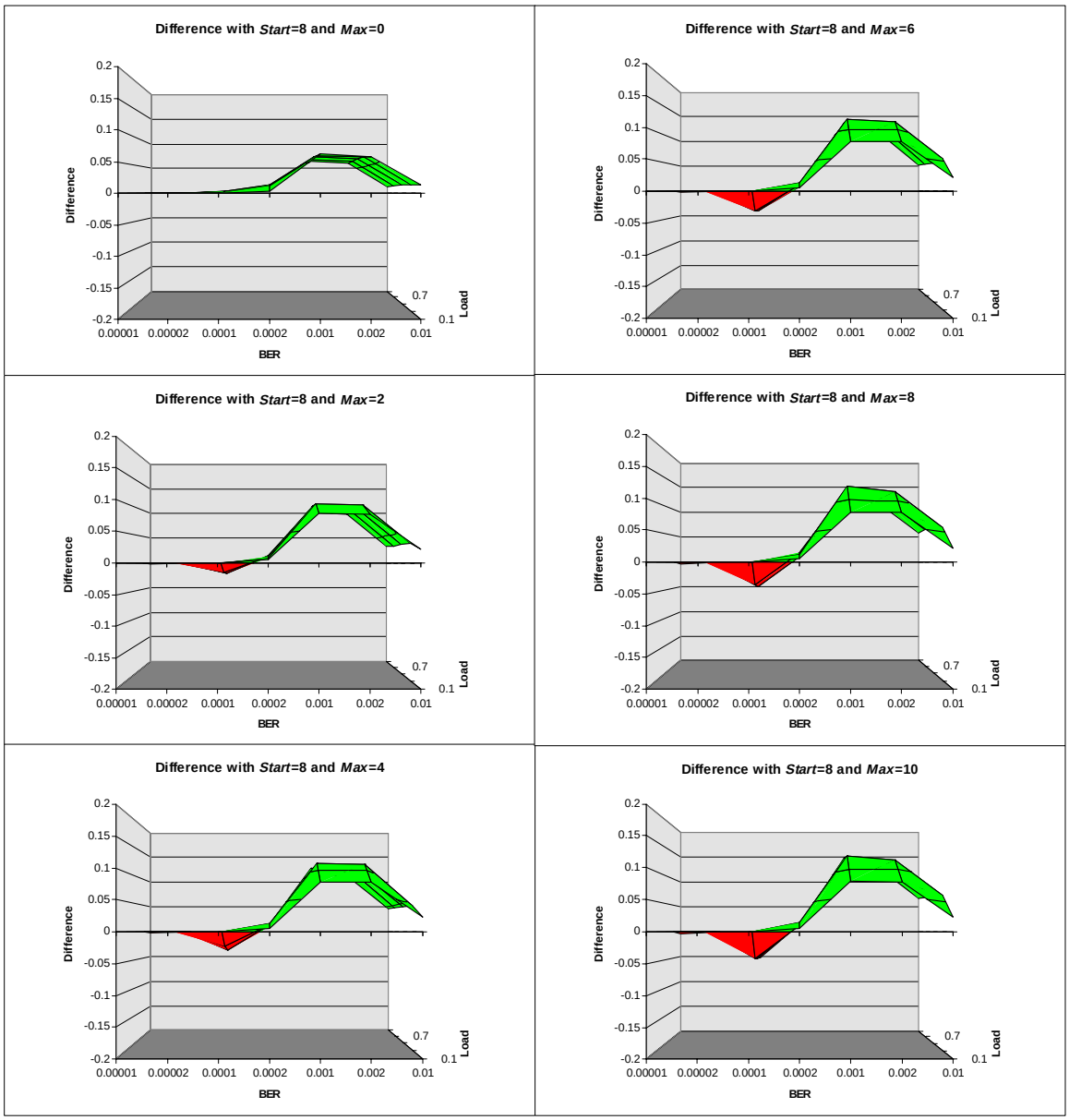

Figure D-6. Difference Plots with *Start*=8

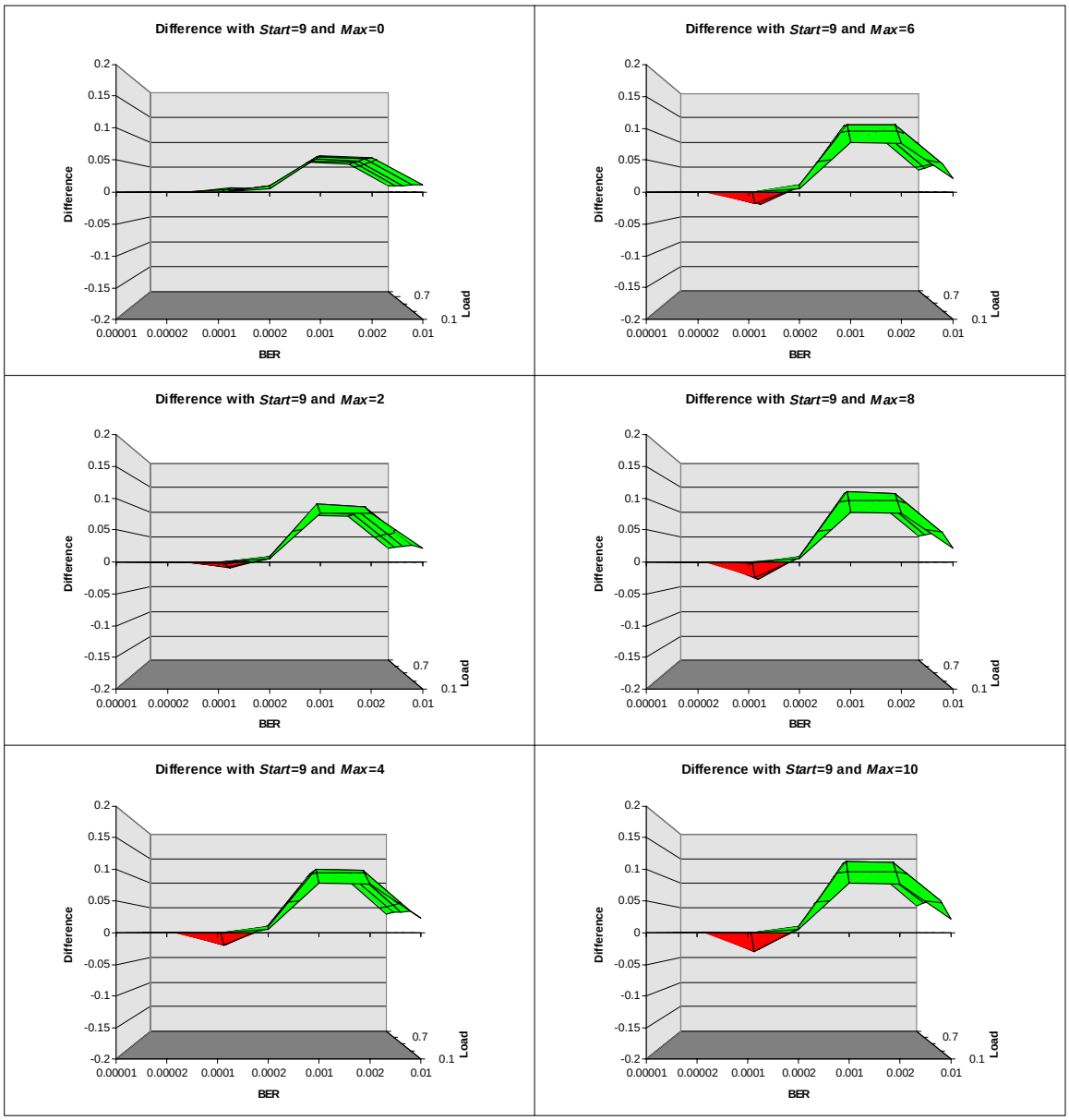

Figure D-7. Difference Plots with *Start*=9

Appendix E. Aggregate Throughput Plots Using Dynamic Gilbert Errors

This appendix contains all plots of aggregate throughput using Gilbert errors during simulations. An explanation of the these plots is found in Chapter 6.

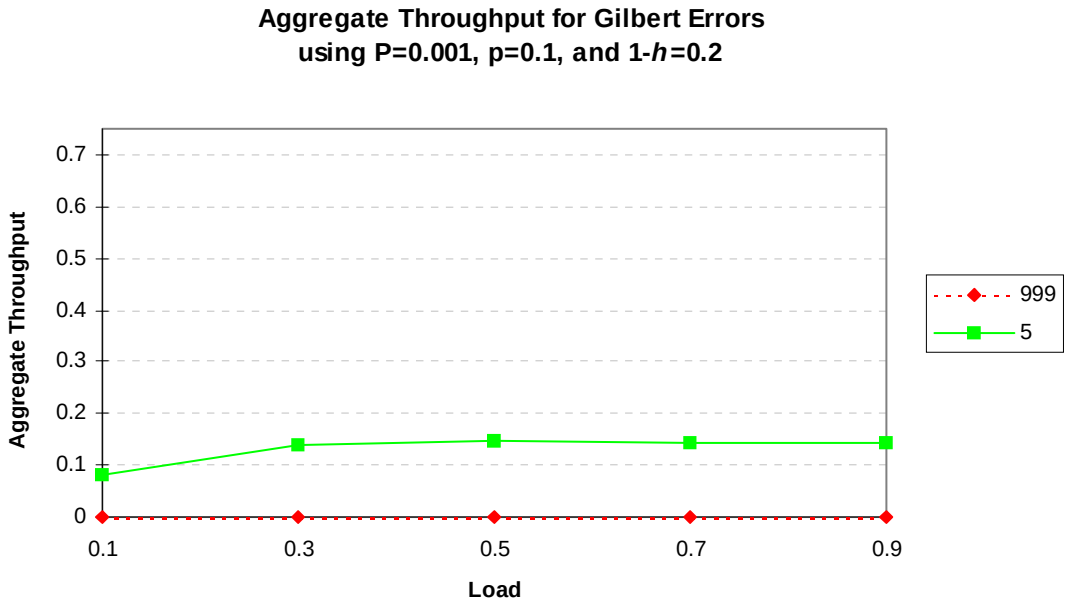

Figure E-1. Throughput for Gilbert Errors (P=0.001, p=0.1, and 1-*h*=0.2)

Table E-1. Relative Confidence Interval Half Widths (P=0.001, p=0.1 and 1-*h*=0.2)

	Start	
Load	999	5
0.1	30.3978	2.965898
0.3	43.79042	5.889535
0.5	46.92217	5.168442
0.7	28.86788	4.444092
0.9	21.3159	4.323746

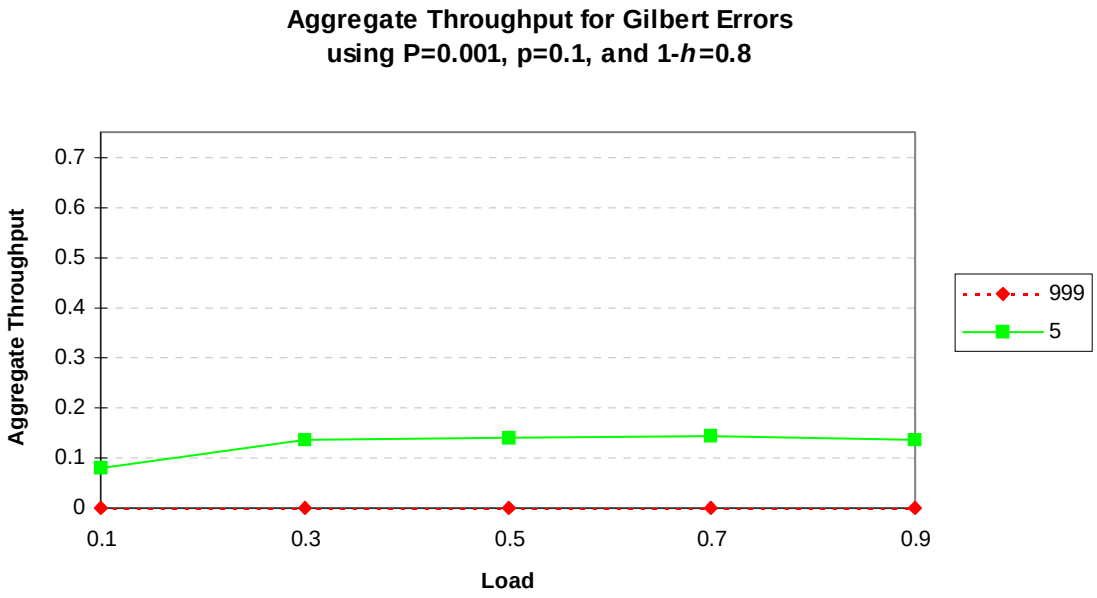

Figure E-2. Throughput for Gilbert Errors (P=0.001, p=0.1, and 1-*h*=0.8)

Table E-2. Relative Confidence Interval Half Widths (P=0.001, p=0.1, and 1-*h*=0.8)

	Start	
Load	999	5
0.1	113.9602	2.841244
0.3	165.144	3.908838
0.5	87.03854	2.853758
0.7	95.34594	5.161551
0.9	23.68889	5.898512

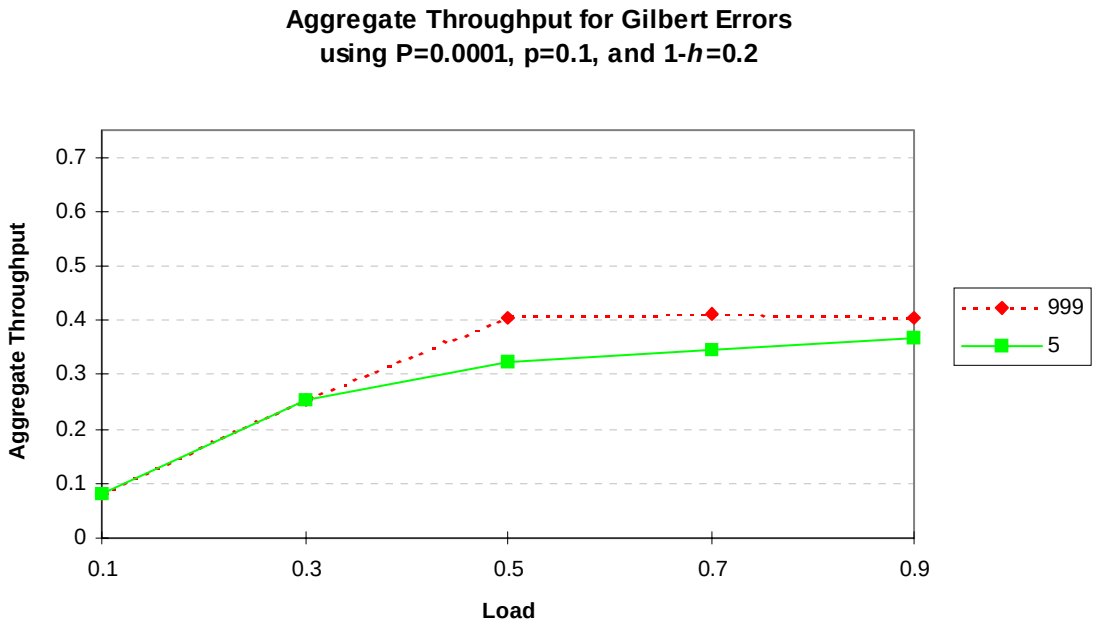

Figure E-3. Throughput for Gilbert Errors (P=0.0001, p=0.1, and 1-*h*=0.2)

Table E-3. Relative Confidence Interval Half Widths (P=0.0001, p=0.1, and 1-*h*=0.2)

	Start	
Load	999	5
0.1	2.801175	2.815971
0.3	3.850543	3.872324
0.5	1.812249	9.268892
0.7	0.75728	6.348609
0.9	0.894591	14.53871

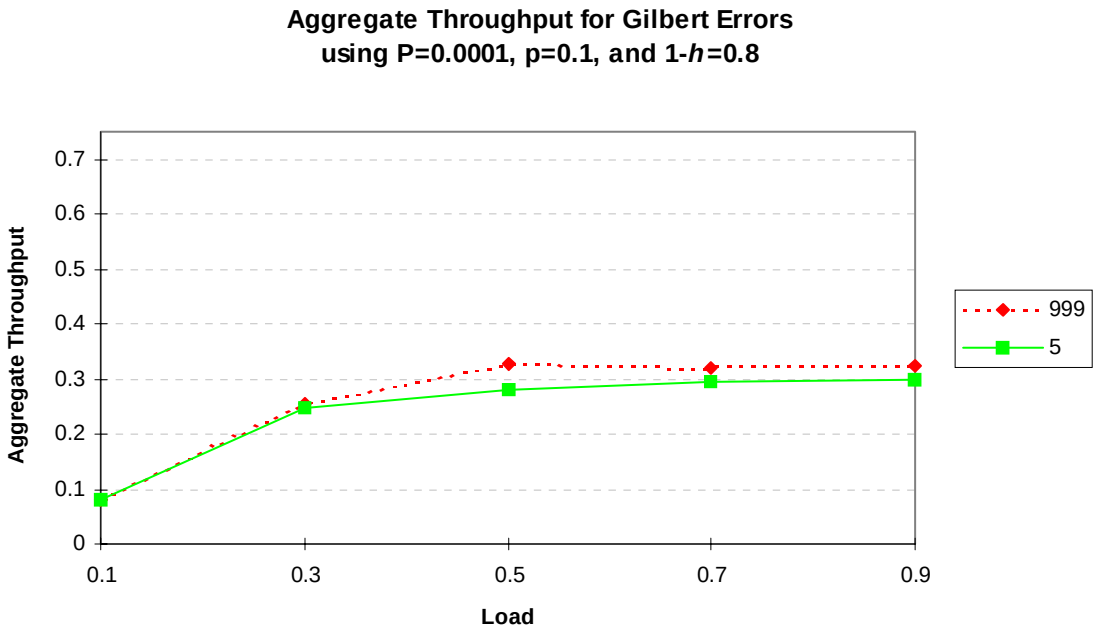

Figure E-4. Throughput for Gilbert Errors (P=0.0001, p=0.1, and 1-*h*=0.8)

Table E-4. Relative Confidence Interval Half Widths (P=0.0001, p=0.1, and 1-*h*=0.8)

	Start	
Load	999	5
0.1	2.815971	2.898997
0.3	3.853111	2.278654
0.5	1.407693	6.133835
0.7	1.140001	6.889448
0.9	0.850338	12.72144

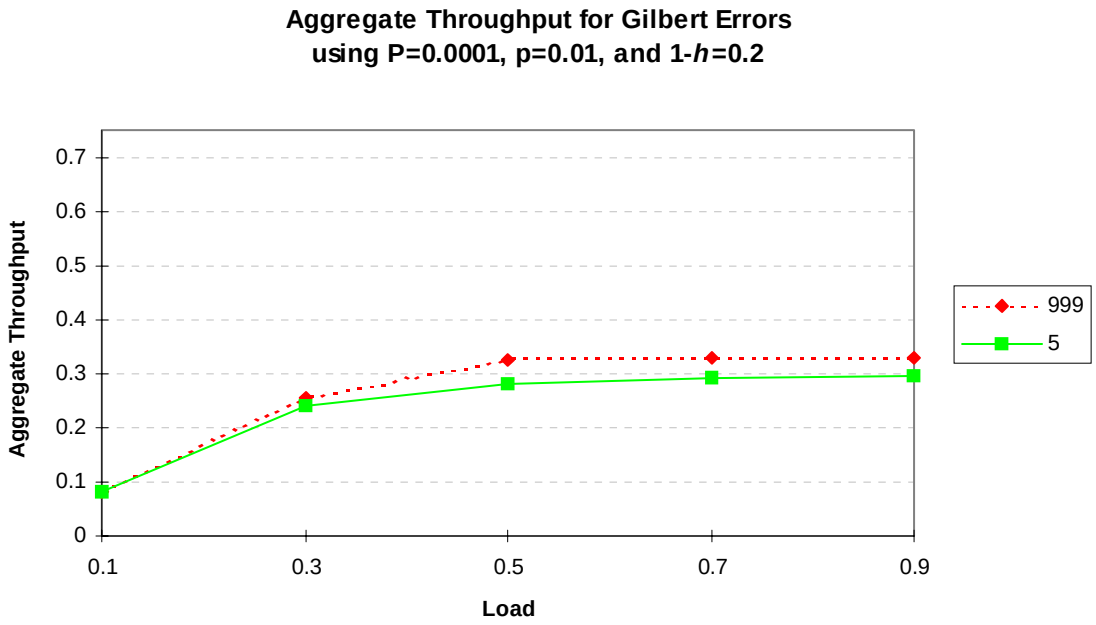

Figure E-5. Throughput for Gilbert Errors (P=0.0001, p=0.01, and 1-*h*=0.2)

Table E-5. Relative Confidence Interval Half Widths (P=0.0001, p=0.01, and 1-*h*=0.2)

	Start	
Load	999	5
0.1	2.788829	2.830094
0.3	3.789847	3.929062
0.5	1.27333	7.418787
0.7	0.998264	9.362855
0.9	0.980708	7.082594

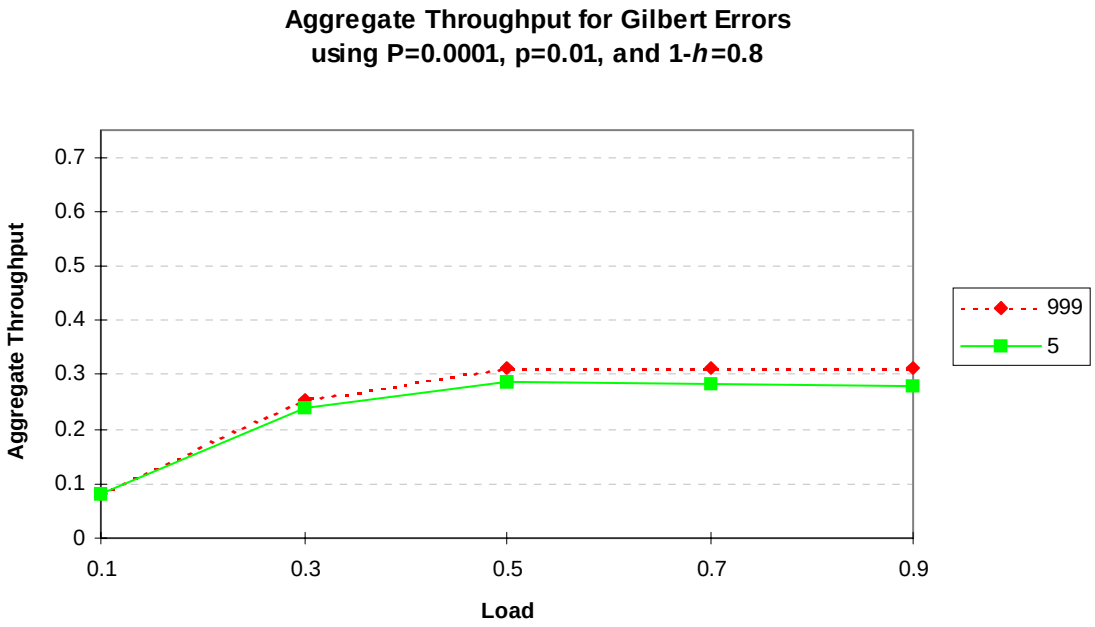

Figure E-6. Throughput for Gilbert Errors (P=0.0001, p=0.01, and 1-*h*=0.8)

Table E-6. Relative Confidence Interval Half Widths (P=0.0001, p=0.01, and 1-*h*=0.8)

	Start	
Load	999	5
0.1	2.852777	2.917026
0.3	3.745864	2.684568
0.5	1.160981	7.560735
0.7	0.976065	12.57306
0.9	2.258793	7.5716

Figure E-7. Throughput for Gilbert Errors (P=0.000001, p=0.1, and 1-*h*=0.2)

Table E-7. Relative Confidence Interval Half Widths (P=0.000001, p=0.1, and 1-*h*=0.2)

	Start	
Load	999	5
0.1	2.788829	2.788829
0.3	3.859965	3.859965
0.5	2.333247	2.333247
0.7	2.155307	2.155307
0.9	1.59876	2.059451

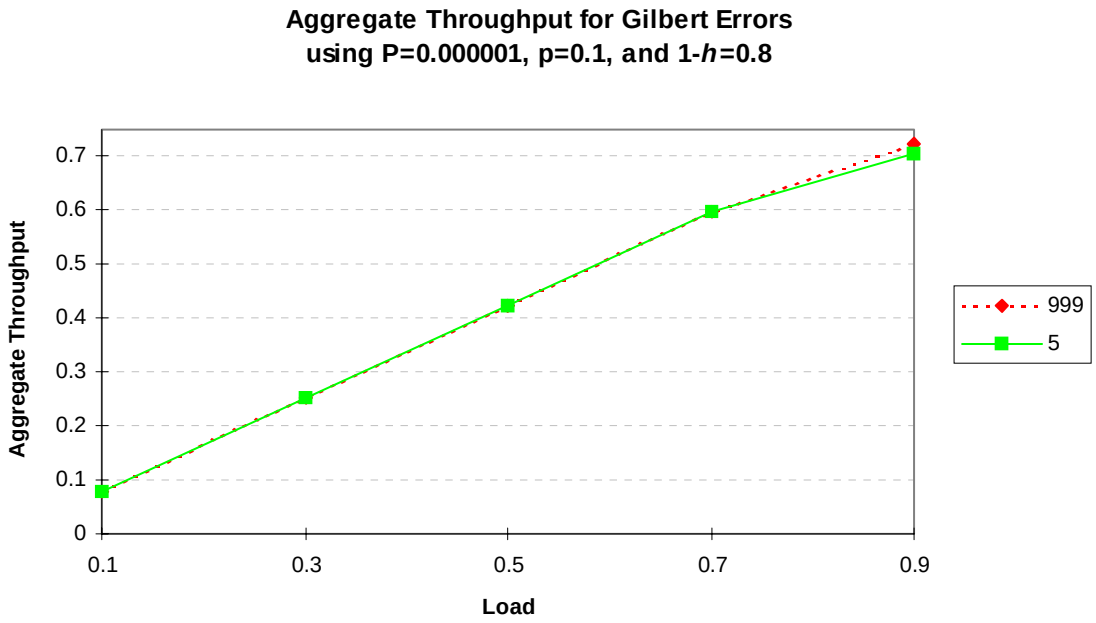

Figure E-8. Throughput for Gilbert Errors (P=0.000001, p=0.1, and 1-*h*=0.8)

Table E-8. Relative Confidence Interval Half Widths (P=0.000001, p=0.1, and 1-*h*=0.8)

	Start	
Load	999	5
0.1	2.788829	2.788829
0.3	3.859965	3.859965
0.5	2.333247	2.333247
0.7	2.15942	2.15942
0.9	1.618513	1.494554

Figure E-9. Throughput for Gilbert Errors (P=0.000001, p=0.01, and 1- *h*=0.2)

Table E-9. Relative Confidence Interval Half Widths (P=0.000001, p=0.01, and 1- *h*=0.2)

	Start	
Load	999	5
0.1	2.788829	2.788829
0.3	3.859965	3.859965
0.5	2.333247	2.333247
0.7	2.140082	2.140082
0.9	0.460789	2.022315

Figure E-10. Throughput for Gilbert Errors (P=0.000001, p=0.01, and 1- *h*=0.8)

Table E-10. Relative Confidence Interval Half Widths (P=0.000001, p=0.001, and 1- *h*=0.8)

	Start	
Load	999	5
0.1	2.788829	2.788829
0.3	3.859965	3.859965
0.5	2.333247	2.333247
0.7	2.152838	2.152838
0.9	0.729795	0.955339

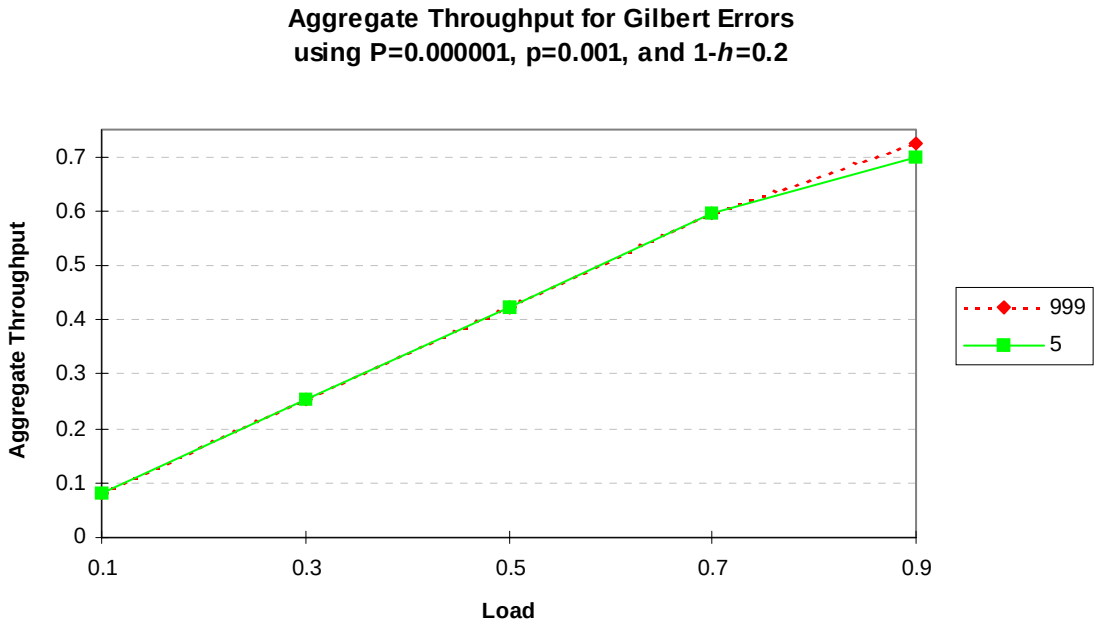

Figure E-11. Throughput for Gilbert Errors (P=0.000001, p=0.001, and 1- *h*=0.2)

Table E-11. Relative Confidence Interval Half Widths (P=0.000001, p=0.001, and 1- *h*=0.2)

	Start	
Load	999	5
0.1	2.788829	2.788829
0.3	3.859965	3.859965
0.5	2.333247	2.333247
0.7	2.163728	2.163728
0.9	0.744578	2.053075

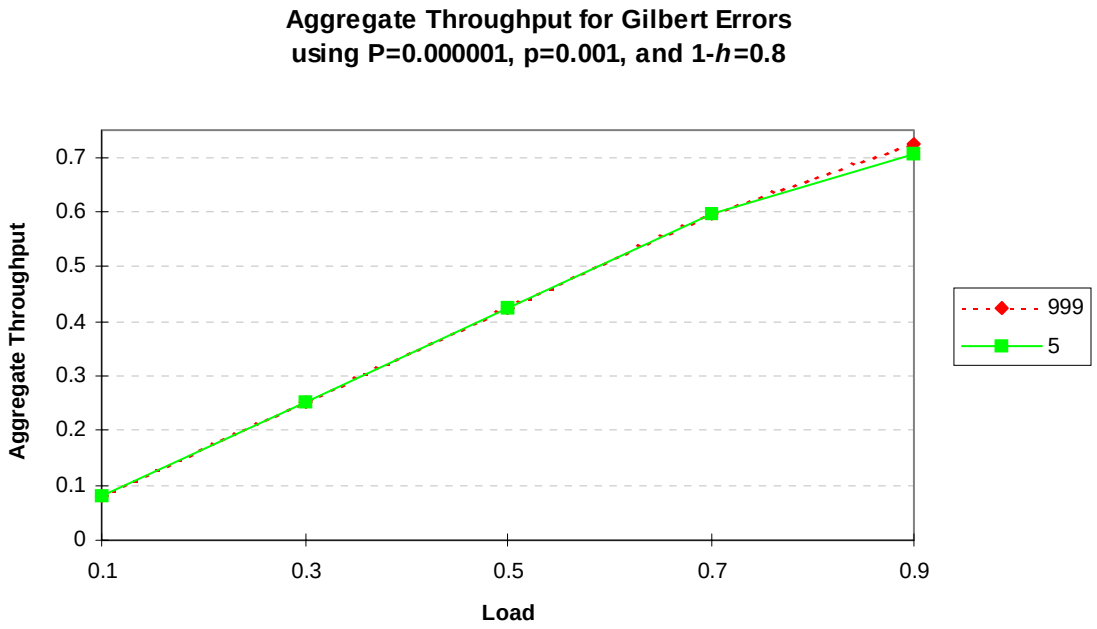

Figure E-12. Throughput for Gilbert Errors (P=0.000001, p=0.001, and 1- *h*=0.8)

Table E-12. Relative Confidence Interval Half Widths (P=0.000001, p=0.0001, and 1- *h*=0.8)

	Start	
Load	999	5
0.1	2.788829	2.788829
0.3	3.859965	3.859965
0.5	2.333247	2.333247
0.7	2.166257	2.166257
0.9	0.766264	1.005286

Appendix F. System Fairness Plots

This appendix contains all plots pertaining to system fairness as described in Chapter 6.

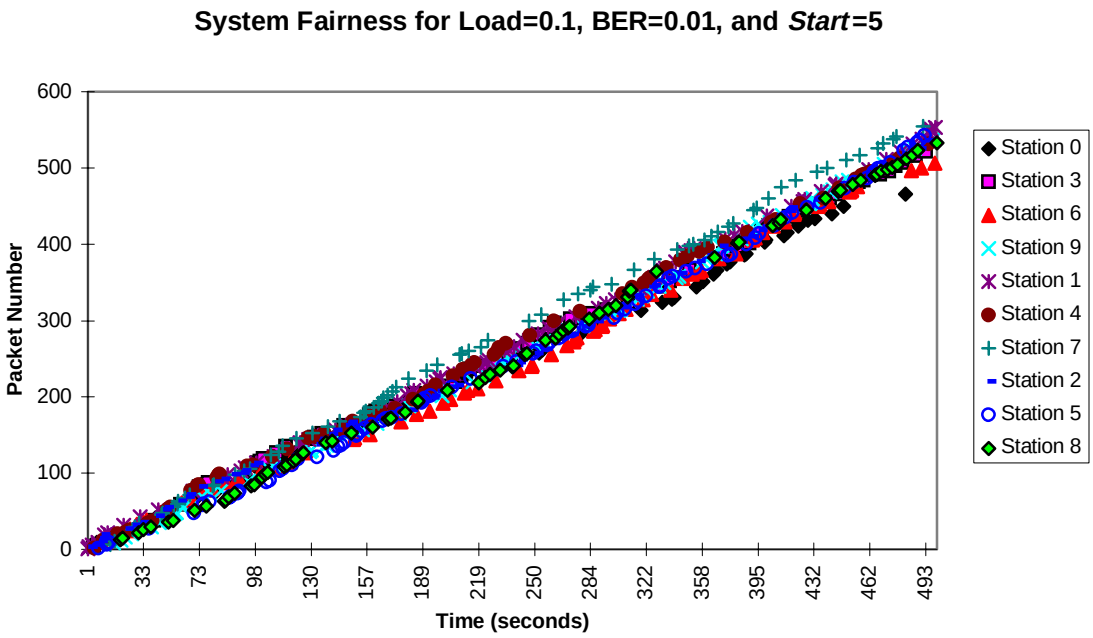

Figure F-1. System Fairness for Load=0.1, BER=0.01, and *Start*=5

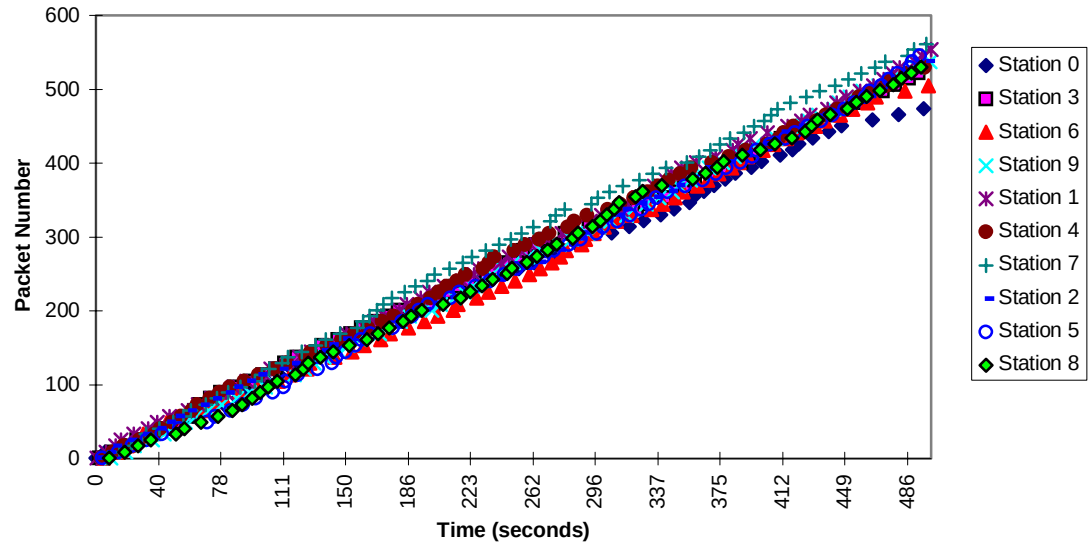

Figure F-2. System Fairness for Load=0.1, BER=0.001, and *Start*=5

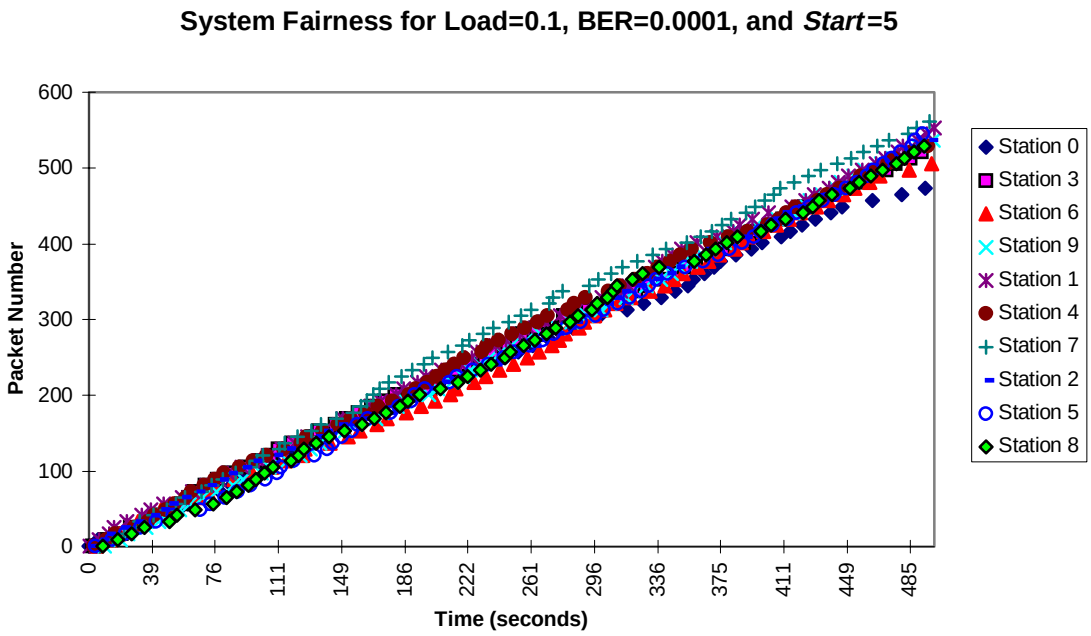

Figure F-3. System Fairness for Load=0.1, BER=0.0001, and *Start*=5

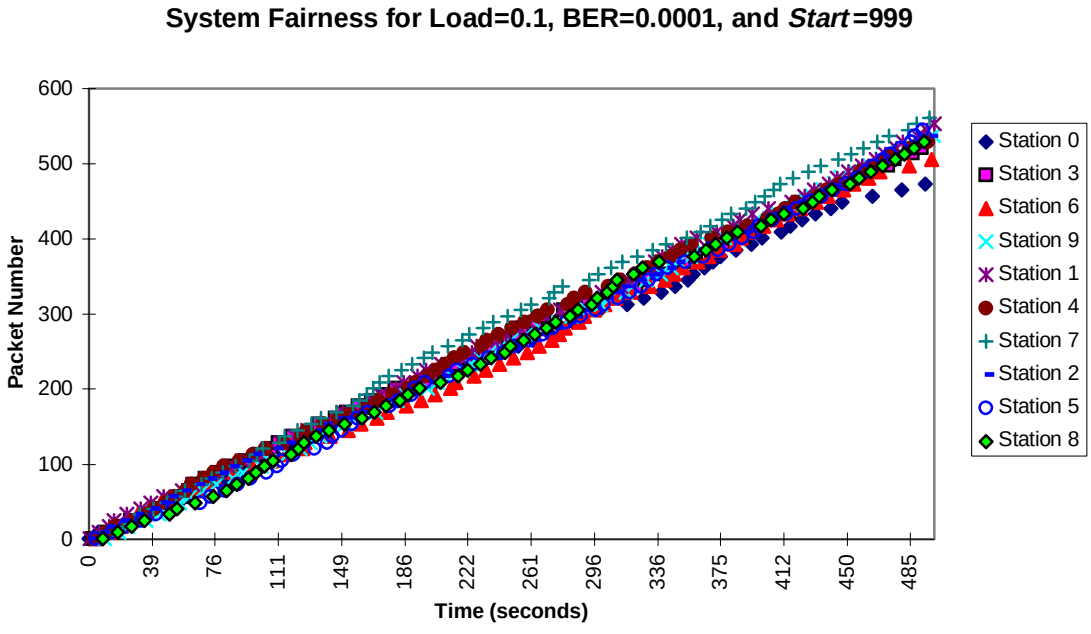

Figure F-4. System Fairness for Load=0.1, BER=0.0001, and *Start*=999

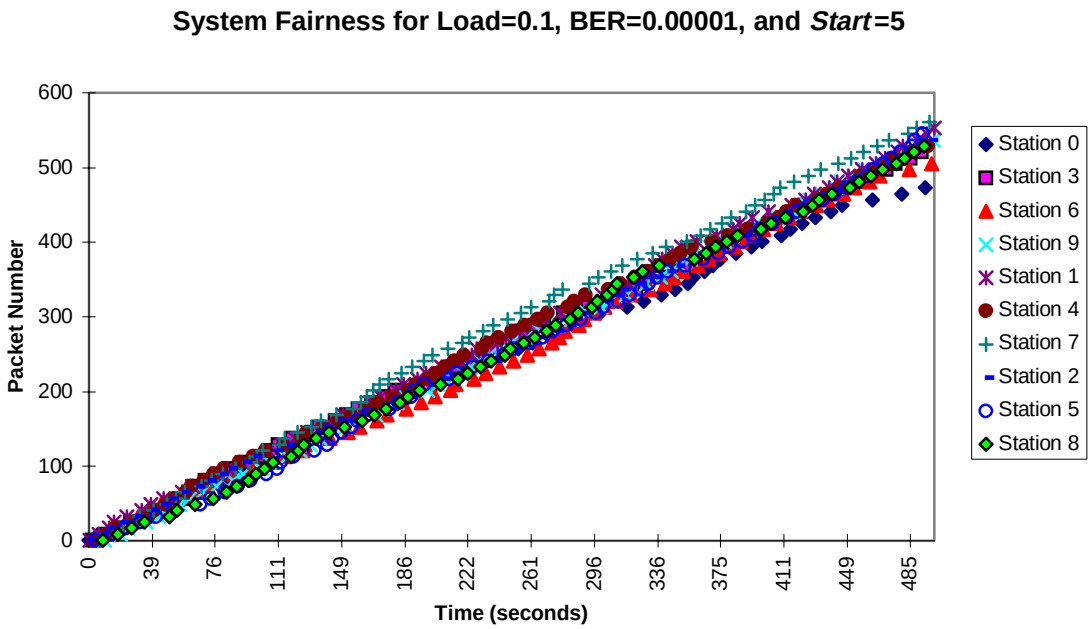

Figure F-5. System Fairness for Load=0.1, BER=0.00001, and *Start*=5

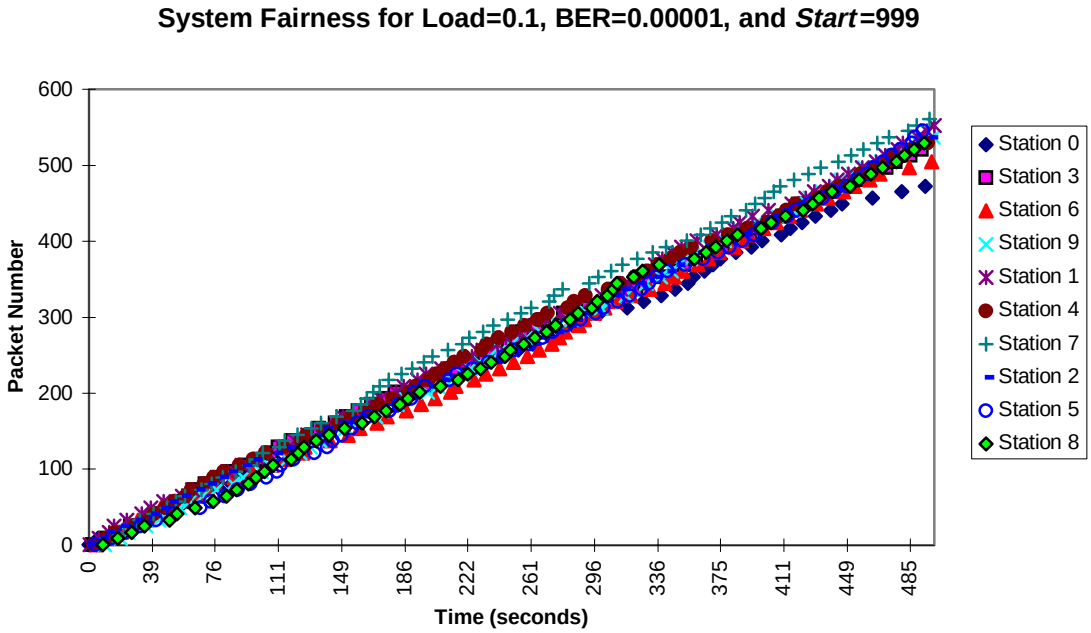

Figure F-6. System Fairness for Load=0.1, BER=0.00001, and *Start*=999

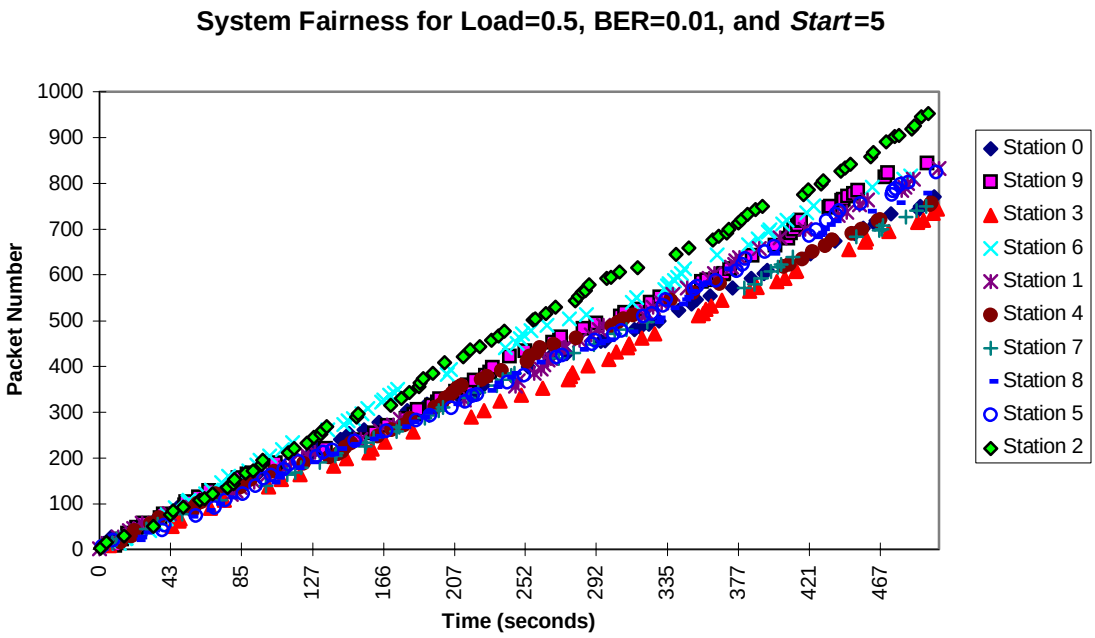

Figure F-7. System Fairness for Load=0.5, BER=0.01, and *Start*=5

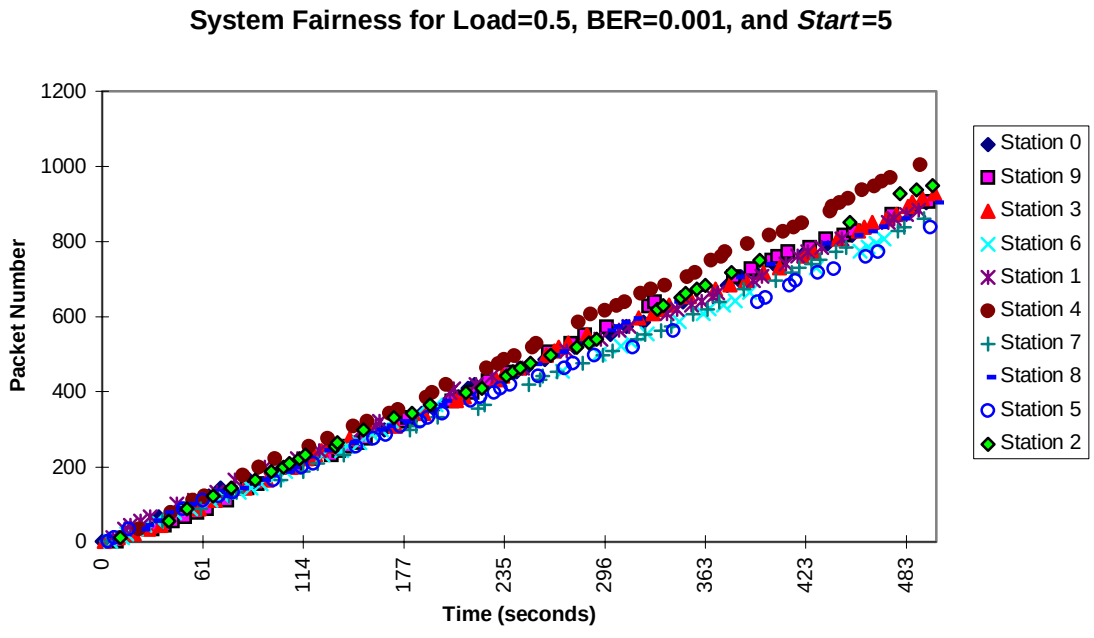

Figure F-8. System Fairness for Load=0.5, BER=0.001, and *Start*=5

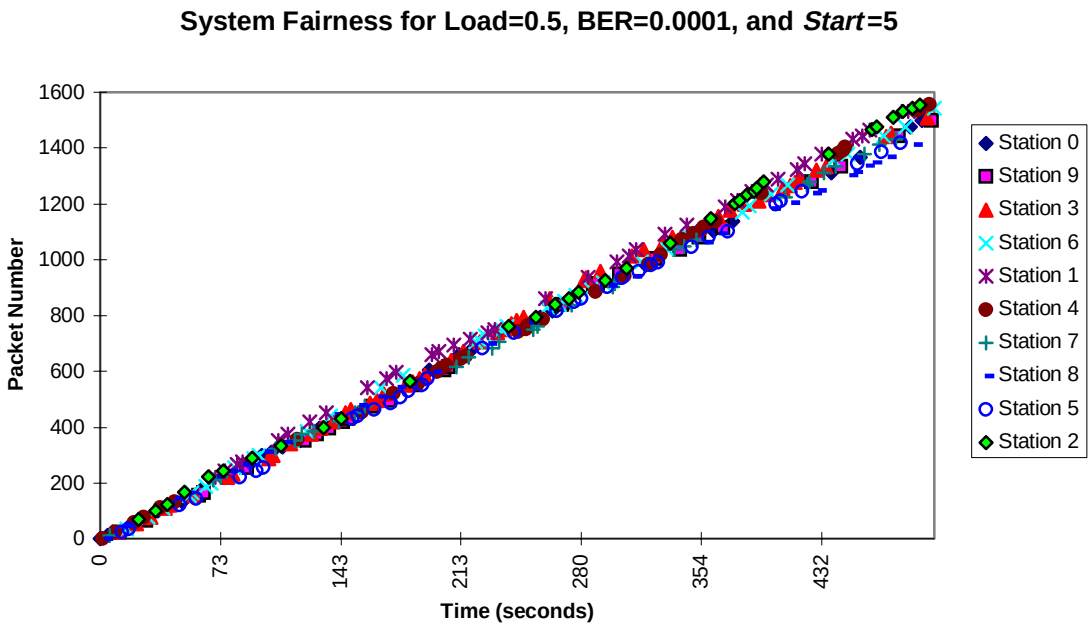

Figure F-9. System Fairness for Load=0.5, BER=0.0001, and *Start*=5

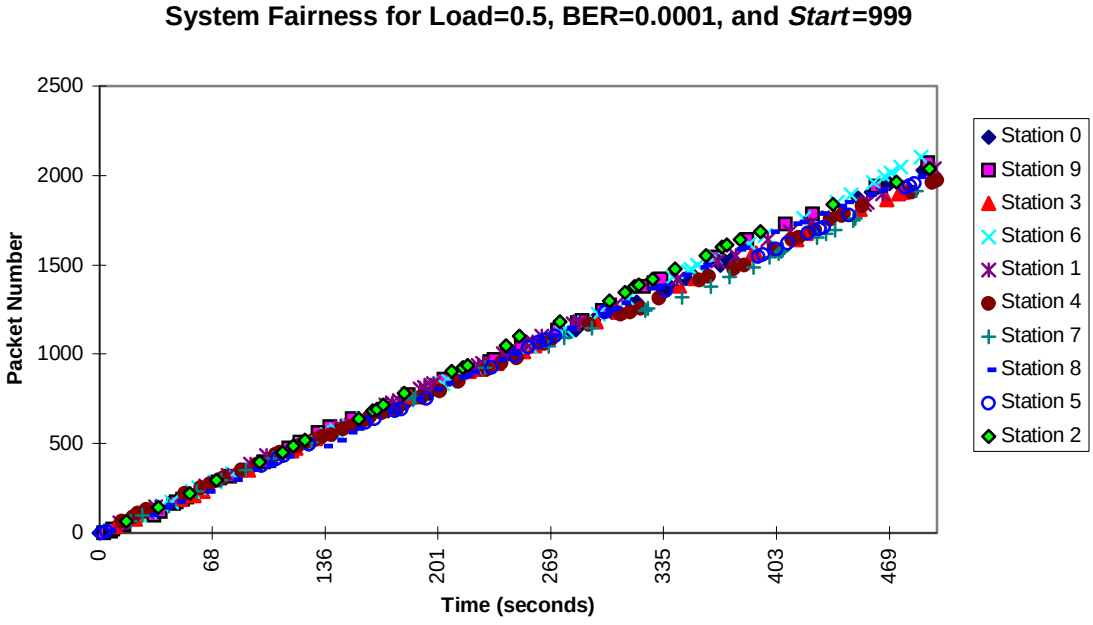

Figure F-10. System Fairness for Load=0.5, BER=0.0001, and *Start*=999

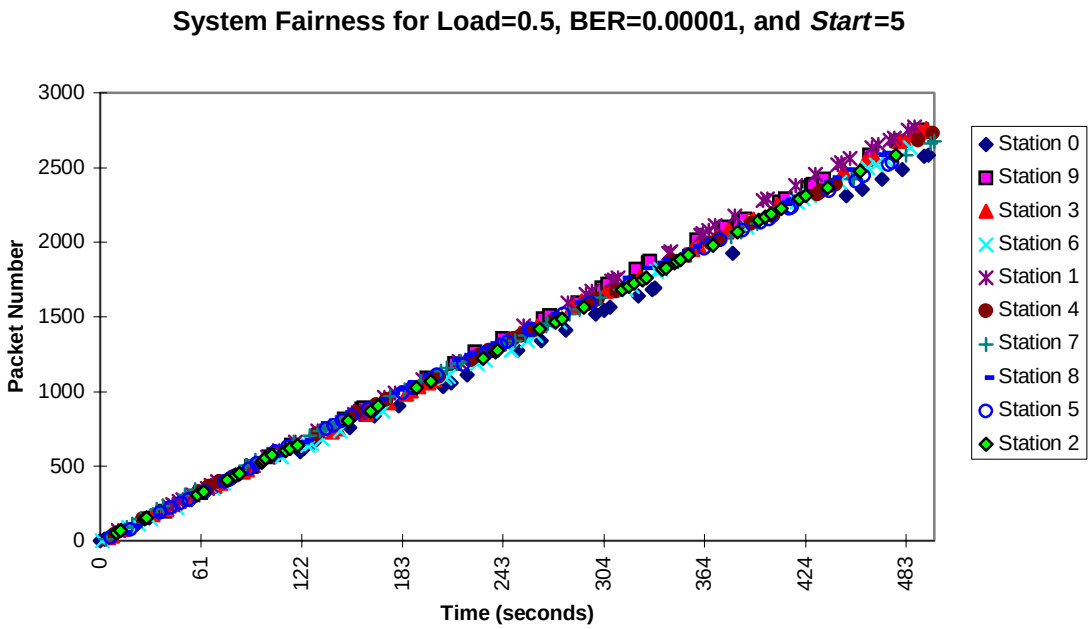

Figure F-11. System Fairness for Load=0.5, BER=0.00001, and *Start*=5

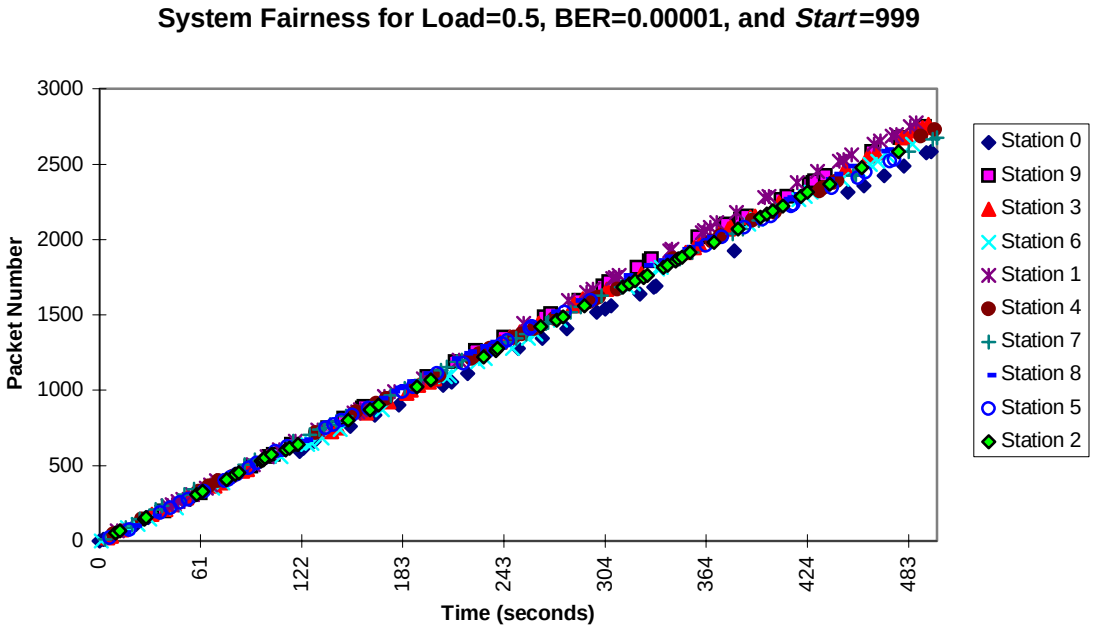

Figure F-12. System Fairness for Load=0.5, BER=0.00001, and *Start*=999

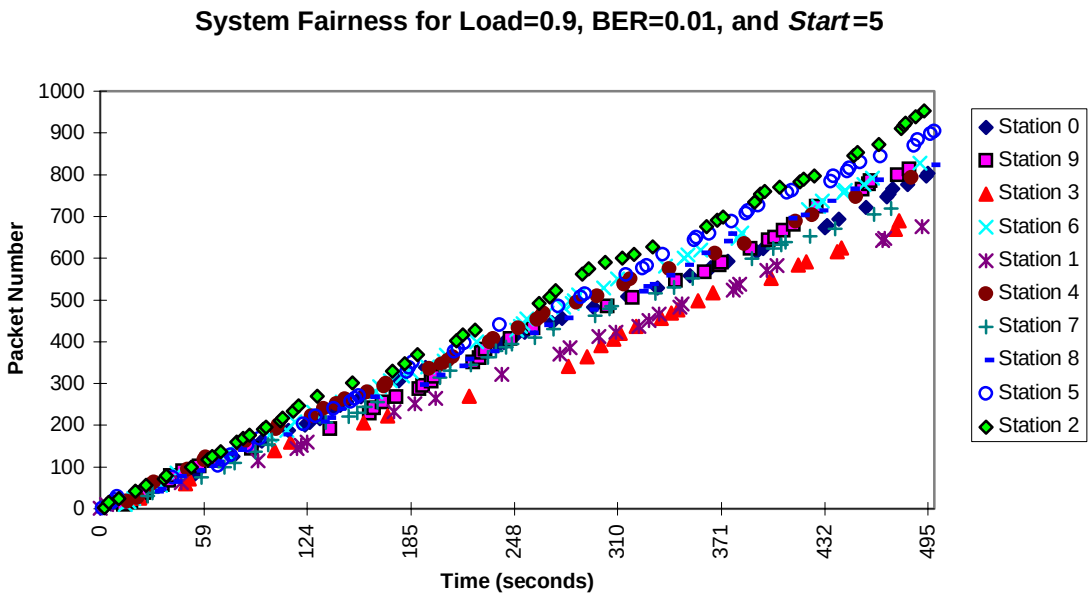

Figure F-13. System Fairness for Load=0.9, BER=0.01, and *Start*=5

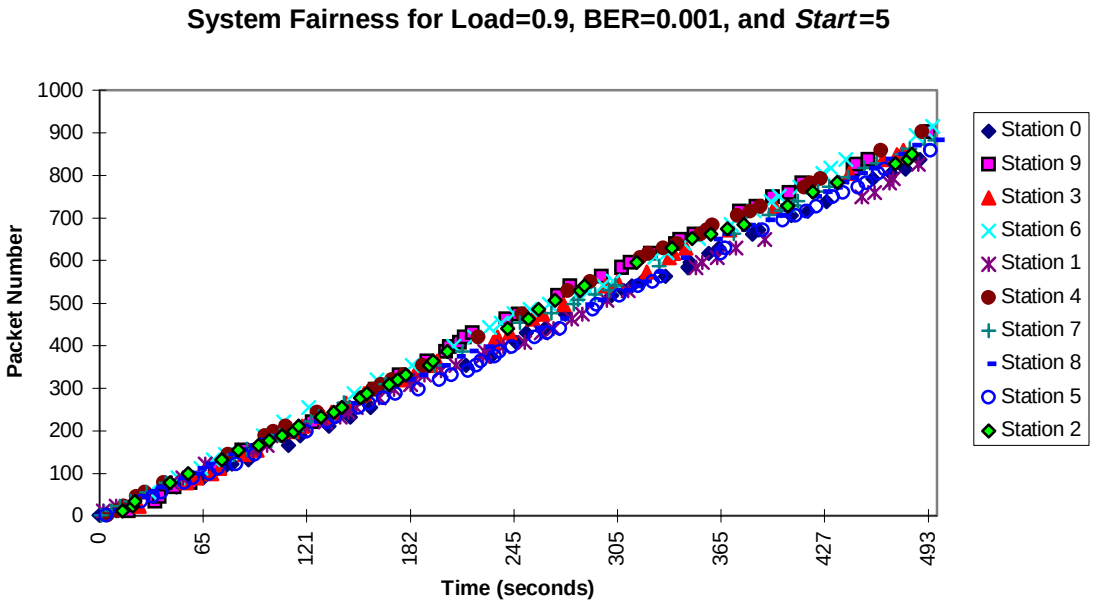

Figure F-14. System Fairness for Load=0.9, BER=0.001, and *Start*=5

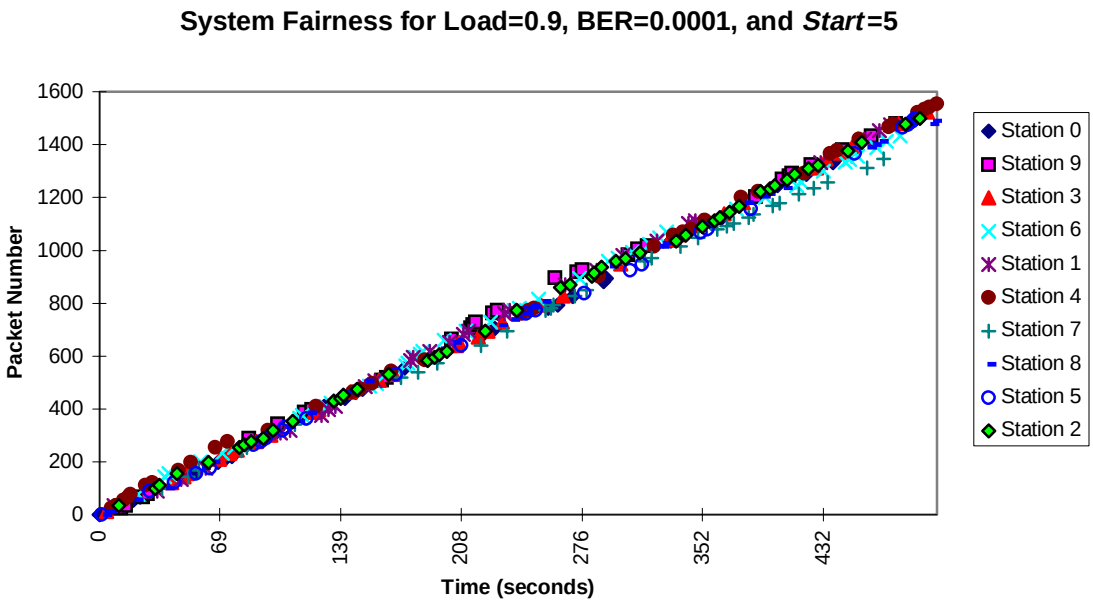

Figure F-15. System Fairness for Load=0.9, BER=0.0001, and *Start*=5

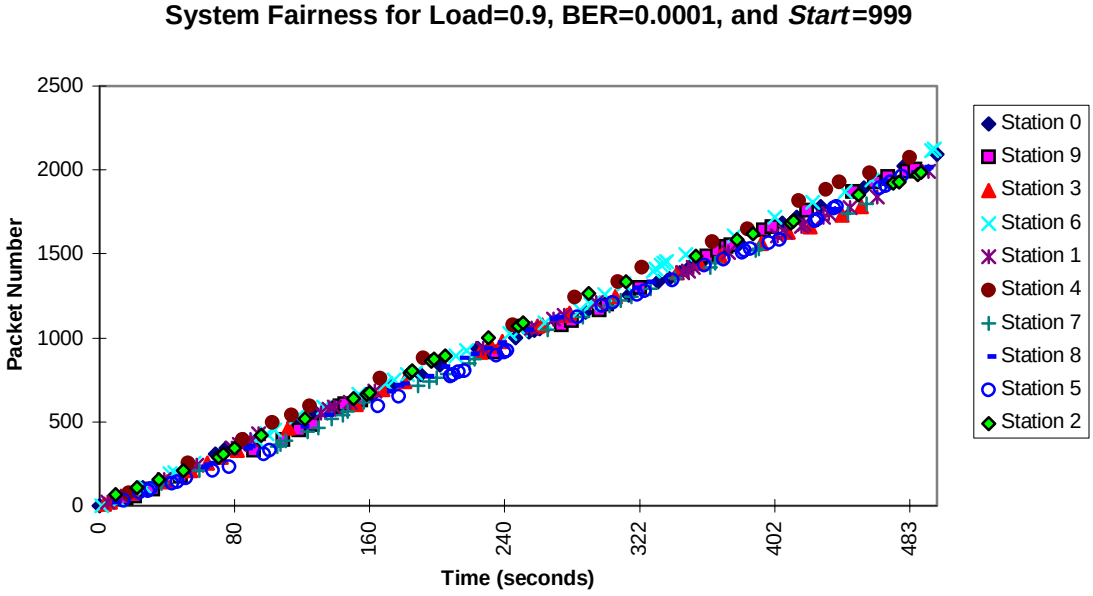

Figure F-16. System Fairness for Load=0.9, BER=0.0001, and *Start*=999

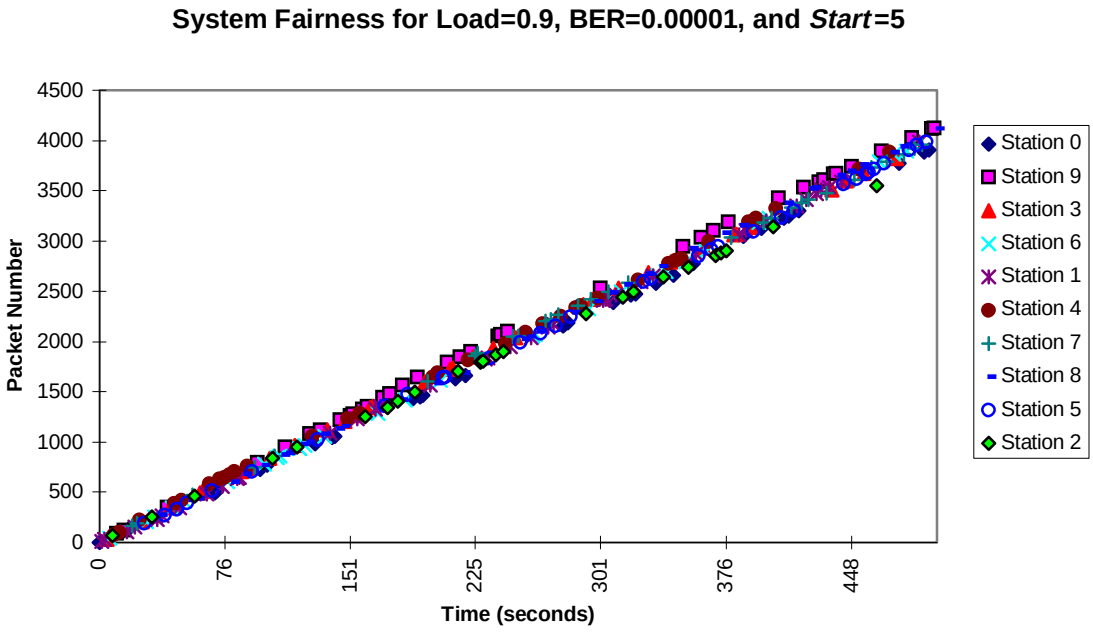

Figure F-17. System Fairness for Load=0.9, BER=0.00001, and *Start*=5

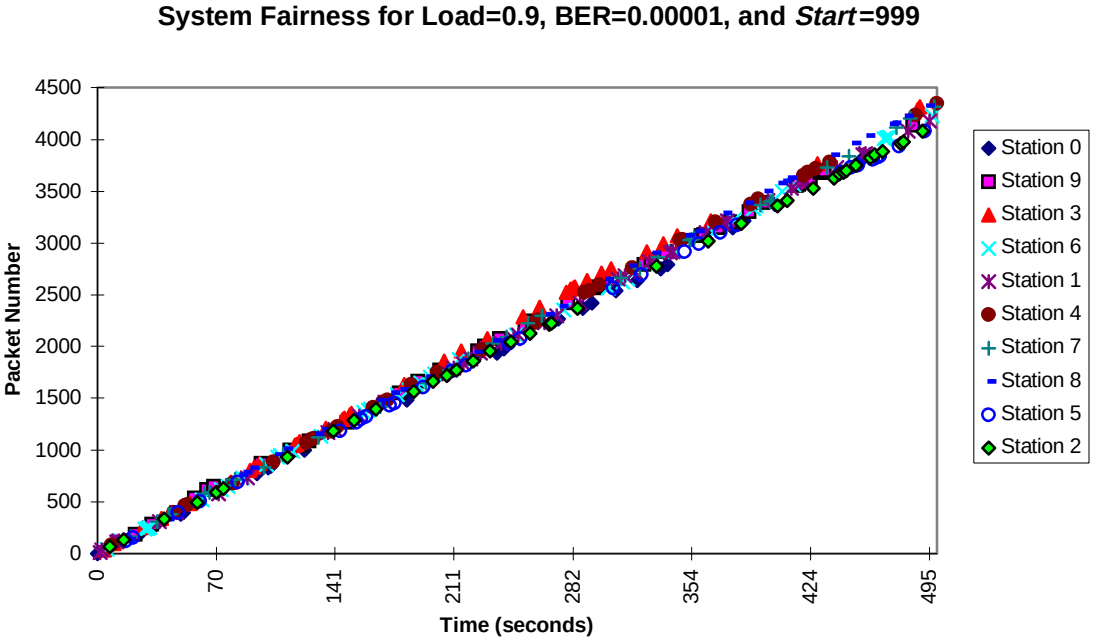

Figure F-18. System Fairness for Load=0.9, BER=0.00001, and *Start*=999

Appendix G. TCP Delay Plots

This appendix contains all plots of TCP delay. An explanation of these plots is found in Chapter 6.

TCP Delay using CATER

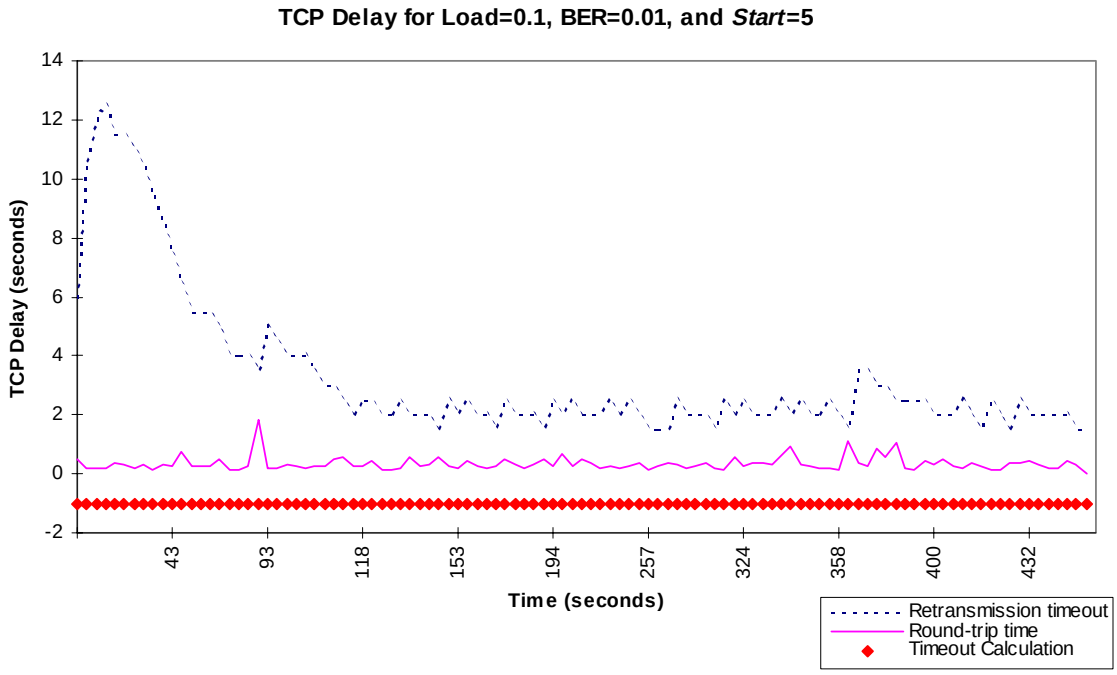

Figure G-1. TCP Delay for Load=0.1, BER=0.01, and *Start*=5

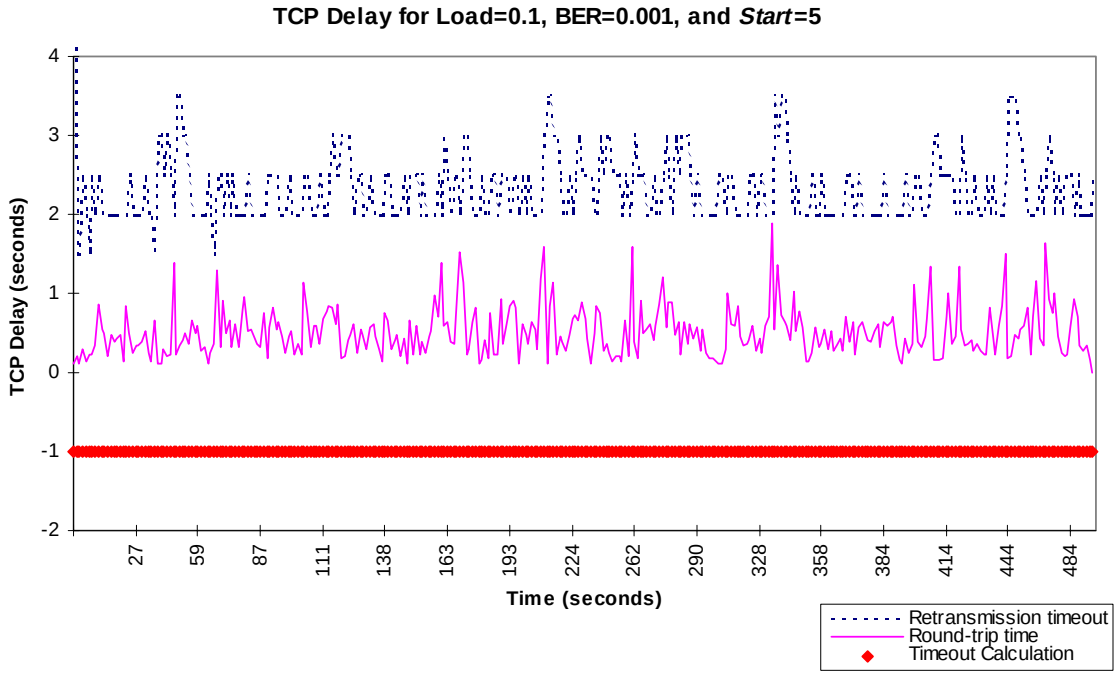

Figure G-2. TCP Delay for Load=0.1, BER=0.001, and *Start*=5

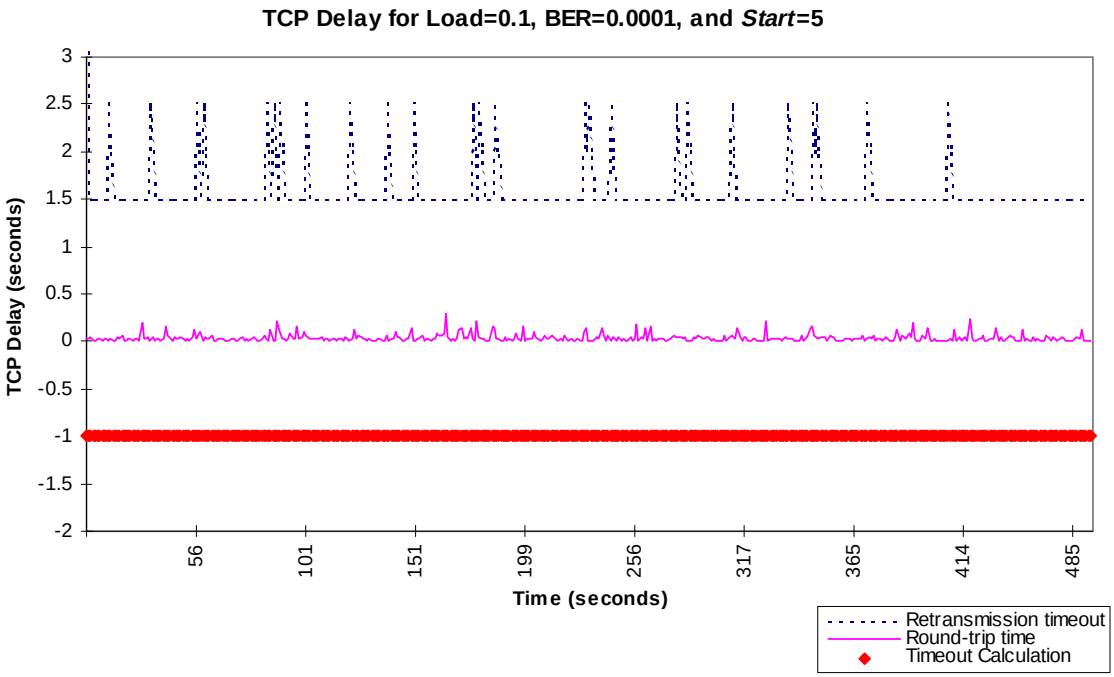

Figure G-3. TCP Delay for Load=0.1, BER=0.0001, and *Start*=5

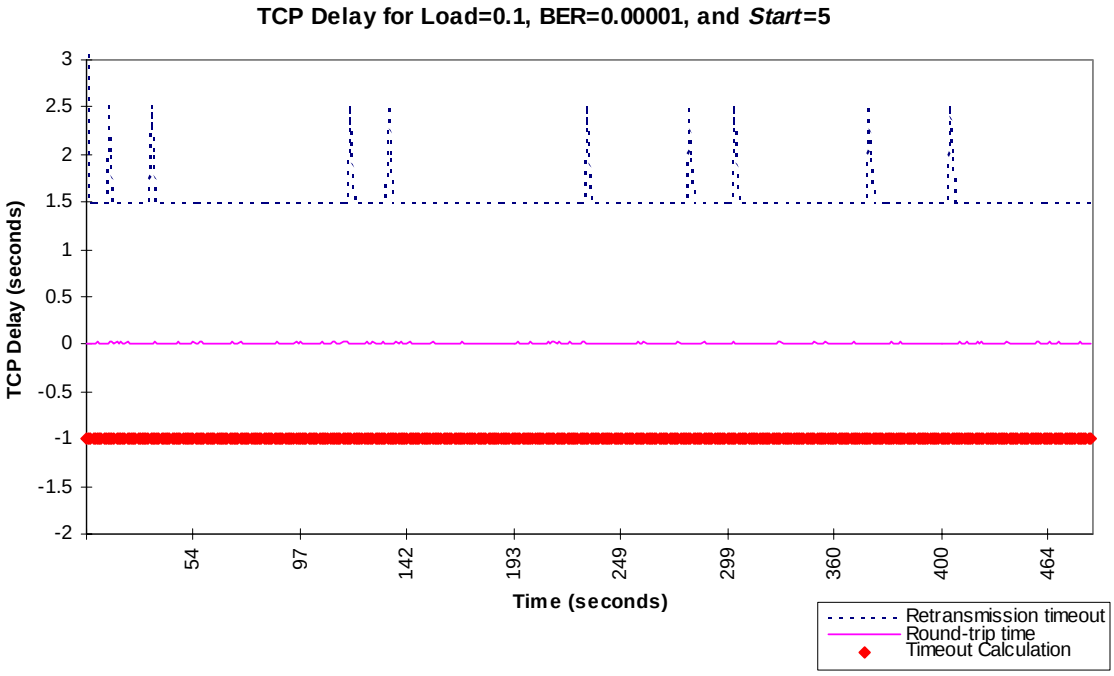

Figure G-4. TCP Delay for Load=0.1, BER=0.00001, and *Start*=5

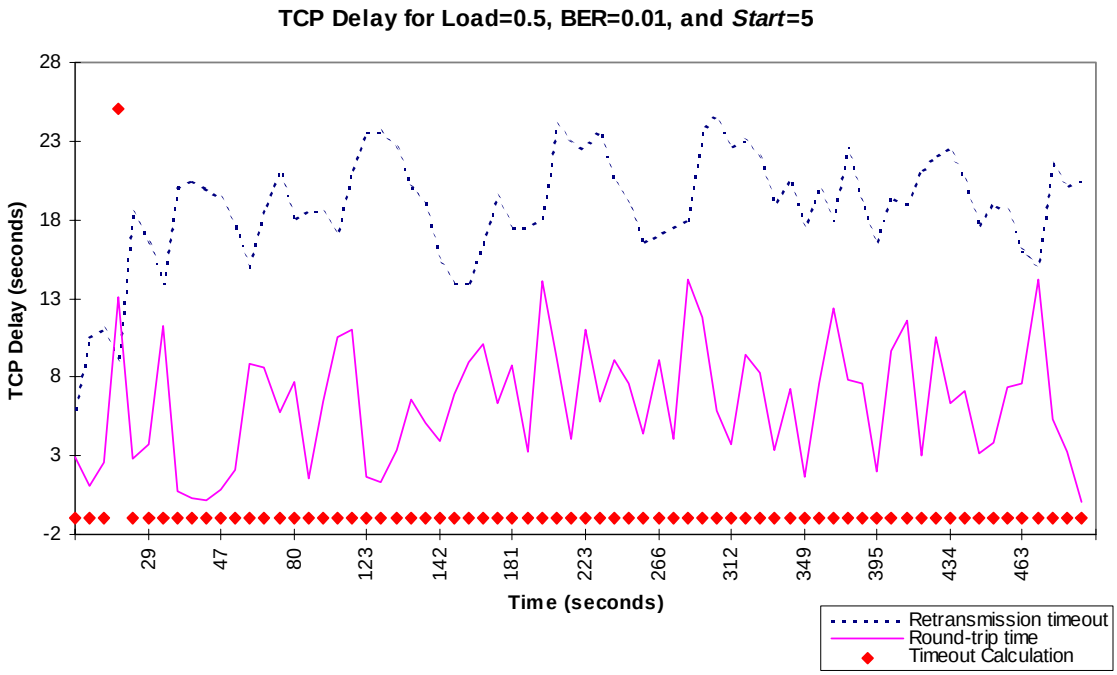

Figure G-5. TCP Delay for Load=0.5, BER=0.01, and *Start*=5

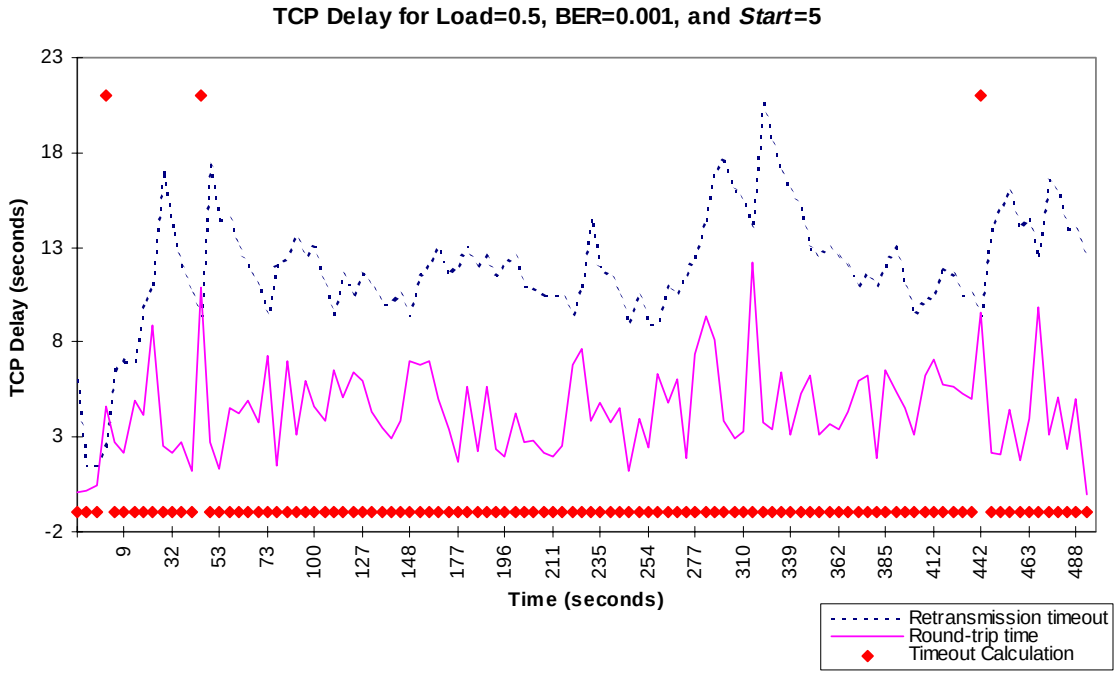

Figure G-6. TCP Delay for Load=0.5, BER=0.001, and *Start*=5

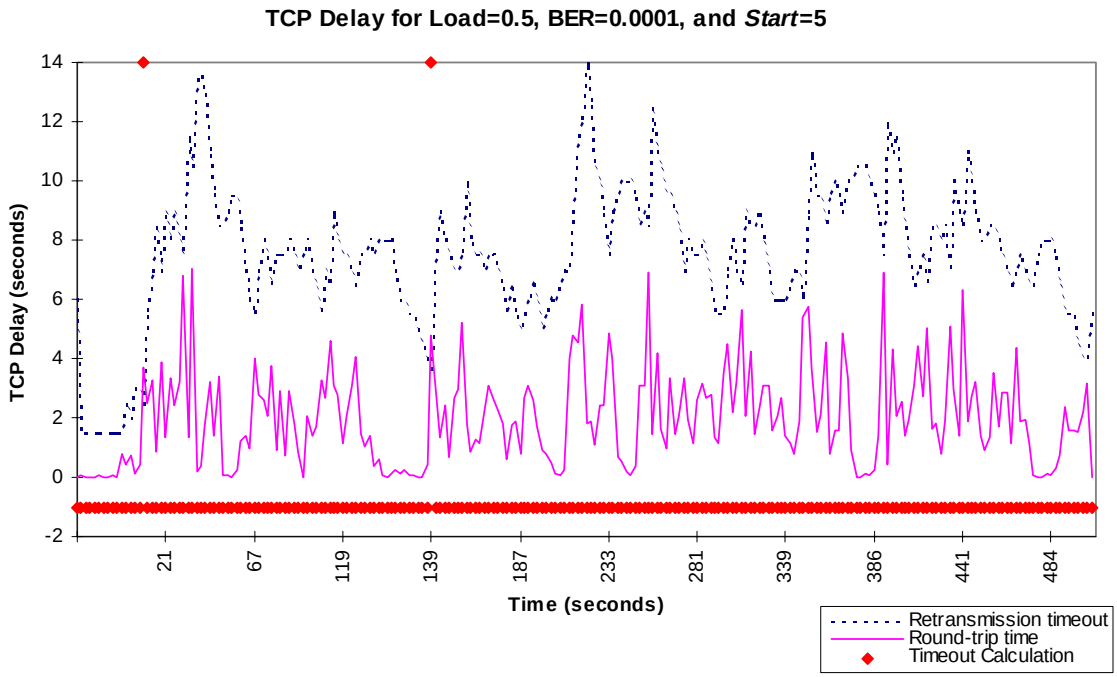

Figure G-7. TCP Delay for Load=0.5, BER=0.0001, and *Start*=5

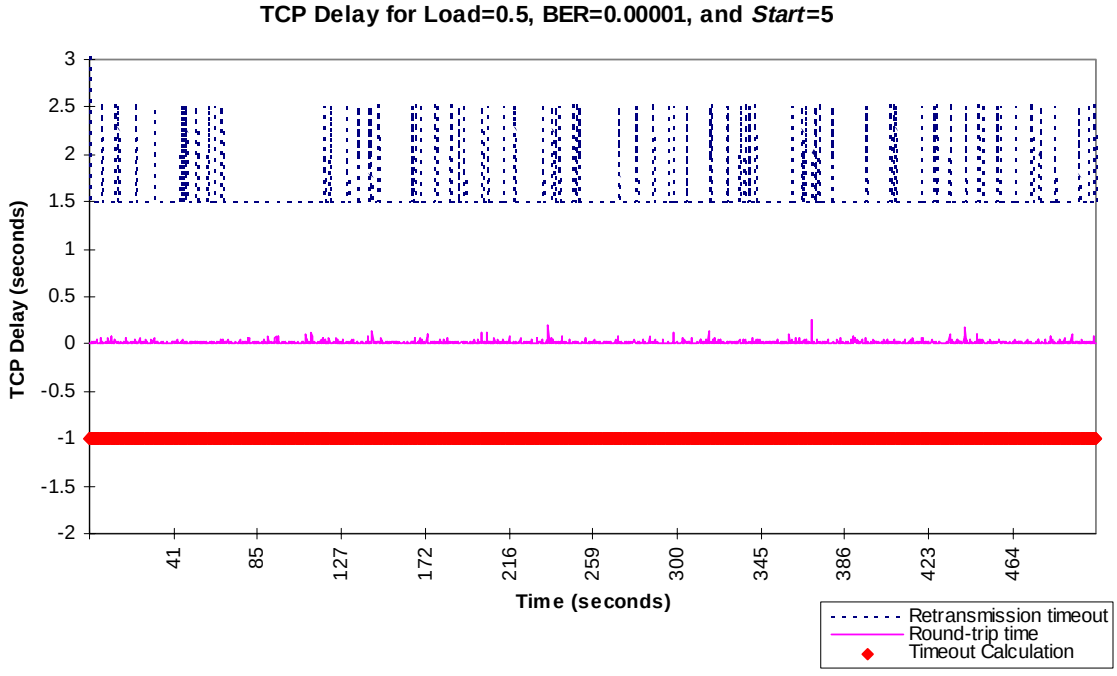

Figure G-8. TCP Delay for Load=0.5, BER=0.00001, and *Start*=5

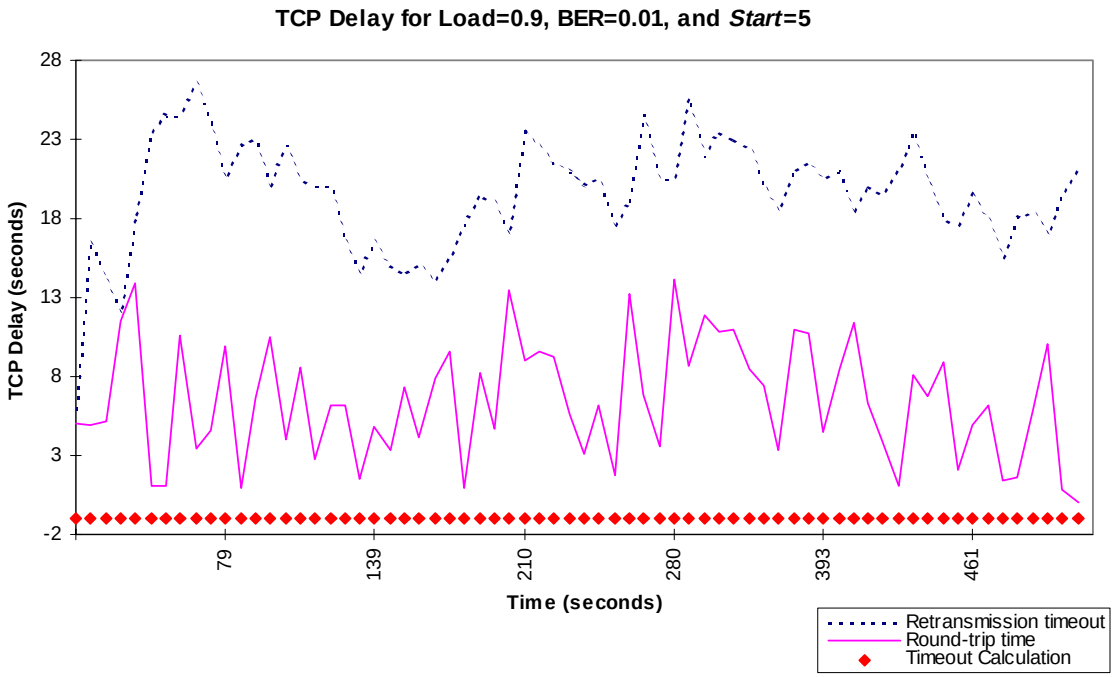

Figure G-9. TCP Delay for Load=0.9, BER=0.01, and *Start*=5

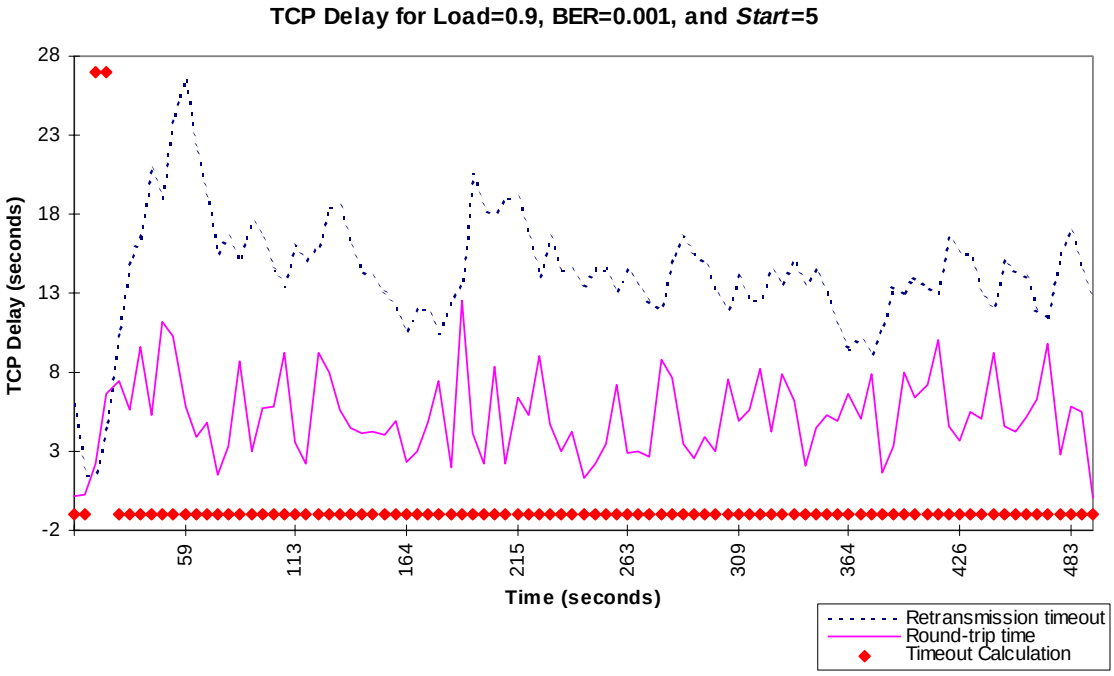

Figure G-10. TCP Delay for Load=0.9, BER=0.001, and *Start*=5

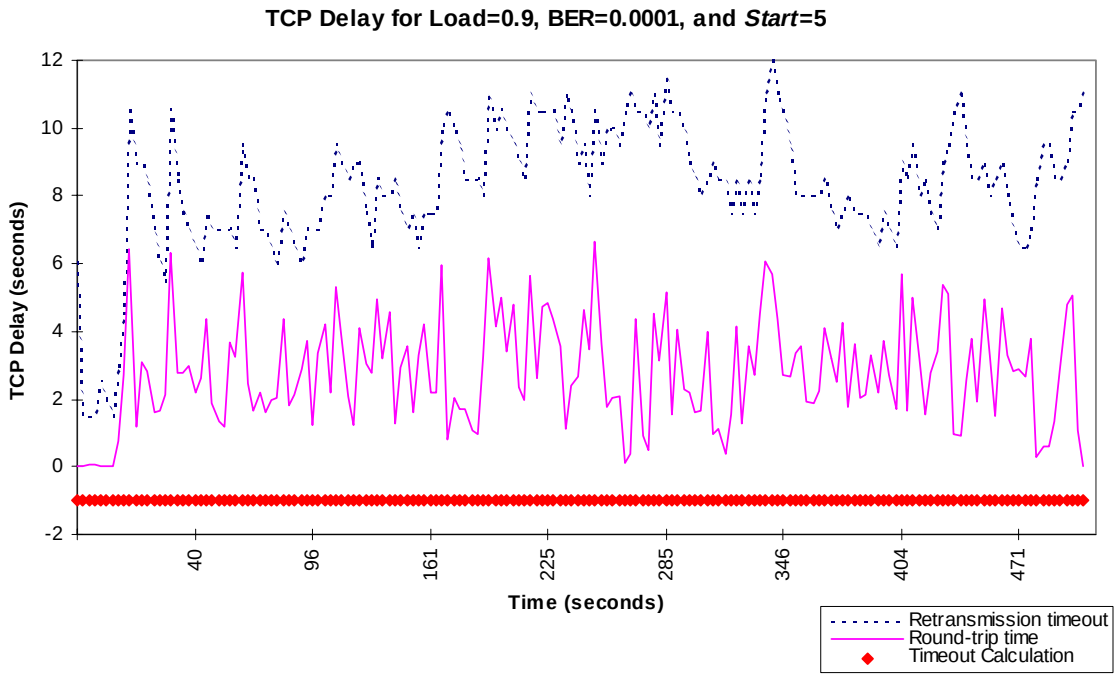

Figure G-11. TCP Delay for Load=0.9, BER=0 .0001, and *Start*=5

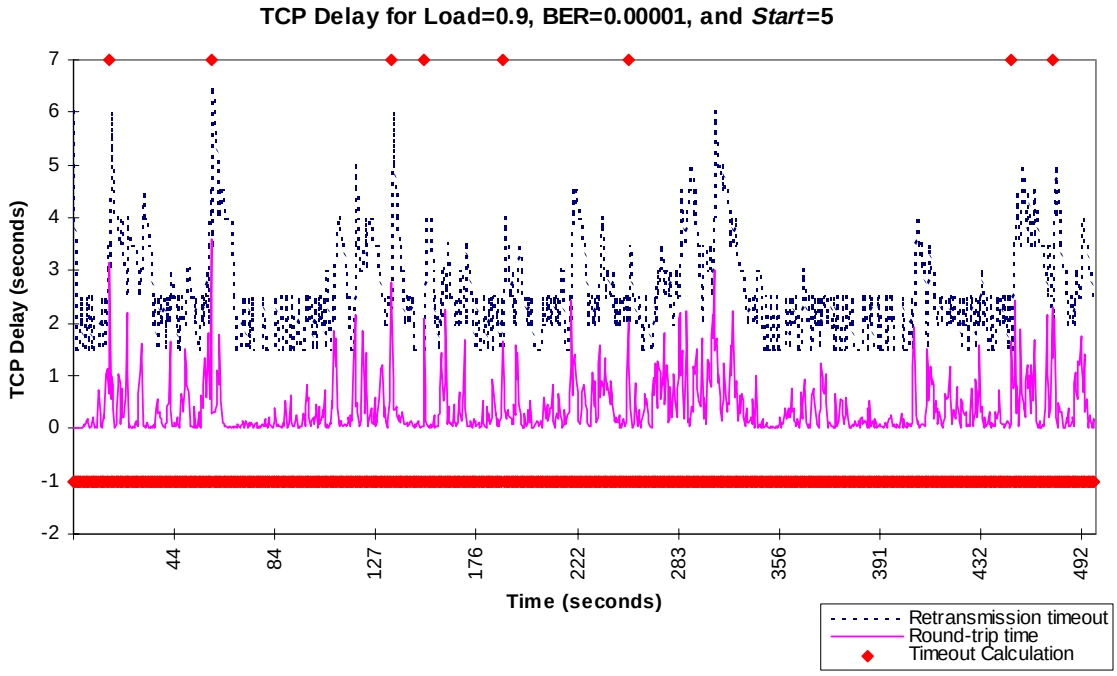

Figure G-12. TCP Delay for Load=0.9, BER=0.00001, and *Start*=5

TCP Delay using IEEE 802.11

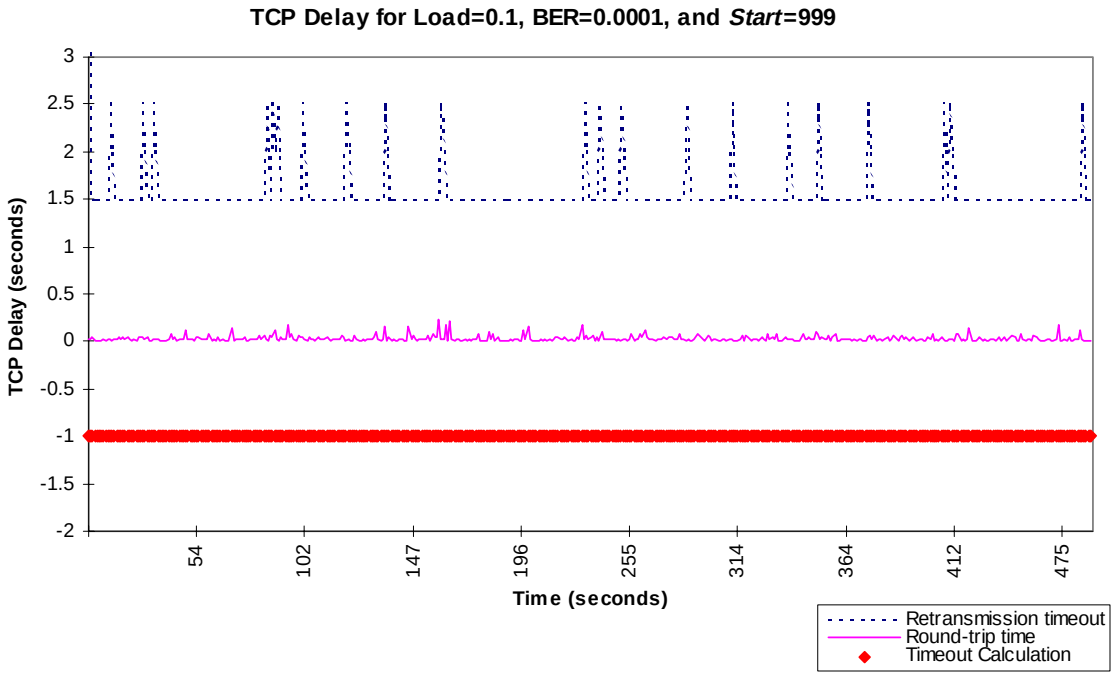

Figure G-13. TCP Delay for Load=0.1, BER=0.0001, and *Start*=999

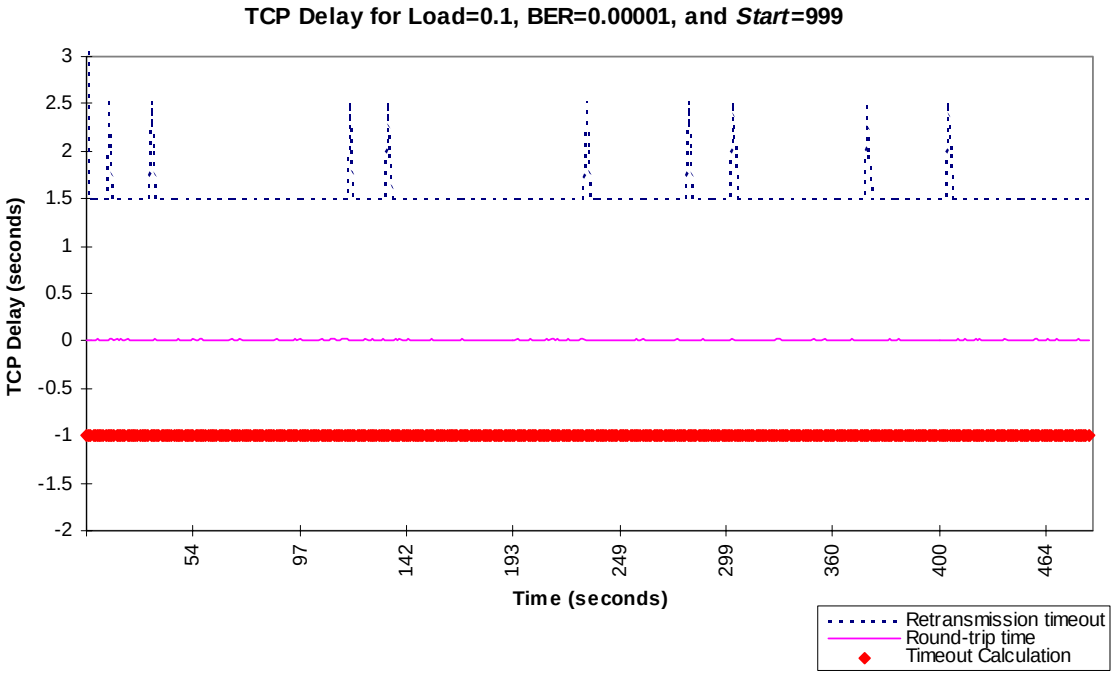

Figure G-14. TCP Delay for Load=0.1, BER=0.00001, and *Start*=999

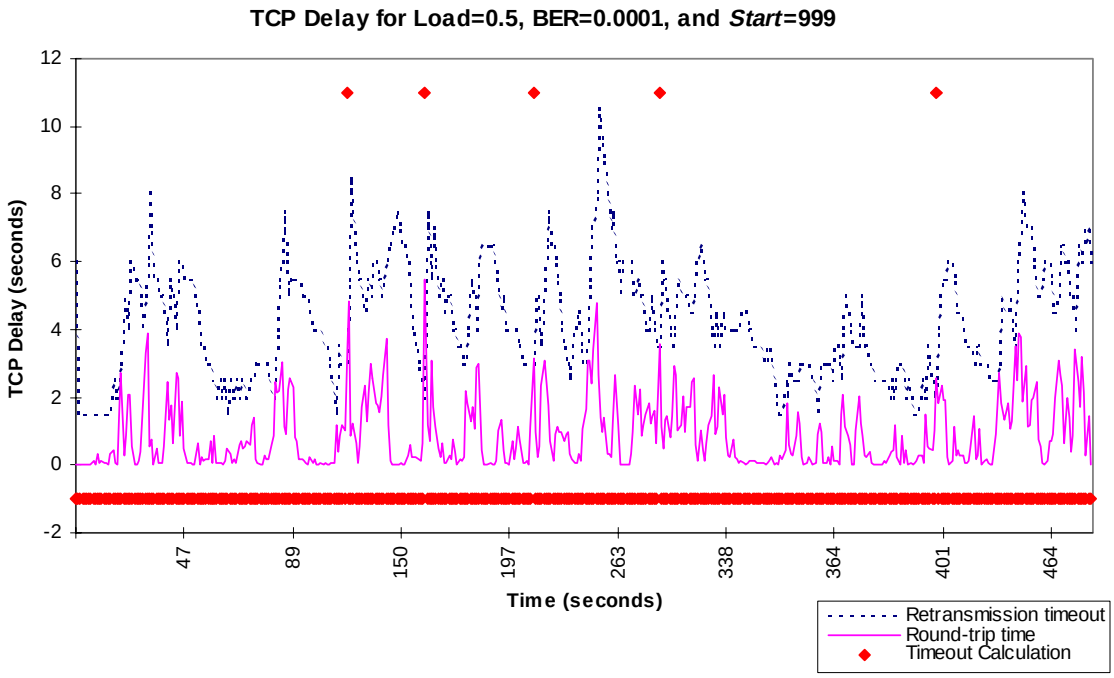

Figure G-15. TCP Delay for Load=0.5, BER=0.0001, and *Start*=999

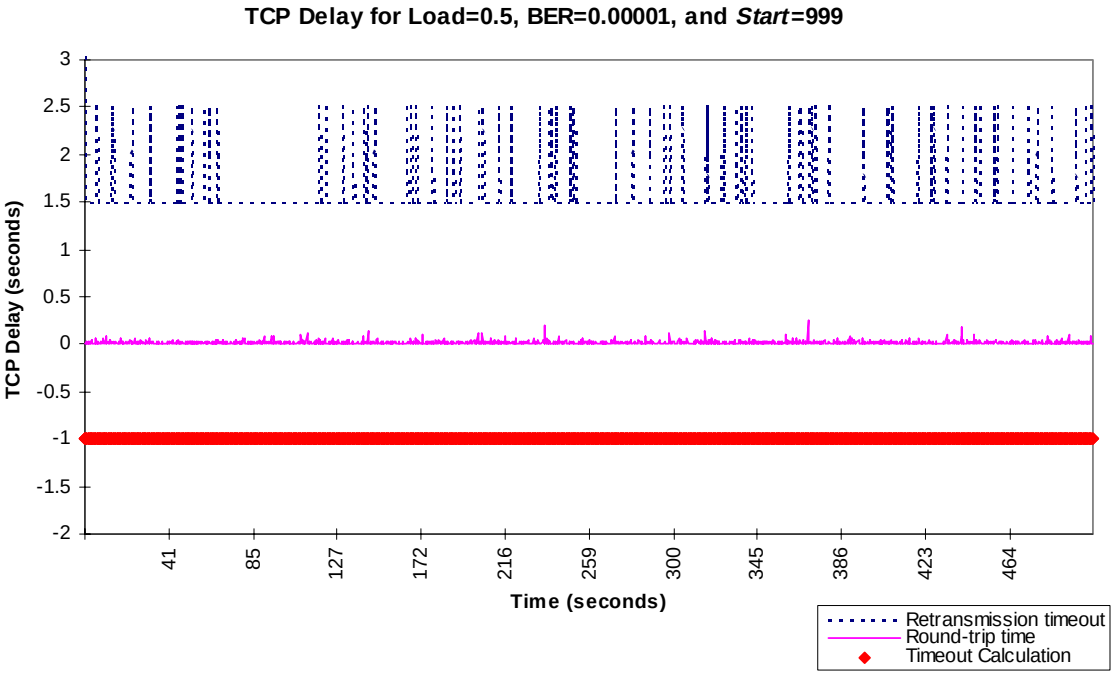

Figure G-16. TCP Delay for Load=0.5, BER=0.00001, and *Start*=999

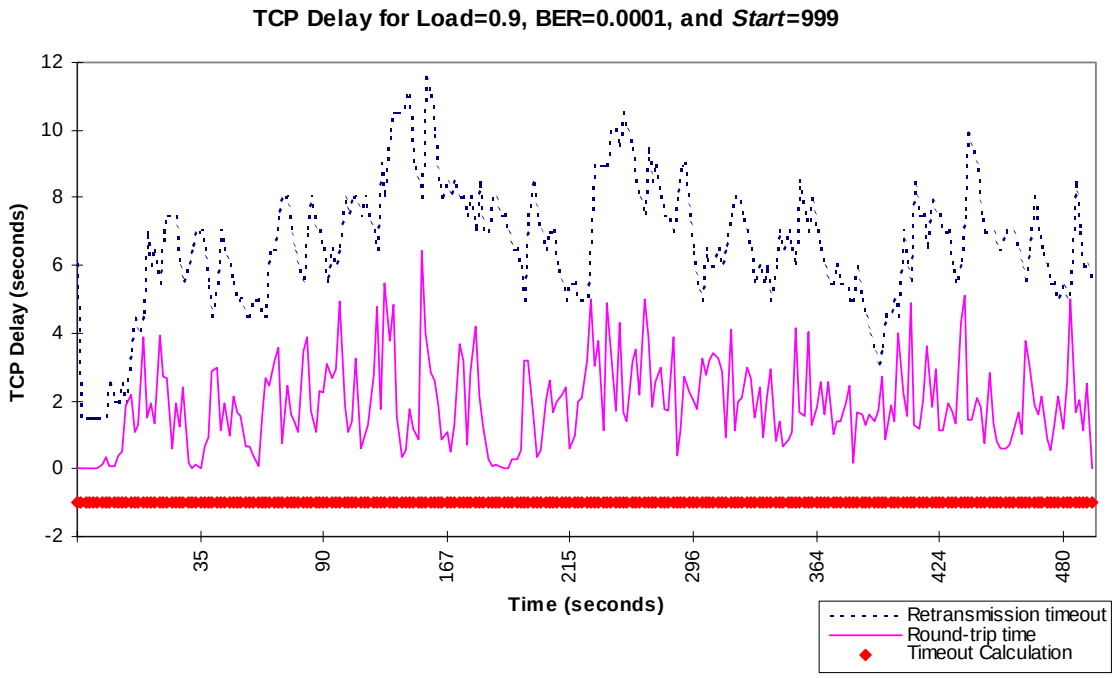

Figure G-17. TCP Delay for Load=0.9, BER=0.0001, and *Start*=999

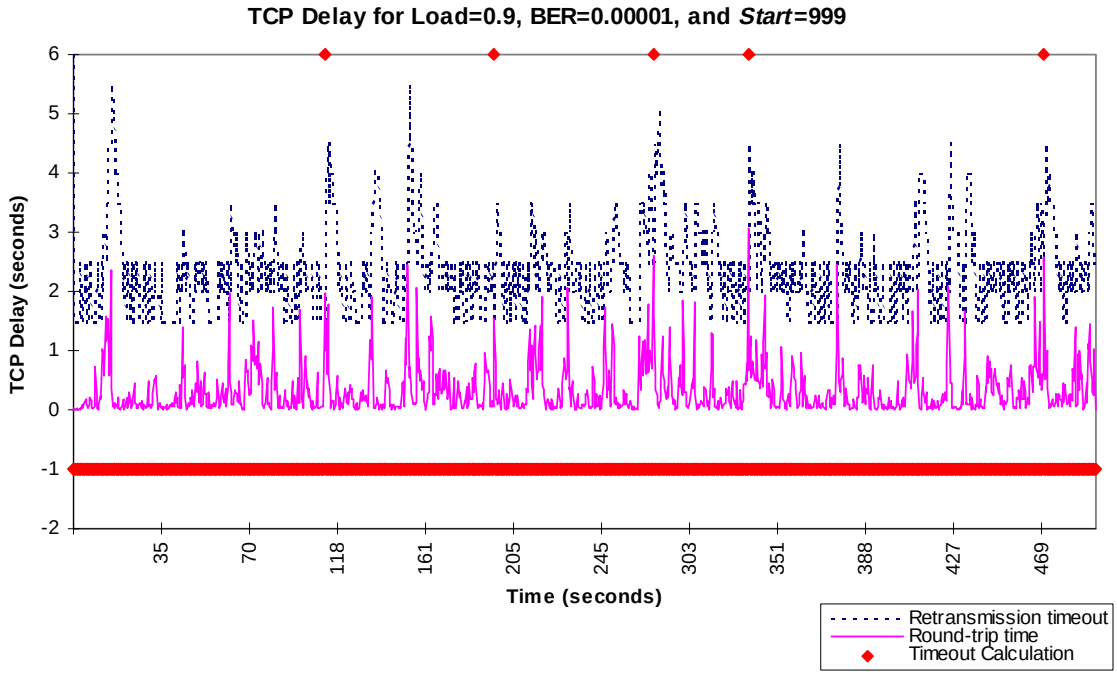

Figure G-18. TCP Delay for Load=0.9, BER=0.00001, and *Start*=999

Appendix H. Comparison of Aggregate Throughput Surface Plots

This appendix contains a comparison of aggregate throughput surface plots. The emulation plots are compared against the three station simulation plots. The graph on the left plots the emulation results, and the graph on the right plots the simulation results. An explanation of these plots is found in Chapter 7.

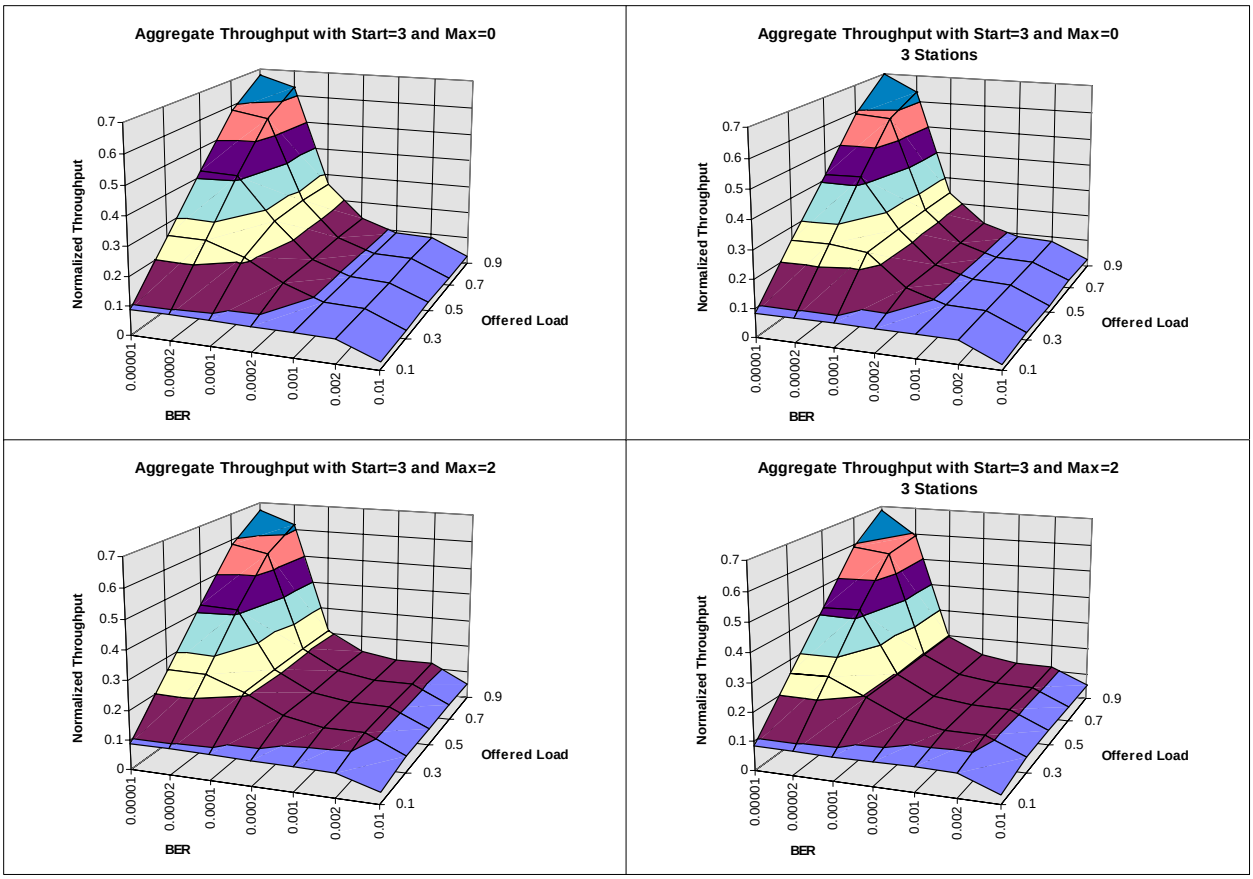

Figure H-1. Aggregate Throughput with *Start*=3 and *Max*=0 & 2

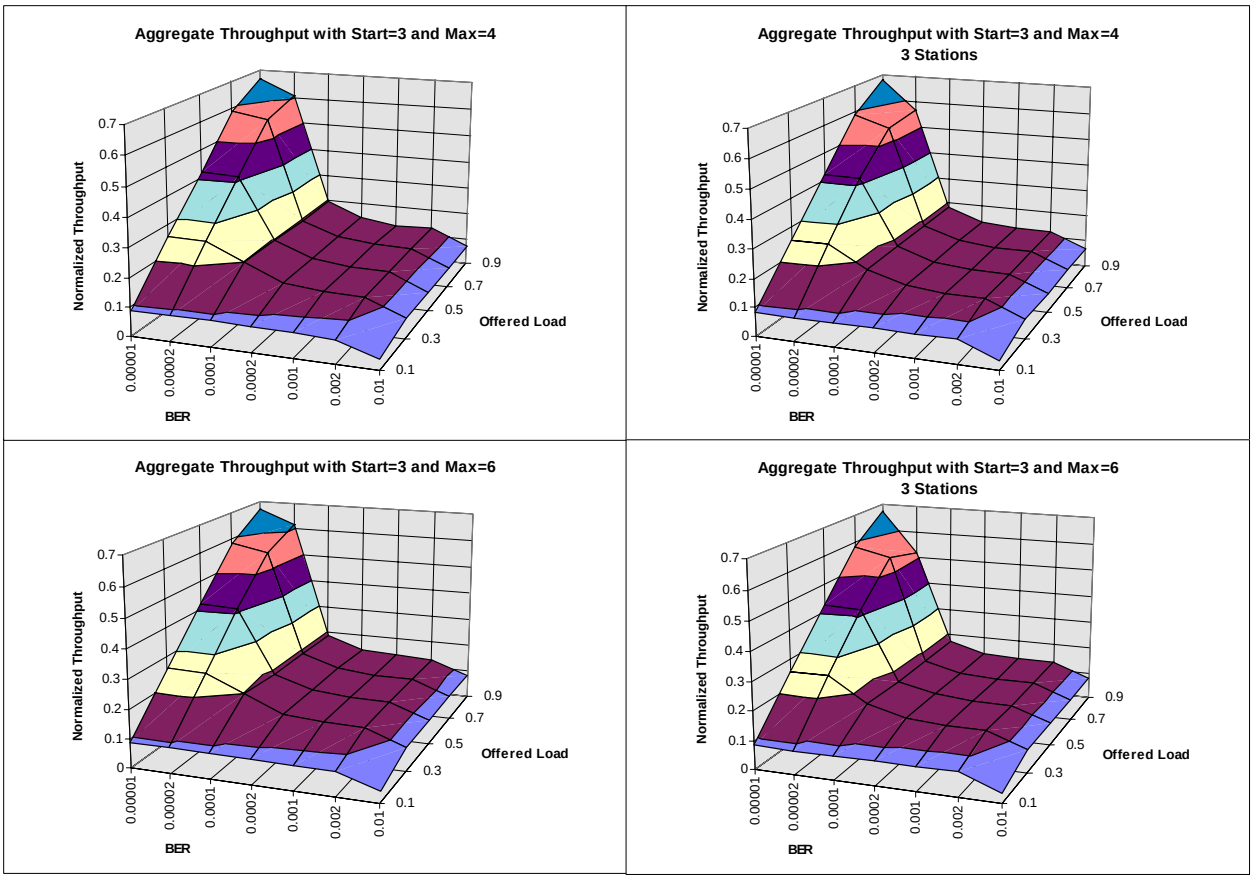

Figure H-2. Aggregate Throughput with *Start*=3 and *Max*=4 & 6

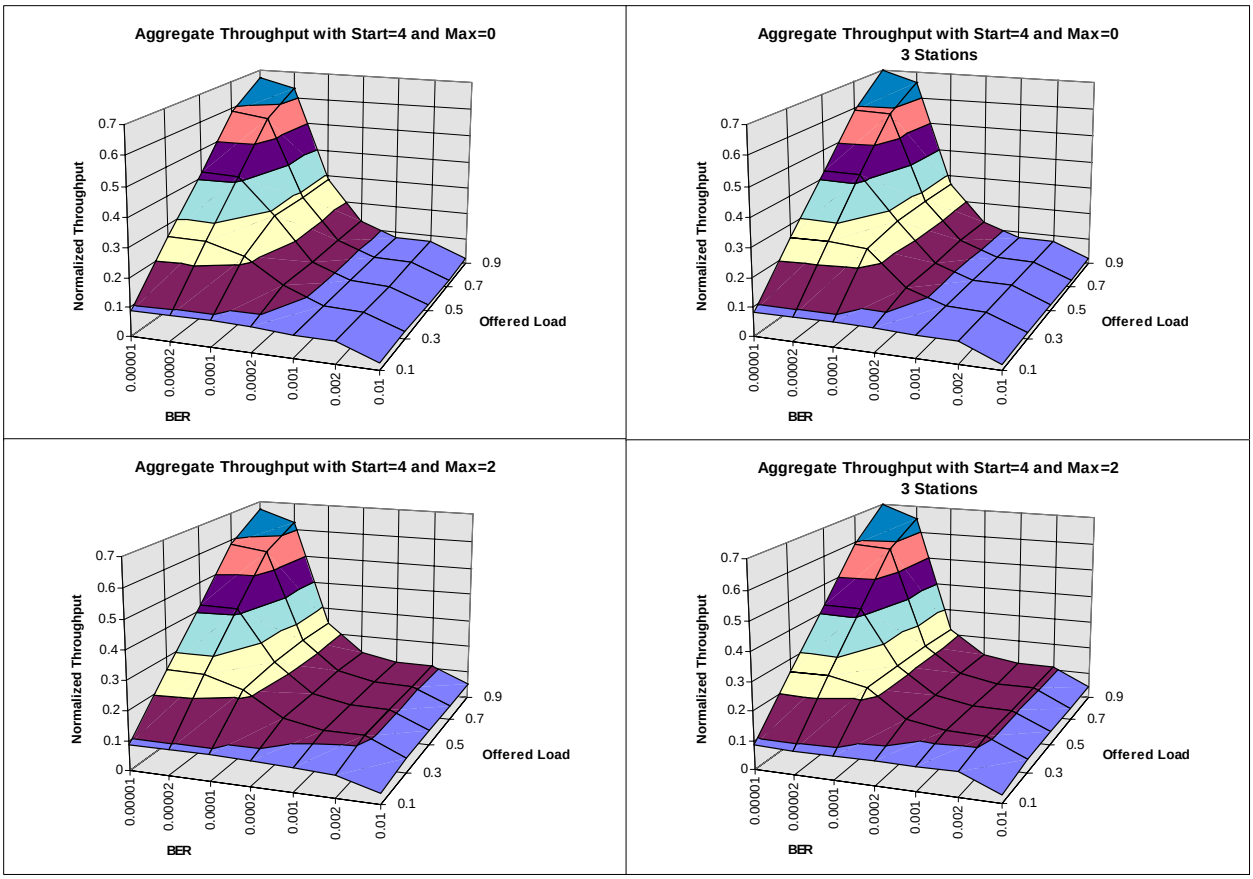

Figure H-3. Aggregate Throughput with *Start*=4 and *Max*=0 & 2

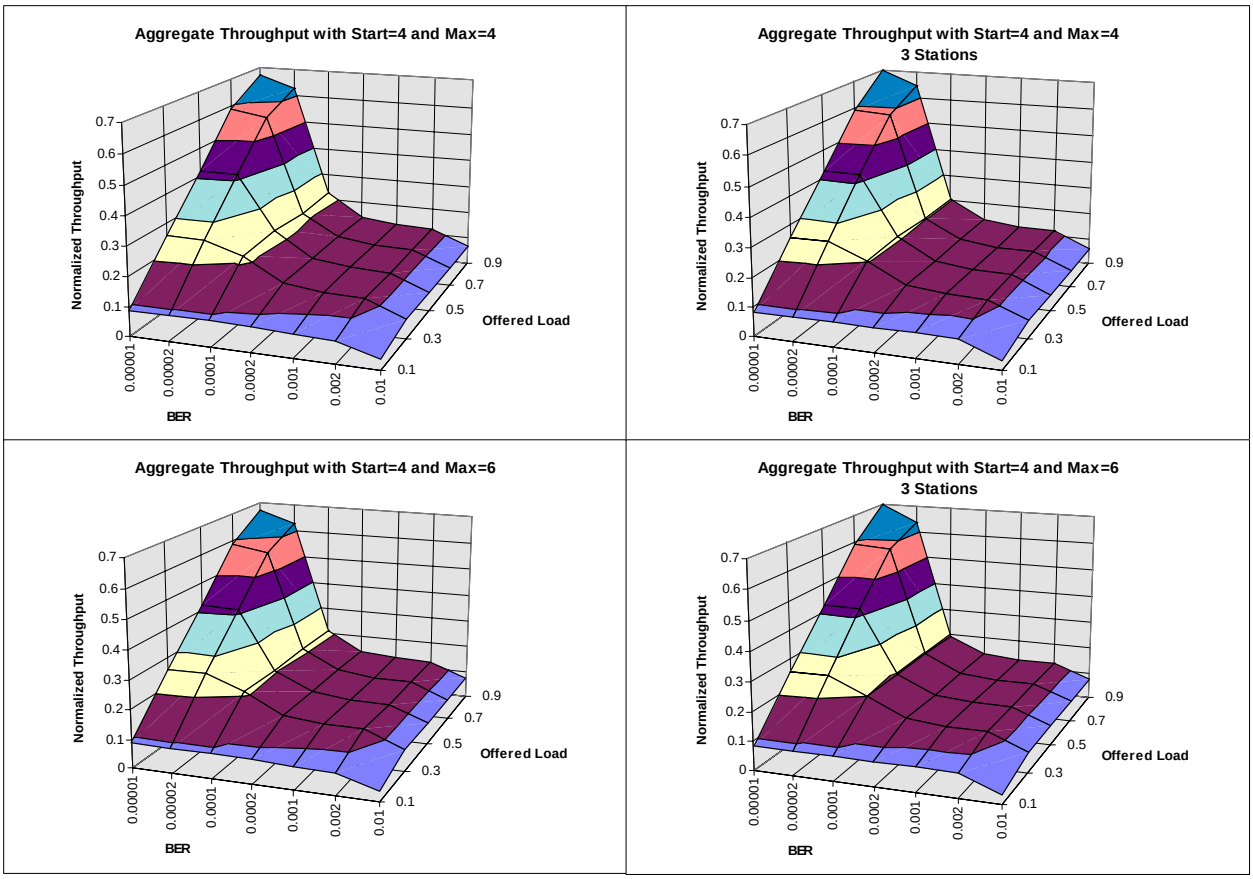

Figure H-4. Aggregate Throughput with *Start*=4 and *Max*=4 & 6

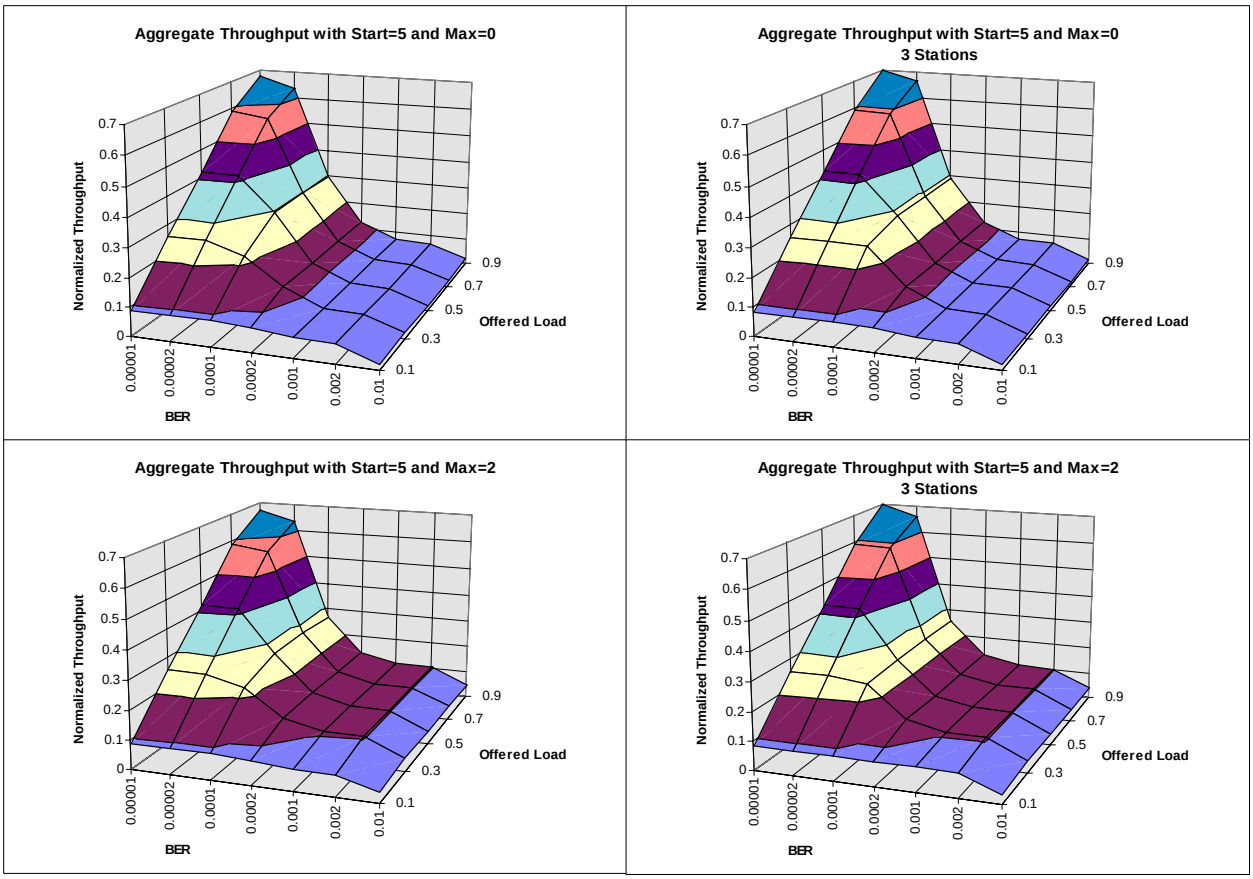

Figure H-5. Aggregate Throughput with *Start*=5 and *Max*=0 & 2

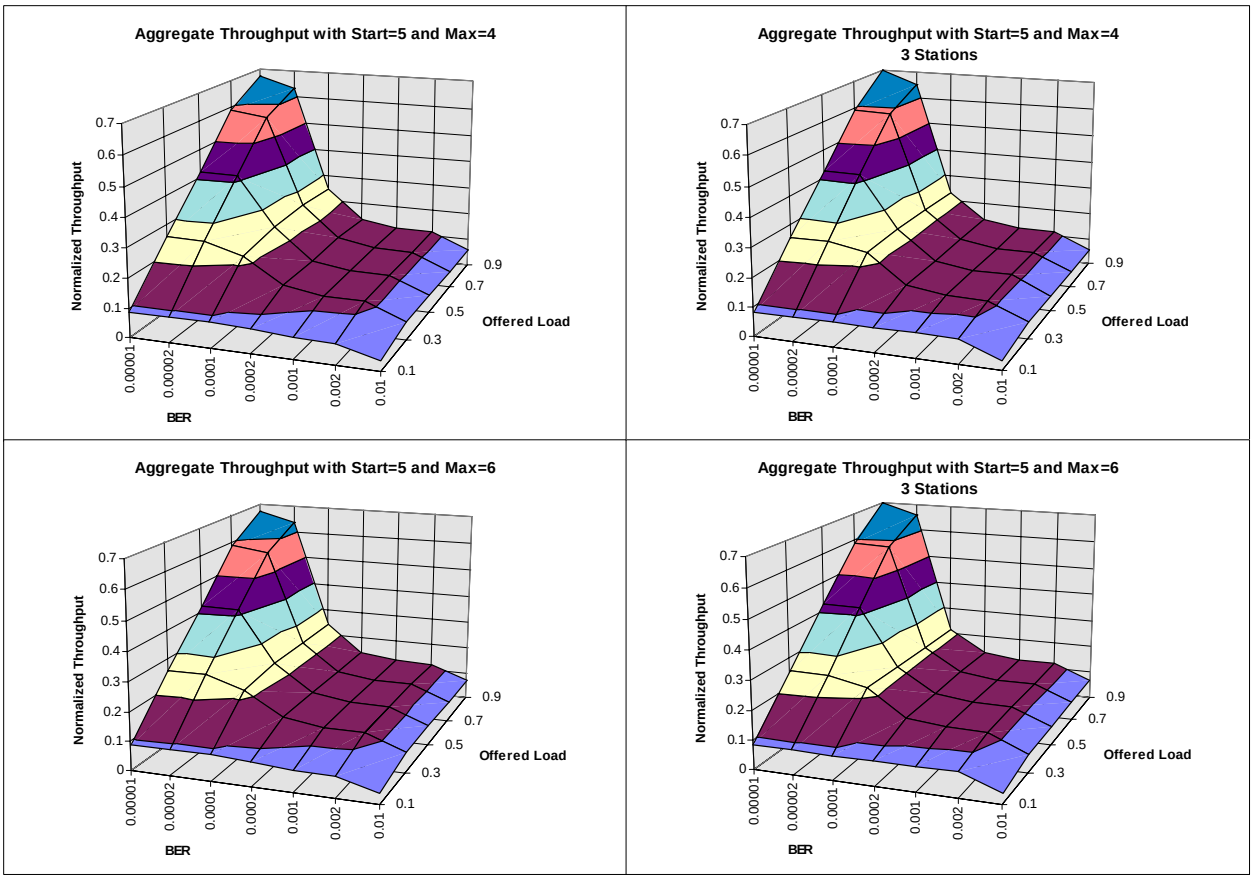

Figure H-6. Aggregate Throughput with *Start*=5 and *Max*=4 & 6

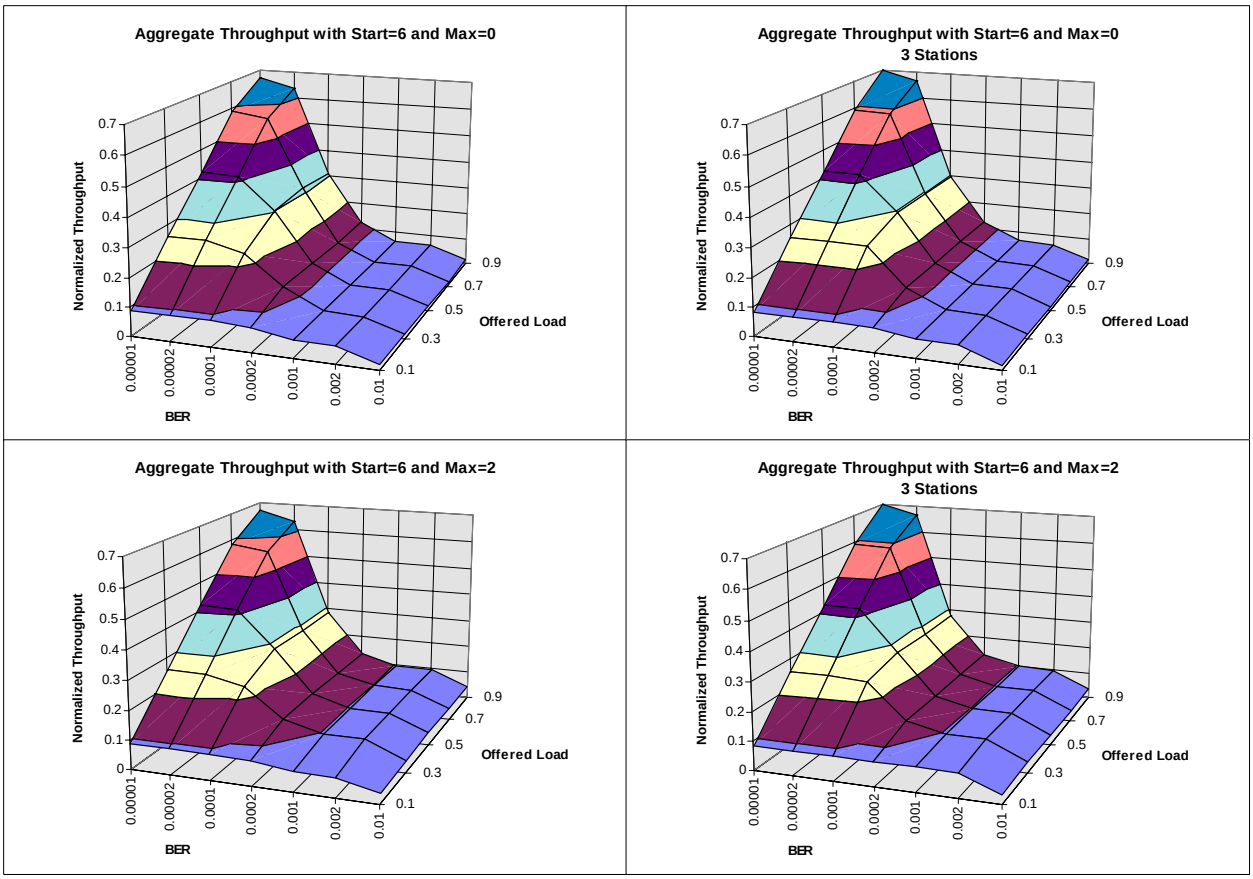

Figure H-7. Aggregate Throughput with *Start*=6 and *Max*=0 & 2

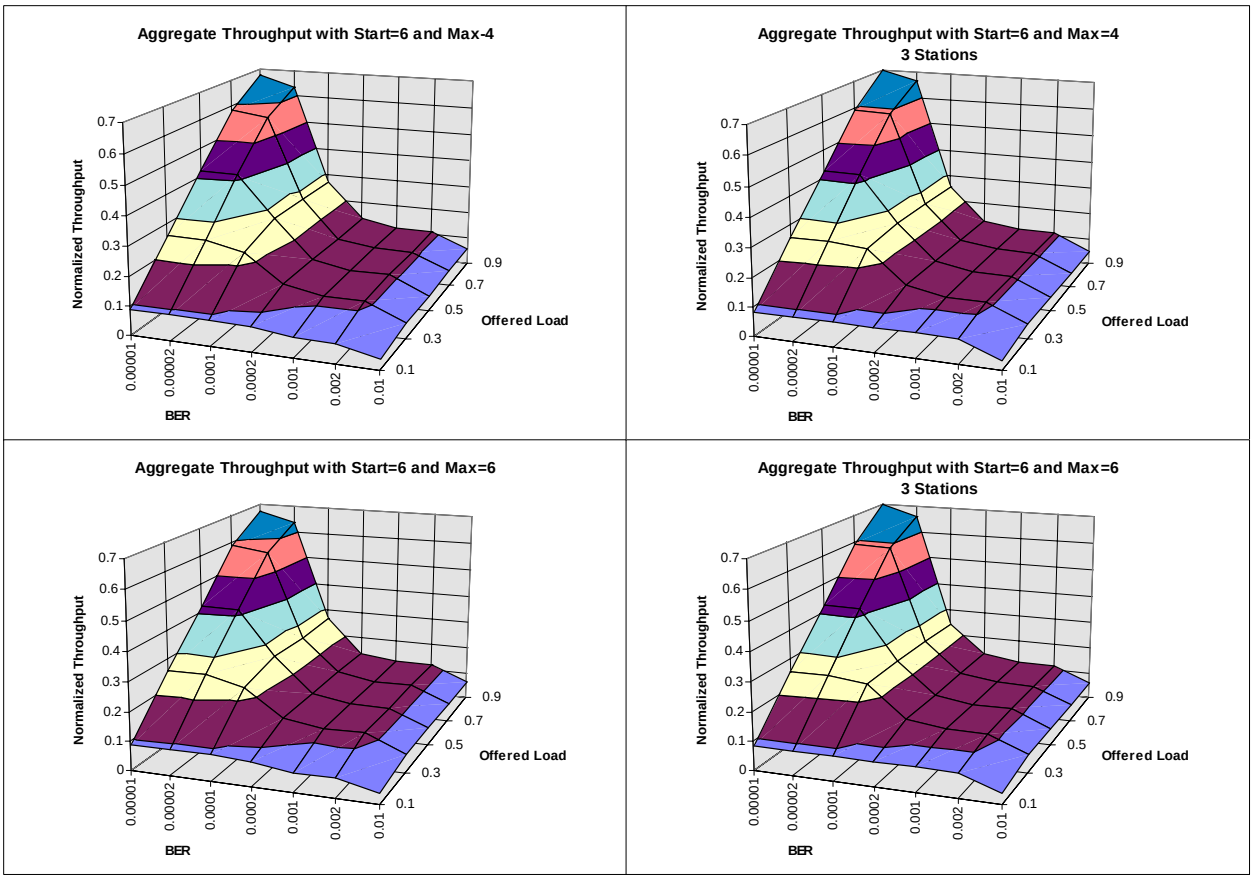

Figure H-8. Aggregate Throughput with *Start*=6 and *Max*=4 & 6

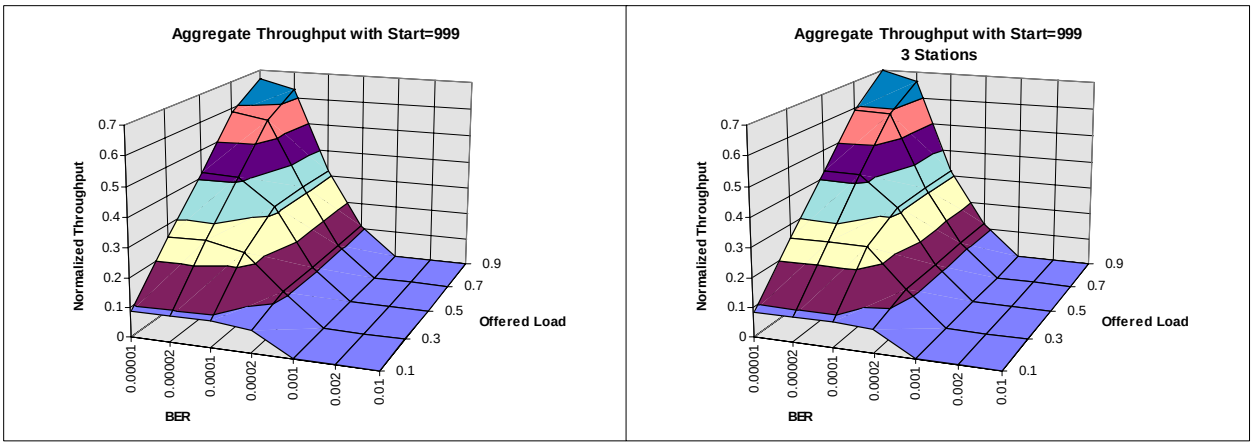

Figure H-9. Aggregate Throughput with *Start*=999

Appendix I. Relative Confidence Interval Half Widths for Emulation Data

This appendix contains all relative confidence interval half width tables for the emulation data. An explanation of these tables is found in Chapter 7.

Relative Confidence Interval Half Widths for *Start*=3

Table I-1. Relative Confidence Interval Half Widths for *Start*=3 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0.077499	0.155791	0.578851	0.698205	1.197556	2.419381
0.3	0.032384	0.026547	1.087755	1.314532	1.055204	0.573462	3.891805
0.5	0.055245	0.179707	1.755922	2.856431	1.549506	0.803322	2.821177
0.7	0.313841	0.566987	1.368692	2.195756	0.523152	1.429872	7.788818
0.9	0.527841	0.553669	1.391934	2.133039	1.101261	0.547995	3.021143

Table I-2. Relative Confidence Interval Half Widths for *Start*=3 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0.077555	0	0.123029	0.43981	0.455506	0.480898	3.56741
0.3	0.059148	0.106017	1.352938	2.103547	1.414196	1.034478	6.289234
0.5	0.179571	0.321234	2.384686	1.846729	0.72944	0.706264	4.572749
0.7	0.403343	0.536483	3.318656	1.403401	0.607884	0.823059	3.409833
0.9	0.579422	0.808189	4.035822	1.318157	1.373144	1.848732	8.089435

Table I-3. Relative Confidence Interval Half Widths for *Start*=3 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.291352	0.386409	0.54122	0.244332	4.176932
0.3	0.064785	0.177884	1.595937	0.833833	1.331753	1.238262	3.817467
0.5	0.137964	0.431201	3.556295	1.880875	1.04954	1.831299	8.232548
0.7	0.225081	0.313588	2.365939	0.909637	1.503678	1.787725	5.141712
0.9	0.572112	1.332639	2.36827	1.456991	1.342987	0.845703	6.551131

Table I-4. Relative Confidence Interval Half Widths for *Start*=3 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.145782	0.230556	0.465681	0.568569	3.257725
0.3	0.032384	0.236044	1.169487	1.659718	1.120007	1.796308	4.396748
0.5	0.074672	0.307319	2.957058	2.191593	1.820243	1.166874	5.085188
0.7	0.123989	0.246218	1.804787	1.502156	1.175167	1.190977	6.78544
0.9	0.776573	1.978022	3.912263	1.151241	1.076438	1.20855	5.053225

Table I-5. Relative Confidence Interval Half Widths for *Start*=3 and *Max*=8

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0.077555	0.145782	0.459918	0.631526	0.829459	4.19241
0.3	0.026455	0.132752	2.203957	0.607101	0.478659	1.984893	2.968589
0.5	0.179845	0.180625	2.646518	1.63	1.812553	1.37464	3.991689
0.7	0.48069	0.804201	1.106244	1.460784	1.224637	1.070384	5.457962
0.9	0.550721	1.894039	1.420702	0.796933	1.388557	2.368549	3.398385

Table I-6. Relative Confidence Interval Half Widths for *Start*=3 and *Max*=10

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0.077555	0.095402	0.458049	0.553808	0.514936	2.019849
0.3	0.05289	0.145423	2.189031	1.268142	1.631484	1.254141	3.259956
0.5	0.110558	0.241328	2.730364	1.270717	0.999997	0.392526	3.949815
0.7	0.376465	0.916448	1.695386	0.627962	1.137848	0.773192	5.252344
0.9	0.718281	2.318155	4.021196	0.99618	1.533925	1.441307	3.89027

Relative Confidence Interval Half Widths for *Start*=4

Table I-7. Relative Confidence Interval Half Widths for *Start*=4 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.155507	0.376647	1.466667	1.099369	5.337102
0.3	0	0.052956	0.584891	2.1857	1.136201	1.11849	5.898439
0.5	0.053982	0.171408	0.744698	2.269683	0.50865	1.052318	4.97063
0.7	0.325056	0.306599	3.938546	3.079946	0.276878	0.643871	4.020797
0.9	0.142826	0.595313	2.849988	4.40442	0.843415	1.34632	4.412797

Table I-8. Relative Confidence Interval Half Widths for *Start*=4 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.233687	0.125316	0.494385	0.870111	6.270529
0.3	0.026438	0.113999	1.391821	1.171976	1.195489	1.069862	8.942708
0.5	0.068108	0.213525	1.584916	2.511404	0.71164	0.773709	4.261144
0.7	0.500937	0.479582	4.325155	1.8511	0.696138	1.09758	5.789592
0.9	0.394101	0.871477	4.122498	1.155686	2.36901	1.17531	6.453097

Table I-9. Relative Confidence Interval Half Widths for *Start*=4 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.145623	0.269473	0.540439	0.609701	5.162774
0.3	0.026438	0.077205	1.206607	1.842529	0.846931	1.457522	5.265218
0.5	0.054035	0.065833	3.240493	0.561796	1.070685	1.389661	3.350114
0.7	0.341425	0.430082	3.588503	2.129675	0.856667	0.53829	3.279402
0.9	0.275514	0.735493	1.82101	0.549395	1.132498	1.273253	5.55112

Table I-10. Relative Confidence Interval Half Widths for *Start*=4 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.095367	0.341403	0.439357	0.49589	0.744154
0.3	0	0.170678	2.835304	1.114859	1.070842	1.338081	5.423012
0.5	0.051505	0.198682	3.640425	1.922744	0.581812	0.773689	8.283108
0.7	0.245199	0.848184	1.357089	1.202849	1.024138	1.453814	5.067014
0.9	0.274023	0.617568	2.241653	1.451616	0.655416	1.923427	6.226545

Table I-11. Relative Confidence Interval Half Widths for *Start*=4 and *Max*=8

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.145516	0.570469	0.416073	0.600246	4.47615
0.3	0.032384	0.107621	3.465466	1.15952	1.879636	1.397513	4.479637
0.5	0.090711	0.213525	5.01693	1.486322	0.924575	1.630895	4.447635
0.7	0.346681	0.496164	2.692311	1.825793	1.448312	1.142814	5.383854
0.9	0.26729	0.697853	2.529836	1.90932	1.258465	1.914481	5.48651

Table I-12. Relative Confidence Interval Half Widths for *Start*=4 and *Max*=10

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.173989	0.384067	0.721758	0.623652	4.473086
0.3	0	0.067548	0.954601	1.708604	0.503841	0.977826	6.229971
0.5	0.055203	0.206853	3.572323	1.107056	1.417985	0.793592	5.380338
0.7	0.302574	0.280972	2.427371	1.166751	1.033275	2.655576	5.575705
0.9	0.462359	0.592947	3.092396	1.168755	1.392084	1.240131	6.939915

Relative Confidence Interval Half Widths for *Start*=5

Table I-13. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.145516	0.691555	0.837899	1.190861	6.806251
0.3	0	0.099059	0.921549	2.898659	0.992659	1.30231	7.405719
0.5	0.059825	0.117201	2.039831	2.666184	1.305433	0.836899	7.938899
0.7	0.308116	0.153413	1.623789	2.125852	1.039683	1.069096	6.124259
0.9	0.553294	0.29768	2.525018	1.409753	0.906555	0.790052	4.957911

Table I-14. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.145623	0.328288	0.482946	0.593445	4.12283
0.3	0	0.089791	0.869875	2.599659	1.951135	0.671648	4.370878
0.5	0.119705	0.117291	2.369517	1.605748	0.689079	2.341043	3.479329
0.7	0.179469	0.29002	2.732438	1.613495	0.970021	1.474041	6.933162
0.9	0.593042	0.286176	1.846691	2.193961	0.71179	1.118531	6.038657

Table I-15. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.123029	0.418419	0.334269	0.180971	7.796383
0.3	0.026438	0.125549	1.83635	2.807955	1.580727	1.271989	7.179251
0.5	0.082916	0.155051	3.243405	3.243811	1.704944	1.027172	11.59768
0.7	0.212751	0.473335	3.089321	0.94293	1.071315	0.796728	6.03598
0.9	0.527436	0.467891	2.665968	1.906802	0.460517	1.317045	4.445926

Table I-16. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.233687	0.37276	0.679094	1.298326	7.538875
0.3	0.032384	0.032409	1.98629	1.959581	0.748078	1.666562	6.345448
0.5	0.110457	0.188325	4.310451	2.744284	1.364531	1.873054	3.454533
0.7	0.252788	0.356571	4.60577	1.494533	0.577421	0.855319	9.274696
0.9	0.34723	0.540598	3.016414	2.485182	1.68529	1.781526	6.933268

Table I-17. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=8

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.26406	0.236702	0.642378	0.515946	2.05447
0.3	0	0.110792	1.426366	0.930372	1.529901	1.641497	8.918796
0.5	0.081405	0.242987	2.159137	1.364195	0.674029	1.682756	2.409384
0.7	0.152129	0.491519	2.624557	1.529613	0.973505	1.475848	2.416967
0.9	0.395782	0.471021	3.034807	1.896362	1.15673	1.747266	5.116234

Table I-18. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=10

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.145516	0.579799	0.589652	0.565586	4.138858
0.3	0	0.067497	1.388432	1.440541	1.266953	0.803485	4.658785
0.5	0.060932	0.183003	1.704581	1.110151	0.822481	1.339595	7.778201
0.7	0.260429	0.242126	4.044597	1.918563	1.761835	1.153851	7.071011
0.9	0.466876	0.580561	3.345368	1.474763	2.539125	1.247096	2.345844

Relative Confidence Interval Half Widths for *Start*=6

Table I-19. Relative Confidence Interval Half Widths for *Start*=6 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.077697	0.78498	0.676465	0.755453	2.55956
0.3	0.026438	0.032425	0.413684	2.869728	1.201588	0.50044	5.729931
0.5	0.104907	0.23303	0.957745	2.309682	0.639039	0.936238	5.17152
0.7	0.113997	0.262172	1.536925	2.087818	1.372985	1.944683	6.830371
0.9	0.389021	0.512061	2.168914	2.658952	0.526717	1.049669	1.376708

Table I-20. Relative Confidence Interval Half Widths for *Start*=6 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.145516	0.343031	0.847268	0.740541	6.494544
0.3	0.026438	0.049542	0.916466	2.066789	1.314534	1.444548	10.53803
0.5	0.074643	0.249486	3.86371	0.800408	1.10163	1.10728	5.709389
0.7	0.192906	0.187208	1.895613	2.067222	2.020439	0.66545	8.470528
0.9	0.385105	0.203117	3.357589	0.648978	0.356313	0.891784	7.117093

Table I-21. Relative Confidence Interval Half Widths for *Start*=6 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.145516	0.904558	0.366801	0.57792	8.554831
0.3	0.026438	0.089791	0.825731	2.118876	2.287959	0.632391	4.512583
0.5	0.078961	0.130999	0.369983	1.361255	1.720809	1.908461	7.861179
0.7	0.167227	0.172632	2.278986	1.41809	0.810135	1.067798	8.108535
0.9	0.656948	0.314384	2.67514	0.756336	1.11548	2.032952	11.43572

Table I-22. Relative Confidence Interval Half Widths for *Start*=6 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.123029	0.92591	0.667369	0.756634	3.186264
0.3	0.026438	0.064841	0.785669	2.198482	1.118086	1.081291	6.277062
0.5	0.08689	0.061444	1.075569	2.174698	1.385542	1.071038	6.163714
0.7	0.108247	0.12223	1.778904	2.051306	1.304952	1.836154	10.4577
0.9	0.086993	0.550303	3.054558	2.032365	0.793568	2.216803	6.7251

Table I-23. Relative Confidence Interval Half Widths for *Start*=6 and *Max*=8

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.095193	0.909871	0.858241	0.840391	5.938502
0.3	0	0.107554	1.356497	1.59563	2.050965	1.644868	7.853796
0.5	0.098347	0.259246	3.641205	1.660908	1.358776	1.501005	7.868487
0.7	0.42263	0.332133	4.08289	1.678736	2.247095	0.862777	7.560017
0.9	0.403784	0.414118	1.754103	3.188444	1.278588	1.439745	5.844803

Table I-24. Relative Confidence Interval Half Widths for *Start*=6 and *Max*=10

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.198233	0.381392	0.421012	1.054424	6.087104
0.3	0.032384	0.067506	0.65092	1.222695	1.019452	1.520481	5.978906
0.5	0.075567	0.095149	2.346862	2.163107	2.477902	1.782635	7.075593
0.7	0.203628	0.496991	3.096627	1.848818	1.117867	1.568069	10.4131
0.9	0.351879	0.133912	2.495406	1.406853	0.643185	2.708088	9.146874

Relative Confidence Interval Half Widths for *Start*=7

Table I-25. Relative Confidence Interval Half Widths for *Start*=7 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.145357	0.279473	0.592618	0.717066	9.190632
0.3	0	0.059184	0.832501	2.50365	0.848923	0.90603	6.786327
0.5	0.129847	0.189913	3.805184	2.891325	0.803974	0.906934	2.655926
0.7	0.109747	0.215692	0.804862	3.429421	0.894165	0.778618	3.36267
0.9	0.549076	0.611535	1.562566	2.406262	0.957044	0.644838	5.357811

Table I-26. Relative Confidence Interval Half Widths for *Start*=7 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.077697	1.038746	1.229332	0.521146	6.290323
0.3	0.032384	0.026465	0.80411	1.497403	0.549194	1.68341	4.445526
0.5	0.098347	0.121907	2.142171	0.988805	1.522113	1.495336	10.7813
0.7	0.311162	0.140607	5.018788	2.619636	1.757054	0.972416	4.29448
0.9	0.507952	0.374876	3.945707	2.512541	1.104141	0.610319	8.699473

Table I-27. Relative Confidence Interval Half Widths for *Start*=7 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.095055	0.377368	0.958788	1.022979	4.530209
0.3	0	0.059184	0.675073	2.564678	0.651153	1.229123	8.737952
0.5	0.116261	0.083727	3.581175	1.871648	1.440617	1.540999	6.837611
0.7	0.326479	0.470975	5.746315	1.808266	1.112372	0.898288	6.983611
0.9	0.366467	0.324272	2.926462	3.699745	1.97865	1.129488	5.694366

Table I-28. Relative Confidence Interval Half Widths for *Start*=7 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.07764	0.695513	0.761992	1.11358	7.663947
0.3	0.032384	0.052949	0.977956	0.959442	0.736715	1.295145	4.698315
0.5	0.085378	0.275378	1.57422	2.325086	1.441342	1.860597	6.646883
0.7	0.101471	0.331983	2.032696	1.917405	1.447956	1.313924	6.600367
0.9	0.23379	0.429178	0.956914	2.532317	3.01079	1.807307	9.690616

Table I-29. Relative Confidence Interval Half Widths for *Start*=7 and *Max*=8

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.122805	0.826057	1.842725	1.767293	6.320011
0.3	0	0.049523	0.583445	2.21171	1.500712	1.534097	2.675296
0.5	0.107332	0.136302	2.552417	1.341751	1.70117	1.195256	11.23608
0.7	0.138998	0.173926	3.679673	1.515172	2.648795	1.185381	8.543799
0.9	0.54538	0.29063	1.782789	1.346477	1.668091	1.150543	4.092615

Table I-30. Relative Confidence Interval Half Widths for *Start*=7 and *Max*=10

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.095055	0.532963	1.101936	0.669606	5.217937
0.3	0.026438	0.077224	0.812036	0.694524	1.190532	1.389905	7.199308
0.5	0.083857	0.130797	4.664846	2.063063	1.511523	1.260413	6.283578
0.7	0.067134	0.42166	2.233391	0.238795	1.143105	0.693558	12.02051
0.9	0.407097	0.512084	2.851317	2.752961	1.225276	0.749689	4.196076

Relative Confidence Interval Half Widths for *Start*=8

Table I-31. Relative Confidence Interval Half Widths for *Start*=8 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.07764	0.598017	0.610981	0.30462	8.835452
0.3	0.026438	0.077176	0.762324	3.22752	0.613255	0.922703	11.8423
0.5	0.079712	0.123682	1.560669	2.226437	0.468184	1.161578	7.627542
0.7	0.146266	0.2605	1.689823	1.560392	0.904686	1.068085	6.34747
0.9	0.317037	0.299113	1.206505	1.895394	1.264014	0.70489	1.430872

Table I-32. Relative Confidence Interval Half Widths for *Start*=8 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.095228	0.6082	0.634486	0.757675	8.159874
0.3	0.026438	0.032429	0.74276	0.86447	2.060598	1.510517	9.027088
0.5	0.070033	0.184838	1.758518	3.793226	0.759387	1.744347	5.12936
0.7	0.315638	0.121915	2.085086	4.103226	1.35568	1.295524	10.68389
0.9	0.34051	0.557262	1.647754	1.975192	0.81907	1.313778	4.335077

Table I-33. Relative Confidence Interval Half Widths for *Start*=8 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0	1.195507	1.690267	0.93855	5.254616
0.3	0.026438	0.026471	0.736878	1.931737	1.803153	1.302622	9.19991
0.5	0.051485	0.156004	2.678096	0.966909	1.250155	2.106985	11.74999
0.7	0.243032	0.371082	3.574448	2.709132	0.966518	1.146153	10.17312
0.9	0.203748	0.389295	2.929973	1.37449	0.936923	1.893928	10.2012

Table I-34. Relative Confidence Interval Half Widths for *Start*=8 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.122805	0.780719	1.742579	0.690451	8.963523
0.3	0	0.052949	0.829679	1.847358	1.505245	1.499992	7.063551
0.5	0.099739	0.162941	3.031695	2.305061	2.637638	1.735709	6.34735
0.7	0.140953	0.169074	1.623261	0.790163	1.695202	2.181813	8.384039
0.9	0.323689	0.38721	4.304947	1.853451	2.572	1.942674	4.389296

Table I-35. Relative Confidence Interval Half Widths for *Start*=8 and *Max*=8

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.077697	0.811548	1.225267	0.560754	3.676135
0.3	0.032384	0.08785	1.17627	1.459097	2.174028	2.070984	9.768132
0.5	0.065154	0.140937	2.141272	2.156638	0.571699	2.373785	5.803333
0.7	0.286956	0.20801	3.937974	1.475164	0.8698	1.53637	5.654976
0.9	0.384824	0.204747	4.03746	2.20549	1.701241	1.541819	7.065165

Table I-36. Relative Confidence Interval Half Widths for *Start*=8 and *Max*=10

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0	0.703657	0.755007	1.026858	5.641577
0.3	0.026438	0.049542	0.676431	1.549089	1.413032	2.024507	6.077751
0.5	0.10169	0.215301	3.923013	2.159716	1.808873	1.417562	11.36743
0.7	0.109247	0.366684	4.238136	1.683124	3.007187	1.18392	9.014937
0.9	0.355996	0.603655	3.392422	2.068342	1.613066	0.998807	5.169517

Relative Confidence Interval Half Widths for *Start*=9

Table I-37. Relative Confidence Interval Half Widths for *Start*=9 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.190526	0.774319	0.600592	1.294314	6.326896
0.3	0	0.032425	0.820043	2.561739	0.619182	1.075192	5.586845
0.5	0.072839	0.19916	3.224514	3.879043	0.977036	0.862323	4.667375
0.7	0.276704	0.163727	0.493136	1.930227	1.003631	0.987465	6.507994
0.9	0.534194	0.490281	2.06902	3.559415	1.077706	1.186917	5.252716

Table I-38. Relative Confidence Interval Half Widths for *Start*=9 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.123029	1.367843	1.150561	0.952847	7.909308
0.3	0	0.049542	0.785985	2.168101	1.116398	1.485663	9.655843
0.5	0.083011	0.139964	2.527528	2.642445	1.024637	1.372886	10.42506
0.7	0.182295	0.315499	1.81536	3.13846	0.779707	1.677415	6.699678
0.9	0.212443	0.49193	1.829264	2.374574	0.37276	1.809028	12.27516

Table I-39. Relative Confidence Interval Half Widths for *Start*=9 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.095367	1.352109	0.942937	0.994216	7.026001
0.3	0.032384	0.049542	0.39085	1.719967	0.95728	1.532076	2.807113
0.5	0.100975	0.084672	1.998402	1.251157	1.27628	1.635609	14.80992
0.7	0.170555	0.146065	2.403776	3.913808	1.315304	1.09788	7.099057
0.9	0.487868	0.311518	1.148354	2.45603	2.287954	3.099219	7.60655

Table I-40. Relative Confidence Interval Half Widths for *Start*=9 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.190526	1.087111	1.205786	0.7538	3.973471
0.3	0.026438	0.049554	1.054444	2.246143	2.061862	1.796595	10.70828
0.5	0.112166	0.148306	2.474619	1.236181	0.976479	1.573156	12.59777
0.7	0.198873	0.155687	2.353889	1.566008	2.116518	0.8935	11.87189
0.9	0.288221	0.519338	4.209166	2.62094	1.225918	1.601306	8.69362

Table I-41. Relative Confidence Interval Half Widths for *Start*=9 and *Max*=8

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.123029	0.9189	0.57518	0.828563	6.725314
0.3	0.05289	0.089836	0.29635	1.182497	1.244583	0.884133	9.067882
0.5	0.078919	0.07169	1.987277	1.843135	2.303139	1.071701	6.895278
0.7	0.26072	0.106504	1.441098	1.769322	1.561654	1.416185	11.68248
0.9	0.361604	0.476476	2.428352	1.980946	2.816156	1.667236	11.1735

Table I-42. Relative Confidence Interval Half Widths for *Start*=9 and *Max*=10

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.190526	1.418075	1.331905	0.891432	4.798789
0.3	0.026438	0.032425	1.225454	2.233206	2.421439	1.546363	9.146392
0.5	0.039877	0.157081	2.198036	2.153359	2.031869	1.221902	9.56631
0.7	0.289349	0.269023	4.031191	1.488079	1.262945	1.483655	12.22316
0.9	0.129346	0.678548	3.159246	1.654753	2.363145	1.015345	1.785845

Relative Confidence Interval Half Widths for IEEE 802.11

Table I-43. Relative Confidence Interval Half Widths for IEEE 802.11

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	0	0	0.123254	2.178115	N/A	N/A	N/A
0.3	0.032384	0.077176	0.556967	2.582459	N/A	N/A	N/A
0.5	0.168108	0.138551	2.144014	1.878043	N/A	N/A	N/A
0.7	0.2721	0.168814	1.586428	3.191136	N/A	N/A	N/A
0.9	0.603328	0.374826	1.754896	3.895054	N/A	N/A	N/A

Appendix J. Relative Confidence Interval Half Widths for Three Station Simulation Data

This appendix contains all relative confidence interval half width tables for the three station simulation data. An explanation of these tables is found in Chapter 7.

Relative Confidence Interval Half Widths for *Start*=3

Table J-1. Relative Confidence Interval Half Widths for *Start*=3 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.697764	3.440717	2.566414	2.015626	8.558246
0.3	2.464168	2.526369	1.591346	1.433436	0.788029	0.759192	4.928078
0.5	1.206385	1.23169	2.214557	2.226195	1.063381	1.214385	5.323985
0.7	1.487768	1.886852	3.24946	0.702746	0.745596	0.91807	4.179341
0.9	0.671629	1.161558	1.528956	1.577283	1.032761	1.021851	4.036053

Table J-2. Relative Confidence Interval Half Widths for *Start*=3 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.699166	3.311561	3.502149	3.833065	5.519491
0.3	2.487983	2.521477	2.440854	1.111676	0.936342	0.765761	5.050126
0.5	1.206385	1.222327	1.169241	1.227833	0.732029	0.963605	4.412479
0.7	1.128782	1.48701	3.223812	2.407284	0.516171	0.875156	4.220904
0.9	2.03018	2.808823	1.996848	1.665797	0.563252	0.838916	6.281391

Table J-3. Relative Confidence Interval Half Widths for *Start*=3 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.704068	3.476641	3.493928	3.603982	8.153539
0.3	2.487983	2.521477	2.403494	1.486425	0.700345	1.119418	3.060361
0.5	1.206385	1.204396	2.44404	1.638552	0.712903	0.495822	6.011754
0.7	0.91639	1.631153	2.016564	1.196724	0.810052	0.768113	3.856469
0.9	1.888545	4.044533	2.225108	0.981272	1.738578	1.206534	2.983554

Table J-4. Relative Confidence Interval Half Widths for *Start*=3 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.704068	3.473741	3.513612	3.602791	6.14496
0.3	2.487983	2.521477	4.18024	0.694666	1.087888	1.566538	2.762334
0.5	1.206385	1.527941	3.906904	0.659049	0.982178	1.790512	3.605625
0.7	1.247277	2.292481	0.998509	0.807884	0.860905	1.206831	2.862692
0.9	2.688584	3.81855	2.332803	0.627794	0.329966	0.996858	5.666028

Relative Confidence Interval Half Widths for *Start*=4

Table J-5. Relative Confidence Interval Half Widths for *Start*=4 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.470649	3.614398	1.055077	1.299007	9.527344
0.3	2.487983	2.521477	1.610107	2.527368	1.414443	0.86438	2.027986
0.5	1.20702	1.247964	2.716451	2.66897	0.735418	0.660552	2.834578
0.7	1.529834	1.414033	3.150669	1.7973	1.121791	0.658453	6.744481
0.9	0.736211	0.369869	2.044501	2.413999	0.829863	0.562155	3.314251

Table J-6. Relative Confidence Interval Half Widths for *Start*=4 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.600932	3.628429	3.704845	3.571033	9.981518
0.3	2.487983	2.521477	2.197343	1.488043	1.029197	1.219476	4.485287
0.5	1.20702	1.2307	2.725258	0.663799	0.585714	0.918055	3.932
0.7	1.590151	1.36749	1.329687	1.835799	0.572476	0.783229	6.414309
0.9	0.577756	1.103513	3.015576	0.676146	0.651922	1.129514	8.491831

Table J-7. Relative Confidence Interval Half Widths for *Start*=4 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.600932	3.390867	3.825341	3.645816	7.456949
0.3	2.487983	2.521477	1.073253	0.933562	0.568441	0.669093	5.797619
0.5	1.20702	1.225445	1.391948	1.05622	1.638039	1.28089	4.670158
0.7	1.649314	1.596966	2.009294	1.637027	1.393561	1.271949	0.347798
0.9	0.524591	1.601625	2.674956	1.761325	0.779836	1.386819	3.438454

Table J-8. Relative Confidence Interval Half Widths for *Start*=4 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.600932	3.390867	3.825341	3.57902	9.427145
0.3	2.487983	2.521477	3.659284	1.558643	1.056402	1.782563	3.706331
0.5	1.20702	1.220091	2.67411	0.622677	0.438038	1.285116	3.551586
0.7	1.660603	1.927759	3.259884	0.691017	0.816886	0.588892	2.82641
0.9	0.584671	2.283046	2.029388	0.962721	0.771623	0.612886	2.502158

Relative Confidence Interval Half Widths for *Start*=5

Table J-9. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.569706	3.653168	0.652876	1.253411	8.800751
0.3	2.487983	2.547748	2.109108	2.624274	0.536153	0.879519	7.621808
0.5	1.217014	1.221606	2.716401	2.332229	0.779373	0.85294	4.93767
0.7	1.538722	1.744656	2.117069	1.621406	0.569124	1.3071	11.9172
0.9	0.429279	0.375961	2.465807	2.262793	0.461905	0.774984	5.117302

Table J-10. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.655241	3.614398	3.631565	3.857784	4.671107
0.3	2.487983	2.547748	1.003061	2.858967	0.852769	0.904717	9.237085
0.5	1.217014	1.230123	1.757375	0.914113	1.020629	0.544305	8.206465
0.7	1.561312	1.620023	2.175156	1.684691	0.581719	1.105059	8.807105
0.9	0.550997	0.517613	2.423143	1.05621	0.966609	1.222745	6.87985

Table J-11. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.611518	3.352873	3.598596	3.845642	6.641888
0.3	2.487983	2.547748	2.785232	1.438569	1.410955	1.270882	5.425606
0.5	1.217014	1.221606	2.061055	1.17093	1.458631	0.990501	3.845121
0.7	1.561312	1.5129	1.134488	1.527905	0.660705	0.481404	5.211348
0.9	0.636168	0.710854	1.992008	0.814283	0.920412	1.490469	8.383484

Table J-12. Relative Confidence Interval Half Widths for *Start*=5 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.611518	3.352873	3.606196	3.887384	8.288807
0.3	2.487983	2.547748	3.221922	1.474321	0.58691	1.211395	3.253239
0.5	1.217014	1.221606	3.016773	1.595885	0.770423	0.55025	5.118354
0.7	1.561312	1.446363	2.380598	1.026469	1.207892	0.645126	2.753287
0.9	0.690854	0.727663	1.224894	0.753742	1.711011	1.36543	3.307441

Relative Confidence Interval Half Widths for *Start*=6

Table J-13. Relative Confidence Interval Half Widths for *Start*=6 and *Max*=0

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.840194	3.512941	0.555188	0.518163	14.13343
0.3	2.487983	2.547748	2.2228	2.07789	0.673996	0.56574	12.86092
0.5	1.217014	1.222207	1.739961	2.773425	0.818031	1.210503	9.712515
0.7	1.529453	1.734901	1.880532	1.521703	0.62057	0.971945	11.81199
0.9	0.549657	0.587751	1.377376	1.949457	0.676963	0.842974	9.986116

Table J-14. Relative Confidence Interval Half Widths for *Start*=6 and *Max*=2

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.734489	3.593475	3.673584	3.599696	6.648709
0.3	2.487983	2.547748	1.625377	1.282687	0.564739	1.151638	9.238405
0.5	1.217014	1.2307	1.419908	2.470478	1.185647	0.392774	8.546853
0.7	1.529453	1.757398	2.659149	1.708164	0.696852	1.451503	10.11747
0.9	0.549657	0.490142	1.789149	1.235156	0.573003	1.562472	4.53018

Table J-15. Relative Confidence Interval Half Widths for *Start*=6 and *Max*=4

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.738962	3.519678	3.570399	3.939758	4.805597
0.3	2.487983	2.547748	3.084966	0.805981	1.022691	1.186841	6.855279
0.5	1.217014	1.222892	3.373297	2.11181	1.285407	0.693865	9.144175
0.7	1.529453	1.718163	3.276602	1.738064	1.408921	0.94563	6.336754
0.9	0.549657	0.438541	2.825668	0.491387	0.936791	0.639024	4.896175

Table J-16. Relative Confidence Interval Half Widths for *Start*=6 and *Max*=6

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.738962	3.518734	3.615782	3.886851	8.077753
0.3	2.487983	2.547748	2.472046	2.045066	0.795719	1.165092	5.655875
0.5	1.217014	1.222892	2.137201	2.41963	1.279667	0.84991	5.849488
0.7	1.529453	1.718163	2.93228	1.87511	1.028385	0.929848	4.506424
0.9	0.549657	0.611888	3.328012	0.759387	1.379919	1.259738	4.407213

Relative Confidence Interval Half Widths for IEEE 802.11

Table J-17. Relative Confidence Interval Half Widths for IEEE 802.11

	BER						
Load	0.00001	0.00002	0.0001	0.0002	0.001	0.002	0.01
0.1	3.601693	3.601693	3.738962	3.249395	N/A	N/A	N/A
0.3	2.487983	2.547748	2.395904	1.841625	N/A	N/A	N/A
0.5	1.217014	1.222892	1.630289	2.692736	N/A	N/A	N/A
0.7	1.529453	1.696032	2.07271	3.677631	N/A	N/A	N/A
0.9	0.549657	0.602941	1.56225	2.793269	N/A	N/A	N/A

Vita

Barry E. Mullins

June 1997

Education

August 1994 - June 1997	*Virginia Polytechnic Institute and State University*, Blacksburg, VA, Ph.D. Electrical Engineering.
June 1986 - December 1987	*Air Force Institute of Technology*, Wright-Patterson AFB, OH, M.S. Computer Engineering. Thesis: "The Integration of Artificial Intelligence to Improve the Effectiveness of Electronic Countermeasure Strategies in a Tactical Environment."
August 1979 - May 1983	*University of Evansville*, Evansville, IN, B.S. Computer Engineering, Cum Laude.
August 1989	Squadron Officer School

Work Experience

June 1991 - July 1994	Assistant Professor, Department of Electrical Engineering, United States Air Force Academy, Colorado Springs, Colorado.
January 1988 - May 1991	Deputy Chief of the Computer Technology Section of Wright Laboratory, Eglin Air Force Base, Fort Walton Beach, Florida.
September 1983 - May 1986	Missile Guidance and Control Engineer for the Air-Launched Antisatellite (ASAT) system. 6595 Aerospace Test Group, Vandenberg Air Force Base, Lompoc, California.

Honors/Awards/Affiliations

Brig. Gen R.E. Thomas Award for outstanding contribution to cadet education, United States Air Force Academy, Colorado Springs, CO. May 1994
Brig. Gen R.E. Thomas Award for outstanding contribution to cadet education, United States Air Force Academy, Colorado Springs, CO. May 1993
Electrical Engineering Company Grade Officer of the Quarter. June 1993
Electrical Engineering Company Grade Officer of the Quarter. September 1992
United States Air Force Meritorious Service Medal
United States Air Force Commendation Medal (one oak leaf cluster)
United States Air Force Achievement Medal
Parachutist Wings
USAF Elementary-Level Excellence-In-Competition Bronze Pistol Badge
Member of IEEE Computer Society and Communications Society
Member of the following honor societies: Eta Kappa Nu (Electrical Engineering)
Phi Beta Chi (Science)
Kappa Mu Epsilon (Math)

Publications

B.E. Mullins, N.J. Davis IV, and S.F. Midkiff, "An Adaptive Wireless Local Area Network Protocol That Improves Throughput Via Adaptive Control of Direct Sequence Spread Spectrum Parameters," To appear in *ACM Mobile Computing and Communication Review*, Vol. 1, No. 3, 1997.

B.E. Mullins, N.J. Davis IV, and S.F. Midkiff, "A Wireless Local Area Network Protocol That Improves Throughput Via Adaptive Control," *Proceedings of the 1997 IEEE International Conference on Communications*, pp. 1427-1431, June 1997, Montreal, Quebec, Canada.

B.E. Mullins, "Common Ada Missile Packages (CAMP)," *Proceedings of the Guidance and Control Panel 52nd Symposium on Software for Guidance and Control*, 7-10 May 1991, Thessaloniki, Greece, Sponsored by the Advisory Group for Aerospace Research & Development (AGARD).

B.E. Mullins, "The Expert Missile Maintenance Aid Program - Four Years Later and in the Field," *Proceedings of the Space Operations, Applications and Research Symposium (SOAR '90)*, 26-28 June 1990, Albuquerque, New Mexico, Sponsored by the National Aeronautics and Space Administration, Johnson Space Center, the U.S. Air Force, Space Technology Center, and the University of New Mexico.

B.E. Mullins, "Diagnosing Munition Faults: The EMMA Approach," *Proceedings of the 27th Aerospace Sciences Meeting*, 9-12 January 1989, Reno, Nevada, Sponsored by the American Institute of Aeronautics and Astronautics (AIAA), Washington, DC., AIAA 89-0380, DTIC number: AD-A202442.

B.E. Mullins, "EMMA: The Expert System For Munition Maintenance," *Proceedings of the Second Annual Workshop on Space Operations Automation and Robotics (SOAR '88)*, Wright State University, Dayton, OH, 20-23 July 1988, Sponsored by the National Aeronautics and Space Administration, Washington DC., and the U.S. Air Force, Washington DC., NASA Conference Publication 3019, Pages 31-39, DTIC Number AD-A202558.