Dynamic Code Sharing Algorithms for IP Quality of Service in Wideband CDMA 3G Wireless Networks

Carl E. Fossa Jr.

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

Nathaniel J. Davis IV, Chairman

Charles W. Bostian

Scott F. Midkiff

Timothy Pratt

Srinidhi Varadarajan

April 25, 2002

Blacksburg, Virginia

Keywords: OVSF Code Assignment, 3G, WCDMA, IP QoS, wireless communications, network performance

Dynamic Code Sharing Algorithms for IP Quality of Service in Wideband CDMA 3G Wireless Networks

Carl E. Fossa Jr.

(ABSTRACT)

This research investigated the efficient utilization of wireless bandwidth in Code Division Multiple Access (CDMA) systems that support multiple data rates with Orthogonal Variable Spreading Factor (OVSF) codes. The specific problem being addressed was that currently proposed *public-domain* algorithms for assigning OVSF codes make inefficient use of wireless bandwidth for bursty data traffic sources with different Quality of Service (QoS) requirements. The purpose of this research was to develop an algorithm for the assignment of OVSF spreading codes in a Third-Generation (3G) Wideband CDMA (WCDMA) system. The goal of this algorithm was to efficiently utilize limited, wireless resources for bursty data traffic sources with different QoS requirements.

The key contribution of this research was the implementation and testing of two code sharing techniques which are not implemented in existing OVSF code assignment algorithms. These techniques were termed *statistical multiplexing* and *dynamic code sharing*. The statistical multiplexing technique used a shared channel to support multiple bursty traffic sources. The dynamic code sharing technique supported multiple data users by temporarily granting access to dedicated channels. These techniques differed in terms of both complexity and performance guarantees.

Dedication

This work is dedicated to my wife Susan, and to my daughters Christina and Lisa. You each made sacrifices over the past three years that enabled me to complete this work. Your understanding, support, and love helped me make it through the long hours and still keep things in perspective. Thanks! I could not have done it without you.

Acknowledgments

I would like to acknowledge the many people without whom I could not have completed this dissertation. First, I would like to thank my advisor Nathaniel Davis. His guidance and insight throughout this research, as well as his editing during the preparation of this document, were invaluable. Next, I would like to thank my committee members Charles Bostian, Scott Midkiff, Timothy Pratt, and Srinidhi Varadarajan for their support of this research. I would also like to thank Luiz DaSilva for his insight to self-similar traffic sources, and Amitabh Mishra for his help with QoS in 3G wireless networks. Finally, I would like to thank Roberto Conte, Anwer Khan, Tim Gallagher, Eric Shea and Max Robert for their friendship and advice.

Contents

List of Figures

List of Tables

Glossary of Acronyms

$1G$ First-Generation

$2G$ Second-Generation

$3G$ Third-Generation

$3GPP$ Third-Generation Partnership Proposal

$4G$ Fourth-Generation

ABR Available Bit Rate

$AMPS$ Advanced Mobile Phone System

$ARIB$ Association of Radio Industry and Business

ATM Asynchronous Transfer Mode

$AWGN$ Additive White Gaussian Noise

$BCCH$ Broadcast Control Channel

BCH Broadcast Channel

BER Bit Error Rate

BMC Broadcast Multicast Control

CAC Call Admission Control

CBR Constant Bit Rate

$CCCH$ Common Control Channel

$CDMA$ Code Division Multiple Access

$CDPD$ Cellular Digital Packet Data

CDV Cell Delay Variation

CL Controlled Load

CLR Cell Loss Ratio

$CPCH$ Common Packet Channel

CQS Classify Queue and Schedule

$CTCH$ Common Traffic Channel

CTD Cell Transfer Delay

$DCCH$ Dedicated Control Channel

DCH Dedicated Channel

DCS Digital Cellular System

$DiffServ$ Differentiated Services

$DSCH$ Downlink Shared Channel

$DSSS$ Direct Sequence Spread Spectrum

$DTCH$ Dedicated Traffic Channel

$EDGE$ Enhanced Data rates for GSM Evolution

$ETSI$ European Telecommunications Standards Institute

$FACH$ Forward Access Channel

$FAUSCH$ Fast Uplink Signalling Channel

FCC Federal Communications Commission

$FDMA$ Frequency Division Multiple Access

FEC Forward Error Correction

$FHSS$ Frequency Hopping Spread Spectrum

$FIFO$ First In First Out

$GGSN$ Gateway GPRS Support Node

$GMSK$ Gaussian Minimum Shift Keying

$GPRS$ General Packet Radio Service

GPS Global Positioning System

GS Guaranteed Service

GSM Global System for Mobile Communications

$IETF$ Internet Engineering Task Force

$IMT-2000$ International Mobile Telecommunications in the year 2000

$IntServ$ Integrated Services

IP Internet Protocol

ISP Internet Service Provider

ITU International Telecommunication Union

LAC Link Access Control

LAN Local Area Network

$LFSR$ Linear Feedback Shift Register

LPI Low Probability of Intercept

$M - Sequence$ Maximal-Length Sequence

MAC Medium Access Control

MBS Maximum Burst Size

MCR Minimum Cell Rate

MSC Mobile Switching Center

$NAMPS$ Narrowband Advanced Mobile Phone System

$NodeB$ Node Base station

$nrt - VBR$ Non-real-time Variable Bit Rate

$OVSF$ Orthogonal Variable Spreading Factor

$PCCH$ Paging Control Channel

$PCDP$ Packet Convergence Data Protocol

PCH Paging Channel

PCR Peak Cell Rate

PCS Personal Communications Services

PHY Physical Layer

PN Pseudorandom Noise

$PSTN$ Public Switched Telephone Network

PVC Permanent Virtual Circuit

QoS Quality of Service

$QPSK$ Quadrature Phase Shift Keying

$RACH$ Random Access Channel

RFC Request for Comments

RLC Radio Link Control

RNC Radio Network Controller

RRC Radio Resource Control

$RSVP$ Resource Reservation Protocol

$rt-VBR$ Real-Time Variable Bit Rate

SCR Sustainable Cell Rate

SF Spreading Factor

$SGSN$ Serving GPRS Support Node

SLA Service Level Agreement

SNR Signal to Noise Ratio

$SSMA$ Spread Spectrum Multiple Access

SVC Switched Virtual Circuit

TCP Transmission Control Protocol

$TDMA$ Time Division Multiple Access

TF Transport Format

TIA Telecommunications Industry Association

TOS Type Of Service

TTA Telecommunications Technology Association

UBR Unspecified Bit Rate

UDP User Datagram Protocol

UE User Equipment

$USCH$ Uplink Shared Channel

$USDC$ United States Digital Cellular

$UTRAN$ UMTS Terrestrial Radio Access Network

VBR Variable Bit Rate

VC Virtual Circuit

VCI Virtual Channel Identifier

$VOIP$ Voice Over IP

VPI Virtual Path Identifier

WAN Wide Area Network

$WARC$ World Administrative Radio Conference

$WCDMA$ Wideband Code Division Multiple Access

Chapter 1

Introduction

The telecommunications industry underwent significant advances in data communications and wireless communications during the 1990s. This is exemplified by the rapid growth of both the Internet and the cellular telephone network. At the beginning of the 1990s, Internet access was limited primarily to academic and scientific institutions. Likewise, high costs and limited coverage areas restricted the widespread use of cellular telephones. Today, inexpensive Internet access and cellular telephone service are available for both business and home users.

As we enter the 21st century, attempts to integrate voice and data traffic are revealing the limitations of both the Internet and the cellular network. The Internet's best-effort traffic model is not suitable for real-time voice and video traffic. Similarly, the cellular telephone network's low data rate is not sufficient for web browsing or large file transfers. Ongoing research is aimed at improving Quality of Service (QoS) for the Internet, and increasing data rates on cellular networks. This research effort focuses on supporting data traffic, with different QoS requirements, in the emerging Third Generation (3G) cellular networks.

This chapter defines the problem investigated in this research effort. The remainder of the chapter is organized as follows. Section 1.1 states the research problem under inves-

tigation. A brief background of the problem is presented in Section 1.2, and the motivation for this research is given in Section 1.3. Section 1.4 lists the specific questions addressed by this research effort, and Section 1.5 describes the intended purpose of this research. A brief overview of the methodology used is presented in Section 1.6. Finally, Section 1.7 gives a summary of the significant results.

1.1 Problem Statement

This research investigated the efficient utilization of wireless bandwidth in Code Division Multiple Access (CDMA) systems that support multiple data rates with Orthogonal Variable Spreading Factor (OVSF) codes. The specific problem being addressed is that currently proposed *public-domain* algorithms for assigning OVSF codes make inefficient use of wireless bandwidth for bursty data traffic sources with different Quality of Service (QoS) requirements. Existing algorithms address problems such as initial code assignment, code blocking, and supporting both real-time and best effort traffic sources [1][2][3]. However, all of these algorithms use dedicated spreading code assignment and limit the efficient use wireless bandwidth in several ways:

- They do not support "over-booking" of Variable Bit Rate (VBR) traffic with uncorrelated peak data rates.
- They do not support the statistical multiplexing of bursty data traffic without real-time constraints.
- They do not allow Unspecified Bit Rate (UBR) or best-effort traffic to take advantage of unused bandwidth when VBR traffic is not transmitting at its peak data rate.

This research focused on developing an improved OVSF code assignment algorithm, which increases channel utilization by allowing bursty data users to share spreading codes.

1.2 Background

Orthogonal spreading codes are typically used to separate users on the forward, or downlink, of CDMA cellular communications systems. The good cross correlation properties of orthogonal codes allow the maximum number of simultaneous users. A set of orthogonal codes typically contains N codes of length L, where $N = L$. For example, the commercial IS-95 CDMA standard utilizes 64 Walsh codes that are 64 bits long to separate users on the forward link. The use of equal length spreading codes results in the same data rate for each user. Higher data rates can be achieved, at a cost of receiver complexity, by assigning multiple codes to a user.

Adachi, Sawahashi, and Okawa proposed a method of generating orthogonal spreading codes with different lengths, termed Orthogonal Variable Spreading Factor codes [4]. OVSF codes provide a means of bandwidth allocation in a CDMA system. Each user is assigned a single spreading code, with a spreading factor corresponding to the required data rate. The properties of OVSF codes limit which codes can be simultaneously assigned. As a result, the assignment of OVSF codes has a significant impact on efficient resource utilization. The Third Generation Partnership Proposal (3GPP) has proposed the use of OVSF codes in 3G Wideband CDMA (WCDMA) systems to support data traffic with different data rate requirements[5].

1.3 Motivation

This research is motivated by the lack of public domain algorithms for the assignment of OVSF codes. One factor contributing to the lack of published algorithms is the relative newness of OVSF codes. Most, if not all, existing CDMA systems use spreading codes of the same length. A second, probably more significant factor is the importance of such algorithms to the performance of 3G WCDMA systems. The 3GPP technical specifications

require the use of OVSF channelization codes in the WCDMA forward link, but leave the implementation details up to the equipment vendors [6]. The primary focus of 3G wireless systems is to provide enhanced data services to mobile users. The ability of code assignment algorithms to efficiently support data traffic will differentiate 3G equipment from different wireless equipment vendors. While it is very likely that equipment vendors are researching OVSF code assignment, it is equally likely that commercial interests will prevent the timely publication of their results.

1.4 Research Questions

This research addressed the following questions regarding data traffic performance on the forward link of a WCDMA system. In each case, the performance metrics were channel utilization, and end-to-end delay.

1. What is the baseline performance that can be expected for a single bursty data source on a single, dedicated OVSF code?
2. How does the performance of bursty data traffic compare to the baseline performance when multiple bursty sources are statistically multiplexed on a single OVSF code?
   (a) At what traffic load does the delay become unacceptable?
   (b) Is the statistical multiplexing of bursty data sources suitable for non-real-time data traffic?
   (c) Is the statistical multiplexing of bursty data sources suitable for real-time data traffic?
3. How does the performance of bursty data traffic compare to the baseline performance when multiple bursty sources are "over booked" by dynamically sharing OVSF codes?

(a) At what traffic load does the delay become unacceptable?

(b) Is dynamic code sharing between bursty data sources suitable for non-real-time data traffic?

(c) Is dynamic code sharing between bursty data sources suitable for real-time data traffic?

4. Should a Radio Link Control (RLC) or Medium Access Control (MAC) algorithm use statistical multiplexing, dynamic code sharing, or a combination of both methods to assign OVSF codes for bursty data traffic with different QoS requirements?

5. How does the use of exponential, on-off, and self-similar traffic models affect the performance of a code assignment algorithm using statistical multiplexing or dynamic code sharing?

1.5 Purpose

The purpose of this research is to develop an algorithm for the assignment of OVSF spreading codes in a 3G WCDMA system. This algorithm will support the dynamic sharing of spreading codes. The goal of this algorithm is to efficiently utilize limited, wireless resources for bursty data traffic sources with different QoS requirements. The algorithm should also fit within the framework of the WCDMA RLC/MAC layer without modifying the 3GPP protocol specifications.

1.6 Methodology Overview

A simulation model was developed, using the ten step methodology presented in [7], to test the code sharing algorithm. The simulation was designed in OPNET Modeler 7.0 using a top-down approach. It implements two different code sharing techniques, which are

not used in currently proposed OVSF code assignment algorithms. These techniques are termed *statistical multiplexing* and *dynamic code sharing*. Performance of the algorithms was measured in terms of channel utilization, and end-to-end delay. Simulation parameters were selected to accurately model the WCDMA forward link. Factors that were varied in the simulation include the code sharing algorithm, traffic loading levels, and traffic distributions. The simulation was run using deterministic, exponential, on-off, and self-similar traffic distributions. Analysis was conducted to verify the simulation results using both deterministic and exponential traffic sources.

1.7 Significant Results

The results of this research indicate that both statistical multiplexing and dynamic code sharing offer improved channel utilization for bursty data traffic. The baseline channel utilization for a single exponential traffic source was 83 percent, and that for a single on-off traffic source was 50 percent. The largest improvement in channel utilization for statistical multiplexing was for the case of self-similar traffic generated by multiplexing ten on-off traffic sources. This scenario resulted in a 26 percent improvement in channel utilization with an average end-to-end delay below 10 ms. The dynamic code sharing technique made efficient use of unused channel bandwidth by transmitting medium and low QoS traffic when the high QoS traffic was not transmitting at the peak data rate. The improvement in channel utilization varied depending upon the data rate of the high QoS traffic. Exponential traffic sources gave improvements in channel utilization between 0 and 275 percent with high QoS end-to-end delay below 32 ms. Likewise, on-off traffic sources results showed improvements in channel utilization between 20 and 350 percent with high QoS end-to-end delay below 200 ms. Finally, dynamic code sharing for self-similar traffic produced improvements in channel utilization between 50 and 65 percent with high QoS end-to-end delay below 5 ms. The algorithm developed in this research effort offers significant improvement in channel utiliza-

tion for bursty traffic sources with different QoS requirements, without adversely affecting the throughput or delay of high QoS sources.

1.8 Summary

This research effort focused on supporting data traffic with different QoS requirements in the emerging 3G cellular networks. The specific problem being addressed is that currently proposed OVSF code assignment algorithms inefficiently utilize bandwidth for bursty traffic sources. The research questions presented in Section 1.4 are answered by evaluating two methods of sharing wireless bandwidth. These methods are statistically multiplexing bursty data sources on a single OVSF code, and dynamically sharing a group of OVSF codes between bursty data sources. The purpose of this research was to develop an algorithm for the assignment of OVSF spreading codes that makes efficient use of wireless resources and supports data traffic with different QoS requirements. The algorithm was designed to fit within the 3GPP WCDMA protocol specifications. OPNET Modeler 7.0, a commercial network simulation package, was utilized to develop and test the algorithm. The algorithm was verified analytically using both deterministic and exponential traffic sources.

This chapter has defined the research problem that is addressed in the remainder of the document. The rest of the document is organized as follows. Chapter 2 presents both the theoretical background, and a review of relevant literature for the problem of OVSF code assignment. Chapter 3 describes the methodology that was used to develop and test an OVSF code assignment algorithm and gives an overview of the simulation design. Chapter 4 presents analysis using both deterministic and exponential traffic sources to verify correct operation of the simulation model. The results from the simulation are presented in Chapter 5, demonstrating the improvement in channel utilization offered by sharing OVSF codes. Finally, Chapter 6 gives conclusions and recommendations for future work.

Chapter 2

Overview of Third Generation Wireless Networks and Code Assignment Algorithms

This chapter provides both the theoretical background, and a review of related research in the area of OVSF code assignment in Third Generation (3G) wireless networks. A basic theoretical background in a number of voice and data communications areas is required to address the topic of this research effort. This is exemplified by the variety of topics presented in this chapter. Section 2.1 provides an introduction to spread spectrum communications and concludes with a discussion of OVSF codes. Section 2.2 provides an introduction to cellular communications, which includes a detailed description of the WCDMA 3G wireless standards. Section 2.3 presents an introduction to Quality of Service (QoS) that includes both ATM and IP data networks. Section 2.4 gives an introduction to self-similar traffic and discusses the challenges associated with using self-similar traffic models in simulation. Finally, Section 2.5 presents an overview of related research in the assignment of OVSF spreading codes.

2.1 Spread Spectrum Communications

Spread spectrum communications generally refers to a modulation scheme that produces a transmission bandwidth much larger than the information bandwidth *independently* of the information signal bandwidth. The concept of spread spectrum communications has its roots in World War II military research efforts aimed at developing a means of secure communications. Most of this research remained classified until the 1970's, and the first *IEEE Transactions in Communications* special issue on spread spectrum communications appeared in 1977 [8][9]. Spread spectrum systems have evolved from strictly military applications to commercial navigation and communication systems. Two prominent examples of this are the Global Positioning System (GPS) navigation system, and the IS-95 Code Division Multiple Access (CDMA) cellular telephone system. This section provides the fundamental background information on spread spectrum communications necessary to understand the principles of Wideband CDMA (WCDMA) and Orthogonal Variable Spreading Factor (OVSF) codes.

2.1.1 Basic Principles of Spread Spectrum

The basic concept of spread spectrum communications is to transmit a signal using a bandwidth that is much wider than the signal bandwidth. Several modulation and coding schemes, such as wideband frequency modulation and Forward Error Correction (FEC) coding, result in increased transmission bandwidth but are not considered spread spectrum. The authors of [8] present the following criteria as necessary for a system to be classified as spread spectrum:

1. The transmitted signal energy must occupy a bandwidth which is much larger than the information bandwidth and which is approximately independent of the information bandwidth.

2. Demodulation must be accomplished, in part, by correlation of the received signal with a replica of the signal used in the transmitter to spread the information signal.

The purpose of transmitting a signal with a bandwidth wider than required is to improve the communications system performance. Spread spectrum systems have the following properties, which are utilized to improve performance relative to narrow band communications systems.

1. *Multiple access.* A spread spectrum system allows multiple users to transmit simultaneously in the same transmission bandwidth. This is accomplished by assigning each user a unique code from a set of codes with low cross-correlation properties. Codes with low-cross correlation properties are also termed orthogonal spreading codes. Demodulation by correlating the received signals with the desired code will recover the desired signal. The undesired signals will remain spread over the transmission bandwidth and will contribute noise power to the desired signal-to-noise ratio ($\frac{S}{N}$). This is illustrated in Figure 2.1. Unlike Time Division Multiple Access (TDMA) or Frequency Division Multiple Assess (FDMA) schemes, Spread Spectrum Multiple Access (SSMA) systems do not have a hard limit on the maximum number of users. Increasing the number of users both decreases the $\frac{S}{N}$ ratio for existing users, and increases the probability of bit error (P_b) for all users.

2. *Multipath interference reduction.* A spread spectrum system is capable of reducing multipath interference. Multipath interference occurs when several copies of a transmitted signal arrive at a receiver due to reflection and refraction. The multiple signals arrive with different distortions in amplitude, phase, and time delay, which causes them to combine both constructively and destructively. This interference produces frequency selective fading of the received signal. Multipath interference is reduced by correlating the received signals with the spreading code to recover the desired signal.

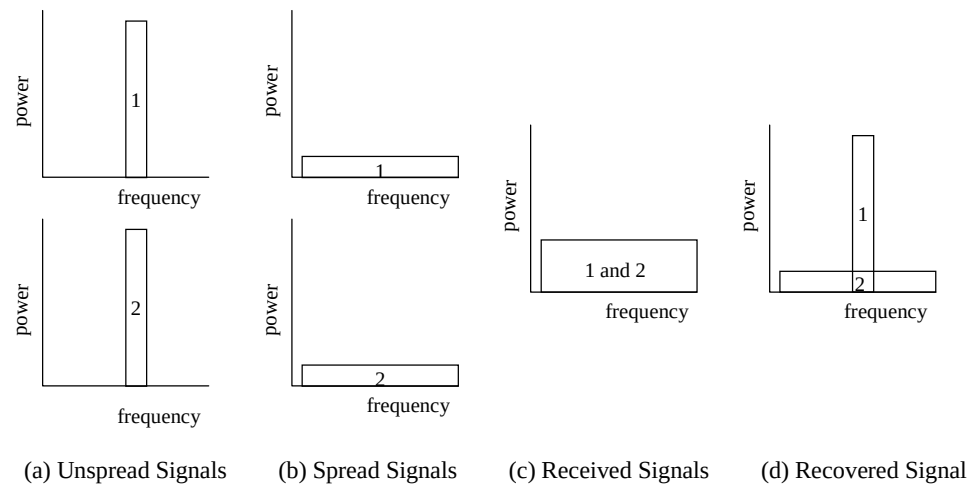

Figure 2.1: Spread Spectrum Multiple Access

Multipath signals with large distortions in amplitude, phase or time delay will appear uncorrelated with the spreading code and will remain spread.

3. *Privacy.* A spread spectrum system offers some degree of privacy because the spreading code is required to recover the transmitted signal.

4. *Anti-jamming.* A spread spectrum system is capable of reducing the effects of narrowband jamming. This is possible because correlating the jamming signal with the spreading code in effect spreads the jamming signal while despreading the desired signal.

5. *Low Probability of Intercept.* Spread spectrum signals are difficult to detect, a characteristic termed Low Probability of Intercept (LPI), because the signal power is dispersed over a wide bandwidth. The transmitted signal is then difficult to distinguish from noise.

The performance improvement offered by a spread spectrum system is termed processing gain (G_p). The ratio of transmitted bandwidth (B_t) to information bandwidth (B_i) is often used to approximate the performance gain of a spread spectrum system.

$$G_p = \frac{B_t}{B_i} \tag{2.1}$$

2.1.2 Types of Spread Spectrum Multiple Access

There are several modulation techniques used to generate spread spectrum signals. A Pseudorandom Noise (PN) sequence is typically used as a spreading code to convert a narrowband signal into a wideband, noise-like signal. In *Frequency Hopping Spread Spectrum* (FHSS), the PN sequence is input to a frequency synthesizer to change the carrier frequency of the transmitted signal periodically. An example of a FHSS system is the IEEE 802.11 wireless LAN. In *Direct Sequence Spread Spectrum* (DSSS)

Frequency Hopping Spread Spectrum

In FHSS, the carrier frequency of the modulated signal changes periodically. The available frequency band is divided into a number of narrowband channels. The spreading code is a PN sequence that selects one of these narrowband channels in a pseudorandom manner. The block diagram of a FHSS transmitter is shown in Figure 2.2. The set of carrier frequencies available to a particular signal is termed a *hop set*. Frequency hopping systems are further classified based upon how quickly the carrier frequency changes with respect to the symbol duration of the modulated signal. In a *fast-hopping* system the carrier frequency changes within the symbol period. In a *slow-hopping* system an entire symbol is transmitted before the carrier frequency changes.

Assigning each user a different hop set gives FHSS the properties discussed in Section 2.1.1. The group of hop sets, or codes, must have low cross correlation properties. This equates to a small probability that users with different hop sets will simultaneously transmit on the same carrier frequency. Multiple access is possible because several users can simultaneously transmit on different carrier frequencies. The number of simultaneous

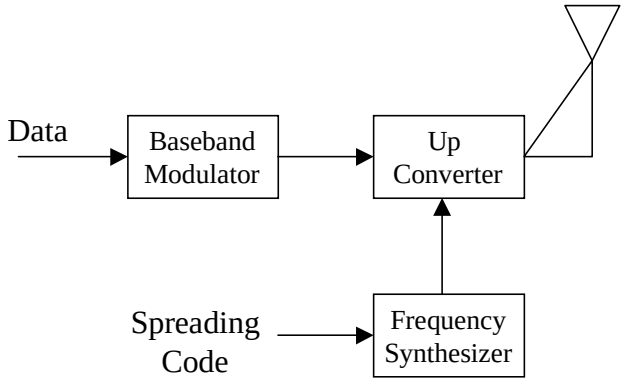

Figure 2.2: Block Diagram of FHSS Transmitter

users is interference limited since adding users increases the probability of simultaneous transmissions on the same carrier frequency. Multipath interference reduction is possible in fast-hopping FHSS since each symbol is transmitted on more than one frequency. The multipath interference affects each carrier frequency differently, and averaging the received carrier frequencies reduces multipath interference. Privacy is provided because a receiver needs to know the PN hopping sequence to demodulate the signal. FHSS reduces the effects of narrowband jamming since the signal is only interfered with when it hops to the carrier frequency being jammed. LPI is provided to some degree by rapidly changing the carrier frequency, but the transmitted signal is detectable above the noise floor.

Direct Sequence Spread Spectrum

In DSSS the data signal is modulated by a PN code sequence that effectively spreads the signal power over a wide bandwidth. The block diagram of a DSSS transmitter is shown in Figure 2.3. The data signal may be analog or digital. The PN *spreading code* is a digital signal that takes on values of $+1$and -1, and the number of code bits per second is commonly called the *chip rate* (R_c). The chip rate is typically much larger than the data symbol rate (R_s), which results in the desired spreading in the frequency domain. The *Spreading Factor* (SF) of a DSSS system is the ratio of R_c to R_s.

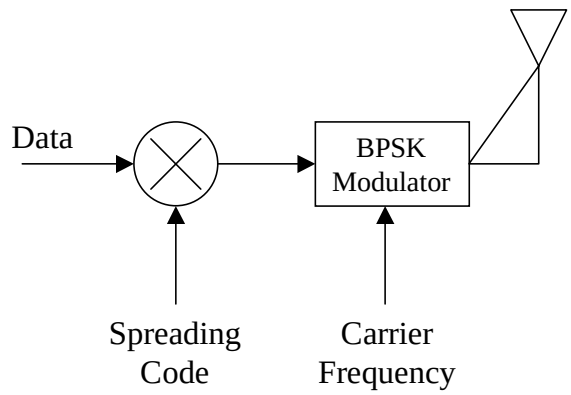

Figure 2.3: Block Diagram of DSSS Transmitter

$$SF = \frac{R_c}{R_s} \tag{2.2}$$

Assigning each user a different spreading code gives DSSS the properties discussed in Section 2.1.1. Multiple access is accomplished by assigning each user a unique spreading code from a set of codes with low cross correlation properties. The receiver recovers a desired signal from a group of spread signals by correlating with the correct spreading code. This demodulates the desired signal but not the signals of other users. The capacity of a DSSS system is interference limited. The spread signal of each user has properties similar to Additive White Gaussian Noise (AWGN). Increasing the number of users effectively decreases the carrier to noise ratio ($\frac{C}{N}$), and increases the probability of bit error (P_b), for all users. Multipath interference rejection is possible if the spreading codes have good autocorrelation properties. An ideal autocorrelation function has a value of zero outside the interval $[-T_c, T_c]$ where T_c is the chip duration. A multipath signal that is delayed by greater than $2T_c$, much like a signal modulated with an uncorrelated spreading code, is not recovered by correlation with the desired spreading code. DSSS provides privacy because the receiver must know the spreading code to demodulate the signal. DSSS reduces the effects of narrowband jamming because correlating with the spreading codes despreads the desired signal and spreads the jamming signal. Thus only a small amount of the jamming power remains in the signal

bandwidth. LPI is provided since the spread signal has properties similar to AWGN and can be hidden in the background noise.

2.1.3 Direct Sequence Spreading Codes

CDMA cellular networks typically use DS spreading codes to support multiple users transmitting on the same carrier frequency. Spreading codes separate users within the same cell, as well as users in different cells. An ideal spreading code would have good autocorrelation properties, low cross-correlation properties, and a large number of available codes. Good autocorrelation properties are desired both to recover the desired signal, and to reduce the effects of multipath signals. Similarly, low cross-correlation properties are important to minimize the effects of interfering signals. A large number of available codes is desirable since this determines the number of simultaneous users in a DSSS system. DS spreading codes can be classified as either *PN sequences* or *orthogonal codes* based upon how they are generated[10]. These "non-ideal" spreading codes can not have both good autocorrelation, and low cross-correlation properties [10]. In addition, the maximum number of available codes is a function of the code length.

Pseudonoise Sequences

PN sequences are deterministic sequences that exhibit randomness properties similar to sampled white Gaussian noise. Since PN sequences are deterministic, they can be generated or stored at the receiver to despread incoming signals. PN sequences are typically generated using a Linear Feedback Shift Register (LFSR), as illustrated in Figure 2.4. The sequence generated by a LFSR is periodic with a maximum period of $N = 2^n - 1$, where n is the number of stages in the shift register. A LFSR sequence with the maximum possible period is called a *Maximal-Length Sequence* (M-sequence). A *generator polynomial* is typically used to describe a LFSR, with each non-zero term corresponding to a feedback connection.

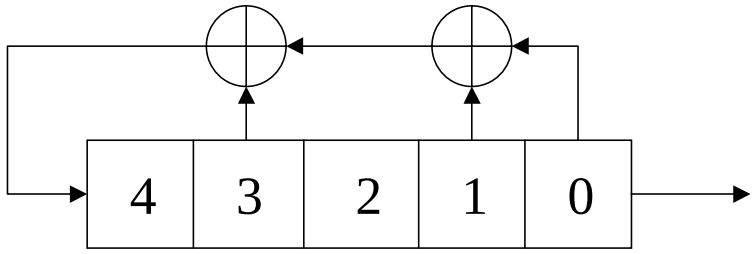

Figure 2.4: Linear Feedback Shift Register

Table 2.1: Maximal-Length Sequences

Shift Registers	Code Length	Number of Codes
3	7	2
4	15	2
5	31	6
6	63	6
7	127	18
8	255	16
9	511	48

The generator polynomials that produce M-sequences for LFSRs with up to thirty-four shift registers are tabulated in [8]. Table 2.1 gives the M-sequence code length, and the number of unique M-sequences, for shift registers of various lengths.

M sequences exhibit the following properties, which make them appear random[11][12][8].

1. *Balance Property.* In each period of the sequence, the number of binary ones differs from the number of binary zeros by at most one.

2. *Run Property.* In each period of the sequence, about one-half of the runs are of length one, one-fourth of the runs are of length two, one-eighth are of length three, and so on. A run is defined as a sequence of all binary ones or all binary zeros, and the length is defined as the number of digits in the run.

3. *Autocorrelation Property.* If a period of the sequence is compared term by term with any cyclic shift of itself, the number of digits that agree and the number of digits that disagree will differ by at most one.

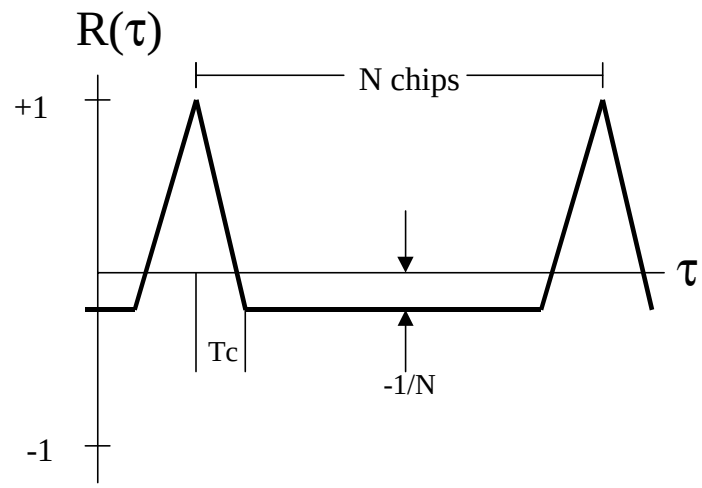

Figure 2.5: PN Autocorrelation Function

The autocorrelation properties of PN sequences are very good for both detection of the desired signal, and rejection of multipath signals. The PN autocorrelation function (Figure 2.5) has a maximum value of one when the codes are perfectly synchronized, and has a value of $-\frac{1}{N}$ when the time delay (τ) is greater than the chip time (T_c). This property of the PN autocorrelation function is useful in overcoming the relatively weak cross-correlation properties of PN sequences. For PN sequences with long periods, the autocorrelation between the original code and a time shifted version of the same code is approximately zero. Therefore, time shifted versions of a single PN sequence can be effectively utilized as different DS spreading codes. The drawback of this approach is that it requires synchronization between users in the system. The IS-95 CDMA cellular network is an example of a synchronous system that uses time shifted PN sequences as different spreading codes. An alternative approach is to linearly combine carefully selected M-sequences, which results in a sequence with both good autocorrelation properties and bounded cross-correlation properties. Gold Codes and Kasami Codes are both generated by combining the M-sequences from different LFSRs [12].

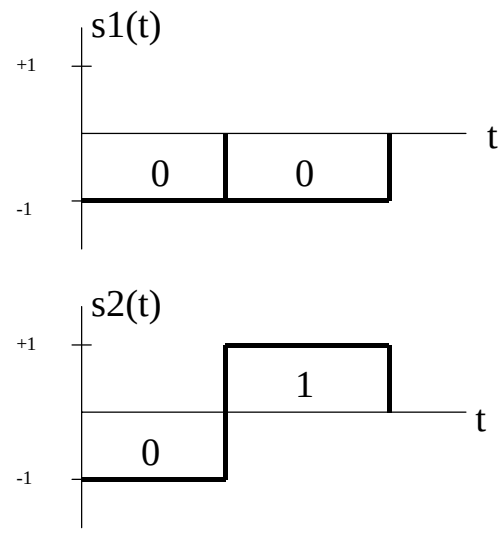

Figure 2.6: Orthogonal Signals

Orthogonal Codes

Orthogonal codes are sets of binary sequences that have a cross-correlation coefficient equal to zero. A set of periodic signals is orthogonal if Equation 2.3 is satisfied for all signals in the set, where T is the period of the signals $s_i(t)$ and $s_j(t)$, and E is the signal energy as defined in Equation 2.4. The set of signals representing the orthogonal binary sequences 00 and 01 is shown in Figure 2.6.

$$\frac{1}{E}\int_0^T s_i(t)s_j(t)dt = \begin{cases} 0 & for \quad i \neq j \\ 1 & for \quad i = j \end{cases} \tag{2.3}$$

$$E = \int_0^T s_i^2(t)dt \tag{2.4}$$

Walsh codes are orthogonal codes that are generated using a Hadamard matrix. A Hadamard matrix is a square matrix with the first row all zeros, and an equal number of ones and zeros on all other rows. Equation 2.5 is used to construct a matrix of dimension $2^n \times 2^n$, where $N = 2^n$ is the length of each row in the matrix H_N. The binary sequences

in each row of the Hadamard matrix form orthogonal codes. The length of the spreading code (N) is equal to the number of orthogonal codes. This is illustrated in Equation 2.6 to generate a set of four orthogonal binary sequences.

$$H_{2N} = \begin{bmatrix} H_N & H_N \\ H_N & \overline{H_N} \end{bmatrix} \tag{2.5}$$

$$H_1 = [0], \quad H_2 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad H_4 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \tag{2.6}$$

A tree-structured method of generating orthogonal codes with different lengths was proposed in [4]. These codes are called Orthogonal Variable Spreading Factor (OVSF) codes. The set of N codes at one level of the tree $\{C_N(1) \cdots C_N(N)\}$ are generated from the previous level of the tree using Equation 2.7. This is illustrated in Figure 2.7 for a three level tree consisting of seven spreading codes. Each level of the tree contains a set of orthogonal spreading codes. The codes $\{C_2(1), C_2(2)\}$ are first generated from $C_1(1)$. Then, codes $\{C_4(1) \cdots C_4(4)\}$ are generated from codes $\{C_2(1), C_2(2)\}$. Codes at different levels of the tree are orthogonal, except in the case where a path can be traced through a particular code to the root of the tree. For example, in Figure 2.7, $C_4(1)$ is orthogonal to $C_4(2)$, $C_4(3)$, $C_4(4)$, and $C_2(2)$, but it is not orthogonal to either $C_2(1)$ or $C_1(1)$. The codes at each level of the tree have a different SF, allowing users to transmit at different data rates. The assignment of OVSF codes must be carefully managed for efficient use of the available bandwidth. As an example, consider Figure 2.7 with the maximum supported data rate $R_S = 60$ kbps using code $C_1(1)$. Two users transmitting at $R_s = 15$ kbps are assigned codes $C_4(1)$ and $C_4(3)$, utilizing one-half of the total available data rate. The system should now be able to support another user transmitting at $R_s = 30$ kbps, but no code supporting this data rate is available

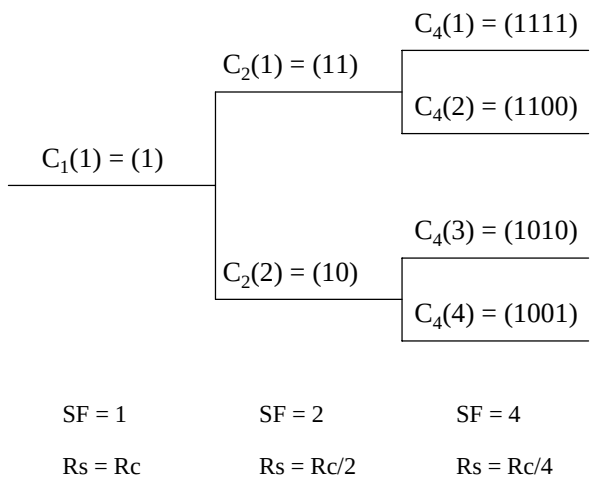

Figure 2.7: Orthogonal Variable Spreading Factor Code Tree

for use. This is because $C_2(1)$ is not orthogonal to $C_4(1)$, and $C_2(2)$ is not orthogonal to $C_4(3)$. The system could support all three users with codes $C_4(1)$, $C_4(2)$, and $C_2(2)$.

$$C_N = \begin{bmatrix} C_N(1) \\ C_N(2) \\ C_N(3) \\ C_N(4) \\ \vdots \\ C_N(N-1) \\ C_N(N) \end{bmatrix} = \begin{bmatrix} C_{N/2}(1) & C_{N/2}(1) \\ C_{N/2}(1) & \overline{C_{N/2}(1)} \\ C_{N/2}(2) & C_{N/2}(2) \\ C_{N/2}(2) & \overline{C_{N/2}(2)} \\ \vdots & \vdots \\ C_{N/2}(N/2) & C_{N/2}(N/2) \\ C_{N/2}(N/2) & \overline{C_{N/2}(N/2)} \end{bmatrix} \tag{2.7}$$

The ideal cross-correlation properties of orthogonal codes make them attractive for separating users in a DS CDMA system. Orthogonal codes are only useful in synchronous systems, as they have extremely poor cross-correlation properties when not synchronized. The downlink of a CDMA cellular network is synchronous, and systems such as IS-95 use orthogonal codes to separate users. However, multipath interference on the downlink results in unsynchronized signals at the receiver. In addition, orthogonal codes have poor autocorrelation properties making signal detection difficult.

Table 2.2: Spreading Codes for IS-95 and WCDMA

Link	Channelization Code	Scrambling Code
IS-95 Forward Link	Walsh code	Time shifted M-sequence
IS-95 Reverse Link	N/A	Time shifted M-sequence
WCDMA Forward Link	OVSF code	Gold code
WCDMA Reverse Link	OVSF code	Kasami code

Multiple Spreading Codes

A combination of PN sequences and orthogonal codes are currently utilized in CDMA cellular networks like IS-95, and will be used in future Wideband CDMA (WCDMA) Third-Generation (3G) cellular networks. The user's data signal is first spread with a unique orthogonal code or *channelization code*. The good cross-correlation properties of orthogonal codes are utilized to separate users within a cell, and each cell can use the same set of orthogonal codes. Then, the signal is spread a second time using a PN sequence or *scrambling code* that is unique for each cell. The good autocorrelation properties of PN sequences are utilized to acquire the signal, reduce multipath interference, and separate users between cells. Each cell can use either a unique PN sequence, or a time shifted version of the same PN sequence. The combination of channelization code and scrambling code specifies a particular channel. Table 2.2 lists the type of spreading codes used on the forward and reverse links for both IS-95 and WCDMA.

2.2 Third Generation Cellular Communications

Wideband CDMA (WCDMA) is the leading air interface proposal for Third Generation (3G) wireless systems. 3G wireless refers to the developing technology standards for the next generation of cellular communications systems. Current Second Generation (2G) cellular systems are primarily voice systems, with data rates up to 19.2 kbps [13]. These include Global System for Mobile Communications (GSM), IS-136 (TDMA) and IS-95 (CDMA). 3G systems will offer data rates up to 2 Mbps to support wireless packet data.

2.2.1 Basic Principles of Cellular Communications

Cellular communications systems are based upon the concept of increasing system capacity by frequency reuse. The first commercial mobile telephone service was introduced in twenty-five United States cities in 1946 [13][14]. This push-to-talk system used a single, high-power transmitter and a large tower to cover distances up to 50 km [13]. The demand for mobile telephone service soon exceeded the capacity of early systems. D. H. Ring proposed the cellular concept in a 1947 Bell Labs internal memorandum as a method to increase system capacity [14].

Cellular Frequency Reuse

Cellular communications systems reuse frequency channels throughout a geographic coverage region. This is accomplished by dividing the coverage region into smaller geographic *cells*. Each cell contains a transmitter and is assigned a subset of the total available frequency channels. The number of cells that exhaust the total available frequency channels is termed a *cluster*. To illustrate how cellular systems increase the overall system capacity, consider a set of S duplex channels. These channels are divided into N groups of k channels each where $k \leq N$. The total number of physical channels is given by Equation 2.8.

$$S = kN \tag{2.8}$$

Each cell in the system is allocated k channels, and there are N cells in a cluster. The number of cells in a cluster must satisfy Equation 2.9, where i and j are integers. The values of i and j determine the location of the nearest cells using the same subset of frequency channels. Cells using the same set of frequencies as a given cell are located by moving i cells in one direction, turning 60 degrees, and moving j cells. This is illustrated in Figure 2.8 for $i = 2$ and $j = 1$.Cellular systems typically use values of N equal to 4, 7, and 12 [13].

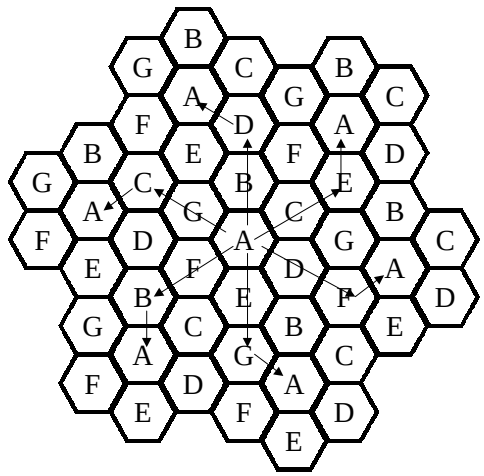

Figure 2.8: Cellular Frequency Reuse with Cluster Size N=7

$$N = i^2 + ij + j^2 \tag{2.9}$$

The cluster of N cells is duplicated M times throughout the system reusing the same frequency channels. The total system capacity is then given by Equation 2.10. This is illustrated in Figure 2.8 for $N = 7$, where cells marked with the same letter are allocated the same set of frequency channels.

$$C = MkN = MS \tag{2.10}$$

Interference and System Capacity

The degree to which frequencies are re-used within a cellular system presents a trade-off between interference and system capacity. The *frequency re-use factor* is equal to $\frac{1}{N}$, where N is the cluster size given in Equation 2.9. A larger frequency re-use factor equates to more channels per cell, increasing the overall system capacity for a fixed cell size. This also decreases the distance between cells using the same frequency channel. The signal power

received from users on the same channel in different cells is called *co-channel interference*. The ratio of desired signal power (S) to co-channel interference power (I) is given by Equation 2.11, where i_o is the number of interfering signals.

$$\frac{S}{I} = \frac{S}{\sum_{i=1}^{i_o} I_i} \tag{2.11}$$

In a thermal noise limited system, the $\frac{S}{N}$ ratio can be improved by increasing the transmitted signal power. The performance of a cellular radio system is interference limited. Increasing the transmitted power for one user will not improve system performance, but will decrease the $\frac{S}{I}$ ratio for users on the same frequency channel in other cells. In a cellular system with equal size cells, the $\frac{S}{I}$ ratio is independent of transmit power and is a function of both the cell radius (R) and the distance between cells using the same frequencies (D). The *co-channel reuse ratio* (Q) is the ratio of $\frac{D}{R}$. The value of Q for hexagonal cells can be calculated from the cluster size (N) using Equation 2.12.

$$Q = \frac{D}{R} = \sqrt{3N} \tag{2.12}$$

The $\frac{S}{I}$ for a given Q is calculated using Equation 2.13, where n is the path loss factor and i_o is the number of co-channel interfering signals. The value of n is typically between two and four [13], and $i_o = 6$ considering only the first tier of interfering hexagonal cells.

$$\frac{S}{I} = \frac{Q^n}{i_o} \tag{2.13}$$

Increasing System Capacity with CDMA

Cellular systems that use CDMA provide a higher overall system capacity than those using either Frequency Division Multiple Access (FDMA) or Time Division Multiple Access

(TDMA). The co-channel interference in a CDMA system is a function of both the co-channel reuse ratio (Q), and the correlation properties of the spreading codes. A CDMA system can use a cluster size of $N = 1$, allowing each cell to use all available frequency channels, provided the spreading codes have good cross-correlation properties. The ability to offer increased cellular system capacity for the same set of physical channels is termed *spectral efficiency*.[1] CDMA cellular systems typically utilize a combination of orthogonal spreading codes and PN sequences to provide both good autocorrelation, and good cross-correlation properties (Section 2.1.3). The spectral efficiency of CDMA is one of the reasons that WCDMA is considered the leading 3G wireless air interface over other TDMA-based proposals.

2.2.2 Evolution of Cellular System Technology

The evolution of cellular communications systems is typically classified based upon technology *generations*. The first commercially operational cellular system was deployed in Chicago, Illinois in 1983 [13][14]. First Generation (1G) cellular systems used analog communications techniques such as Frequency Modulation and FDMA. Second Generation (2G) cellular systems, deployed in the early 1990s, utilized digital communications techniques like TDMA and CDMA. The 1G and 2G cellular systems were initially designed to provide circuit switched voice service, and were later modified to provide low bit rate data service. Efforts aimed at increasing the data rates provided by 2G systems, typically through the use of higher order modulation, are termed 2.5G systems. Emerging 3G cellular standards will replace the 2G radio interface to provide higher data rates and inherent support for packet data, while leaving the 2G core network basically unchanged. Future modifications to the core network, possibly moving to an IP based infrastructure, will be the focus of Fourth Generation (4G) systems.

[1] The term *spectral efficiency* is also used to measure the data rate supported by a particular modulation scheme in a given frequency bandwidth. CDMA is less spectrally efficient than FDMA or TDMA in this sense of the term.

First Generation Systems

The Advanced Mobile Phone System (AMPS) in North America, deployed in 1983, was the first commercial cellular system. AMPS was an FM analog mobile radio system using FDMA with 30 kHz channels. A duplex channel consisted of both a forward and reverse link, and occupied a total of 60 kHz. The Federal Communications Commission (FCC) initially allocated AMPS 40 MHz of spectrum in the 800 MHz frequency band. The AMPS frequency band supported a total of 666 duplex channels. In 1989, the FCC allocated an additional 10 MHz of spectrum, increasing the total number of duplex channels to 832 [13]. AMPS currently occupies the 824-894 MHz frequency band. During the early 1980s, analog cellular systems similar to AMPS were deployed in Europe, Japan, and other parts of the world [14]. AMPS is still widely used in North America, particularly in regions that have not deployed 2G digital cellular systems. Additionally, AMPS serves as a fallback standard, allowing dual-mode 2G cell phones to connect with AMPS when digital service is not available.

Analog cellular systems had limitations in both system capacity, and support for data traffic. In 1991, the capacity of AMPS was increased by the deployment of Narrowband AMPS (NAMPS). NAMPS reduced the channel bandwidth from 30 kHz to 10 kHz by both decreasing the FM modulation index, and adding voice companding [13]. NAMPS utilized the same frequency bands as AMPS, allowing service providers to increase capacity where needed by replacing 30 kHz channels with three 10 kHz channels. In 1993, Cellular Digital Packet Data (CDPD) added support for data traffic in the AMPS frequency band. CDPD is a packet switched network that uses a 30 kHz AMPS channel to provide mobile access to packet data networks at 19.2 kbps [13]. Mobile data users are able to transmit on idle AMPS voice channels, and must hop to a different channel when a voice user begins to transmit. CDPD is termed a *packet overlay network* since it utilizes the existing base station equipment of a cellular network, and requires no additional frequency spectrum. The core network of CDPD consists of packet switches, separate from the AMPS Mobile Switching Center (MSC),

which provide connectivity to public data networks. CDPD can also be utilized as a packet overlay network for 2G digital voice systems.

Second Generation Systems

The digital cellular systems that were deployed in the early 1990s are called 2G wireless systems. 2G systems use digital modulation schemes such as Quadrature Phase Shift Keying (QPSK), and digital multiple access schemes such as TDMA and CDMA. The three primary 2G systems are TDMA[2] (IS-136), CDMA (IS-95), and Global System for Mobile Communication (GSM). The United States Digital Cellular (USDC/IS-54) system was deployed in North America in 1991. USDC had a digital TDMA data channel, but utilized the AMPS analog control channel. In 1993, IS-136 added a digital control channel for a completely digital system. TDMA 2G systems operate in the AMPS frequency band of 824-894 MHz. They use $\pi/4-$Differential QPSK (DQPSK) modulation and voice companding to provide three digital channels in each 30 kHz AMPS channel [13]. CDMA (IS-95) systems using DSSS were deployed in the United States in 1993. Like TDMA 2G systems, they operate in the AMPS frequency band. CDMA uses a spreading factor of 128, and a maximum user data rate of 9.6 kbps, resulting in a chip rate of 1.2288 Mcps. It can support a number of spread users in a channel bandwidth of 1.25 MHz. CDMA systems utilize BPSK and QPSK modulation schemes. Between 1995 and 1997, the FCC auctioned off the 1850-1990 MHz frequency band, commonly called the called Personal Communications Services (PCS) band, to support additional CDMA carriers [14].

GSM was introduced in Europe in 1990 with the intent of replacing the numerous European 1G analog systems with a single digital system. GSM operates in the 890-960 MHz frequency band, and a PCS version termed Digital Cellular System (DCS) 1800 operates in the 1.8-2.0 GHz band [13]. GSM uses Gaussian Minimum Shift Keying (GSMK), slow

[2]Both IS-136 and GSM have a TDMA access scheme. However, cellular service providers use the term TDMA to refer to IS-136.

frequency hopping, and a 200 kHz carrier bandwidth. The GSM TDMA frame structure supports eight users per carrier, at a maximum data rate of 24.7 kbps [13]. GSM is the most widely used 2G standard world wide.

Standards have been developed to both provide data service, and increase the data rate in a GSM network. The General Packet Radio Service (GPRS) is a packet overlay network designed to provide data service in a GSM network. GPRS utilizes the same frame structure as GSM, and supports a maximum data rate of 21.4 kbps [15]. The GPRS infrastructure adds two new nodes to the GSM network, the Gateway GPRS Support Node (GGSN) and the Serving GPRS Support Node (SGSN). The GGSN serves as a gateway to external data networks and tunnels data traffic to the appropriate SGSN. The SGSN is responsible for delivering data packets to the mobile user. Enhanced Data rates for GSM Evolution (EDGE) is an ongoing effort to increase the data rate of GSM. EDGE will provide data rates up to 384 kbps over the basic GSM 200 kHz channel by using higher order modulation schemes. A version of EDGE will also be implemented to increase the data rates of IS-136. Both GPRS and EDGE bring the capabilities of GSM closer to the goals of 3G, and they are often referred to as 2.5G standards. In order to be commercially feasible, 3G systems must exceed the capabilities these enhanced 2G systems.

Third Generation Systems

The intent of 3G wireless systems is to provide wireless data service with data rates of 144 to 384 kbps in wide coverage areas, and 2 Mbps in local coverage areas [16][17]. Other goals of 3G wireless systems include global mobility, support for circuit and packet switching, access via various radio interfaces, and support for different QoS levels [18][19][20]. Possible applications include wireless Internet access, video teleconferencing, and multimedia services consisting of mixed voice and data streams. It is not yet clear if 3G user terminals will more closely resemble existing cell phones, Personal Digital Assistants (PDAs), or portable

computers. 3G wireless standards are being designed with the flexibility to support a variety of devices and applications.

The International Telecommunication Union (ITU) standards for 3G wireless systems are termed International Mobile Telecommunications in the year 2000 (IMT-2000). The 1992 World Administrative Radio Conference (WARC-92) identified the 1800-2200 MHz frequency band for IMT-2000 [16][17]. The ITU IMT-2000 standards are being developed with input from regional standards organizations. These include the European Telecommunications Standards Institute (ETSI), the Telecommunications Industry Association (TIA) in the United States, the Association of Radio Industry and Business (ARIB) in Japan, and the Telecommunications Technology Association (TTA) in Korea. Each of these regional organizations has proposed IMT-2000 standards that take into account regional concerns such as evolution from existing 2G systems, and current allocation of frequency spectrum [17].

The proposed IMT-2000 standards focus on redesigning the base station to mobile user air interface. Initial proposals included both wideband CDMA and wideband TDMA access schemes [10]. At the time of this writing, wideband CDMA has emerged as the leading air interface method for developing 3G standards [10]. The wideband CDMA proposals are further divided in asynchronous schemes, and synchronous schemes. ETSI, ARIB, and TTA submitted asynchronous wideband CDMA proposals to ITU, while TIA, and TTA submitted synchronous proposals. Two international bodies have since been established to resolve differences between regional proposals. The Third Generation Partnership Proposal (3GPP) is working to standardize the asynchronous proposals from ETSI, ARIB, TTA into a single standard termed WCDMA. WCDMA uses a 5 MHz spread bandwidth, a 10 ms frame length, and OVSF codes to support different data rates. 3GPP2 is likewise standardizing the synchronous proposals from TIA and TTA into a single standard called CDMA2000. CDMA2000 uses a 1.25MHz spread bandwidth, a 20 ms frame length, and multiple carriers to support higher bit rates. The North American CDMA2000 proposal is influenced both

by compatibility issues with the synchronous IS-95 CDMA system, and existing frequency allocations in the 1800-2200 MHz band.

2.2.3 Wideband CDMA (WCDMA)

The 3GPP WCDMA air interface proposal is capable of supporting the IMT-2000 3G wireless goals. WCDMA can support users with different bit error rate (BER) requirements and data rates up to 2 Mbps. It has the capability to time multiplex steams, such as voice and data from the same source, over a single physical channel. WCDMA supports packet data transmission using both a shared access scheme and a dedicated access scheme. The WCDMA proposal uses FDD with several users able to access each 5 MHz channel using orthogonal DSSS codes.

System Architecture and Protocols

The WCDMA air interface is part of the overall IMT-2000 system architecture, which is logically divided into the core network, UMTS Terrestrial Radio Access Network (UTRAN), and User Equipment (UE). The labels Uu and Iu refer to the protocol interfaces between the UE and UTRAN, and UTRAN and core network respectively. This is illustrated in Figure 2.9. The core network contains the Mobile Switching Center (MSC), GGSN, and SGSN, which currently exist as part of the GSM and GPRS core network. In addition to switching and control functions, the core network provides access to the Public Switched Telephone Network (PSTN), and public data networks. UTRAN consists of the Radio Network Controller (RNC) and Node Base stations (Node B). These components provide wireless access for the UE. The connectivity between components in the IMT-2000 reference architecture is shown in Figure 2.10.

The protocol architecture for IMT-2000 is logically divided into the access stratum, and the non-access stratum. The interface protocols in the access stratum terminate in the

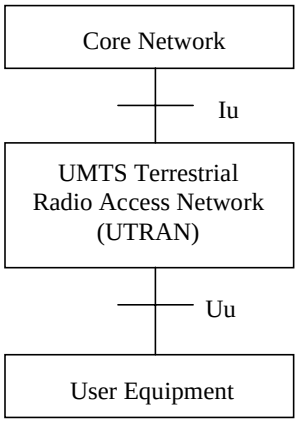

Figure 2.9: IMT-2000 System Architecture

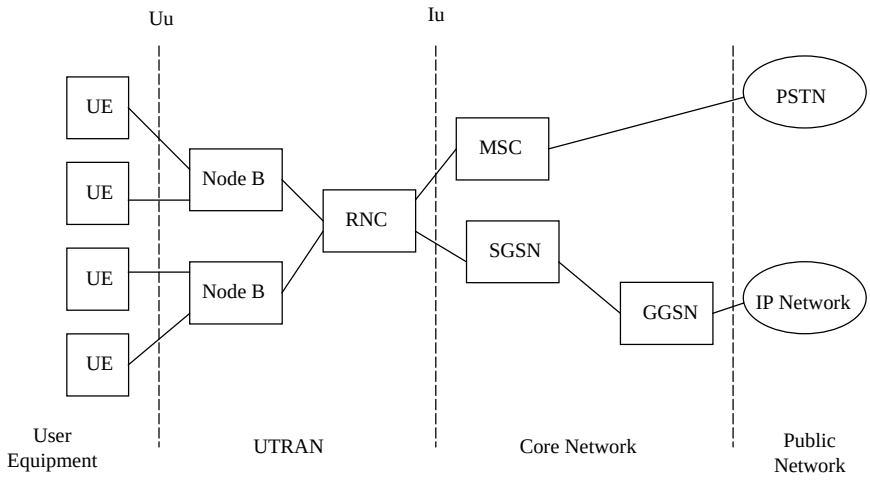

Figure 2.10: IMT-2000 Reference Architecture

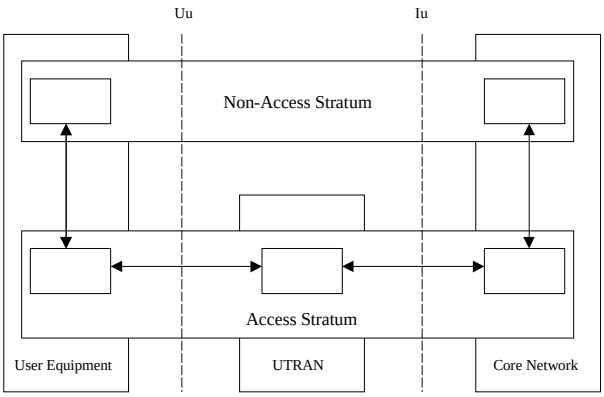

Figure 2.11: Access and Non-access Protocol Strata

UTRAN, while those in the non-access stratum terminate in the core network. Information is exchanged between the access and non-access strata at both the UE, and the core network. This logical separation is illustrated in Figure 2.11.

The protocols are also separated into the control plane and the user plane. Control plane protocols provide signalling functions such as call admission, call setup, resource management, and mobility management. User plane protocols provide the actual transfer of user data. Layer one consists of the physical layer (PHY) in both the user and control planes. Layer two contains the Medium Access Control (MAC) and Radio Link Control (RLC) sublayers in both the control and user planes. In addition, layer two of the user plane may contain the Packet Convergence Data Protocol (PCDP), and the Broadcast Multicast Control (BMC) sublayers. Layer three consists of the Radio Resource Control (RRC), Mobility Management (MM), and Call Management (CM) sublayers in the control plane. In the user plane, layer three is the Link Access Control (LAC) protocol. The control plane and user plane protocol stacks are shown in Figure 2.12, where shaded layers indicate non-access stratum protocols that terminate in the core network.

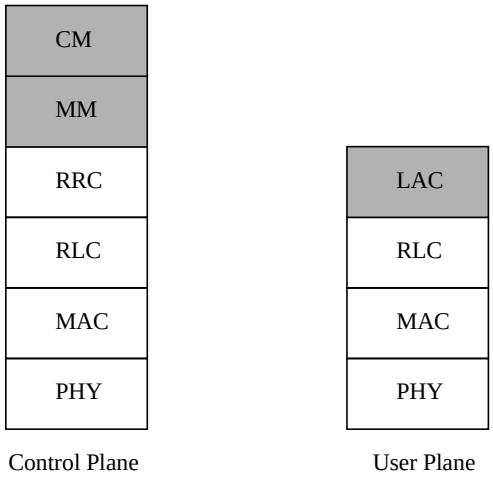

Figure 2.12: WCDMA Protocol Stacks

Physical Layer

The WCDMA physical layer uses DSSS with a chip rate of 3.84 Mcps[3] [5]. WCDMA uses a 10 ms frame, which is broken up into 16 slots of 0.625 ms. This short frame structure allows low delay voice traffic and fast control messages [21]. OVSF codes, called *channelization codes*, are used to spread the data in each frame. The data rate supported by each frame is 15×2^k kbps, corresponding to a SF of $256/2^k$, where $k = 0, 1, \cdots, 6$. The SF of the OVSF code can be changed for each 10 ms frame. The maximum data rate supported in a frame is 960 kbps. On the downlink, control and data slots are time multiplexed. QPSK modulation is used to increase the downlink data rate to 1.92 Mbps, using the same channelization code to spread both the I and Q channels. The tree of OVSF codes can be reused in each cell. A cell unique PN sequence, called a *scrambling code*, is applied after the channelization code to separate users in adjacent cells. On the uplink, user data and control data are transmitted separately on the I and Q channels. Therefore, the maximum uplink data rate is 960 kbps. Unique scrambling codes are used to separate users on the uplink. OVSF codes can not be used for this purpose, since the uplink transmissions from different users

[3] The most recent 3GPP documents specify a chip rate of 3.84 Mcps. Earlier ETSI WCDMA publications specified a chip rate of 4.096 Mcps.

are not synchronized. OVSF codes are used on the uplink to separate streams of data for a single user. WCDMA also uses a variable channel coding scheme to provide differentiated QoS. Standard traffic can be sent with no channel coding. Convolutional codes with coding rates of 1/3 to 1/2 are used for traffic with BER of 10^{-3}. Additional Reed-Solomon outer coding can be added for BER requirements of 10^{-6}[22]. In addition to these channel codes, interleaving is applied to minimize the effects of burst errors. The combination of data rate, FEC, interleaving, and multiplexing of user streams is termed a Transport Format (TF) [23].

The WCDMA physical layer provides information transfer services to the MAC and higher layers. The services provided by the physical layer are termed transport channels. The following transport channels are defined in [24].

- Random Access Channel (RACH): A contention-based uplink channel used for transmission of relatively small amounts of data.
- Common Packet Channel (CPCH): A contention-based channel used for transmission of bursty data traffic in the uplink FDD mode.
- Forward Access Channel (FACH): A common downlink channel used for transmission of relatively small amounts of data.
- Downlink Shared Channel (DSCH): A downlink channel shared by several UEs carrying dedicated control or traffic data.
- Uplink Shared Channel (USCH): An uplink channel shared by several UEs carrying dedicated control or data traffic in the TDD mode.
- Broadcast Channel (BCH): A downlink channel used for broadcast of system information in an entire cell.
- Paging Channel (PCH): a downlink channel used to broadcast control information to an entire cell, allowing efficient UE sleep mode procedures.

- Dedicated Channel (DCH): A channel dedicated to one UE in the uplink or downlink.
- Fast Uplink Signalling Channel (FAUSCH): An uplink channel used to allocate dedicated channels in conjunction with FACH.

Medium Access Control (MAC) and Radio Link Control (RLC) Layer

The MAC layer provides unacknowledged transfer of data units. The functions of the MAC layer include selection of an appropriate TF, multiplexing of user streams on transport channels, and resolving contention and access for shared channels. The MAC layer is also responsible for reallocating physical layer resources based upon direction from the RRC. The MAC layer provides logical channels to the RLC and higher layers, which are mapped onto the physical layer transport channels. The MAC logical channels are classified as either control channels or traffic channels, which provide services to the control plane and user plane respectively. The following logical channels are defined in [24].

- Broadcast Control Channel (BCCH): A downlink channel for broadcasting system information.
- Paging Control Channel (PCCH): A downlink channel used to send paging information when the network does not know the location of the UE, or when the UE is in the sleep mode.
- Common Control Channel (CCCH): A bidirectional channel for transmitting control information between the network and UEs. This channel is used for UEs with no dedicated connection.
- Dedicated Control Channel (DCCH): A bidirectional channel for transmitting control information between the network and a single UE.

- Dedicated Traffic Channel (DTCH): A point-to-point dedicated channel for the transfer of user information between one UE and the network. A DTCH can exist in both the uplink and the downlink.
- Common Traffic Channel (CTCH): A point-to-multipoint unidirectional channel for the transfer of user information to a set of dedicated UEs.

The RLC is responsible for the establishment and release of a Layer 2 connection. The functions performed by the RLC include segmentation and assembly, error correction by retransmission, flow control, duplicate detection, and in sequence delivery of higher layer Packet Data Units (PDUs). The RLC provides three data transfer modes to higher layers[24].

- Transparent Data Transfer. This service transfers higher layer PDUs with no additional protocols except segmentation and reassembly.
- Unacknowledged Data Transfer. This service transfers higher layer PDUs without guaranteeing delivery. In this mode, the RLC does not guarantee that packets will arrive in sequence or at all. Unacknowledged data transfer will not deliver duplicate packets or packets with errors.
- Acknowledged Data Transfer. This service provides reliable data transfer to higher layers. The RLC guarantees that packets will arrive error free, in sequence, with no duplicates. An option that allows out of sequence delivery is also available.

RRC Layer

The RRC layer handles control plane signalling between the UE and UTRAN. The RRC functions include the assignment, reconfiguration, and release of radio resources. This includes code allocation, admission control, handoff, and UE monitoring. The RRC is also responsible for controlling the requested quality of service. The RRC offers broadcast control services to the core network.

2.3 Quality of Service in Data Networks

Quality of Service (QoS) in a communications network refers to the ability to guarantee network performance. The performance of a data network is measured using metrics such as end-to-end latency, delay variation, and packet loss rate. Data networks are being used more frequently to carry a mix of traffic types, each with its own performance requirements. Real-time traffic has stringent latency and delay variation requirements, but packet loss is less important. Non-real-time traffic has strict packet loss requirements, but latency and delay variation are less important. Guaranteeing predictable network performance for an individual traffic source requires either absolute or relative protection from other traffic in the network.

Implementing QoS in a network involves a trade-off between guaranteeing performance, and efficiently utilizing network resources. This trade-off is illustrated by comparing the Public Switched Telephone Network (PSTN) and the Internet. The PSTN is a connection oriented network that provides strict service guarantees in terms of latency and jitter by reserving a fixed amount of bandwidth for each user. If QoS guarantees can not be met, the user is denied access to the network and receives a busy signal. This approach works well for real-time voice traffic, but is inefficient for bursty data traffic. The Internet is a connectionless network that makes efficient use of network resources by statistically multiplexing bursty data traffic. However, the Internet Protocol (IP) makes no network performance guarantees. Asynchronous Transfer Mode (ATM), Integrated Services (IntServ), and Differentiated Services (DiffServ) are methods of implementing QoS in data networks, each of which strikes a different balance between service guarantees and network efficiency.

2.3.1 Basic Principles of QoS

A user's QoS requirements place demands on *end-to-end* network performance. The presence of a single congested router between the source and the destination can result in unacceptable network performance. However, predictable *per-hop behavior* is necessary to make end-to-end service guarantees. QoS also implies the ability to offer different levels of service to different classes of packets. Packets can be classified by traffic type, and by conformance to agreed upon parameters. In order to provide QoS guarantees, a network must have the following basic capabilities.

1. Resource reservation. The network must be capable of allocating end-to-end resources to individual traffic flows or aggregate groups of traffic flows. A signalling protocol is typically used to communicate between network elements and reserve the required resources at each node. Resource reservation results in a Service Level Agreement (SLA), which specifies both the characteristics of the traffic a user may transmit and the performance the network will provide. A Call Admission Control (CAC) algorithm is utilized to determine whether to accept or reject a new call request based upon the availability of network resources.

2. Node Level service guarantees. Each switch or router between the source and destination must be able to provide predictable behavior for different classes of traffic. This implies the ability to Classify, Queue, and Schedule (CQS) traffic based upon performance requirements. Packets entering the node must first be classified according to an established scheme. This is typically accomplished through the use of header information in the packet. The packets are next placed into prioritized queues based upon their classification. Finally, a scheduling algorithm is used to remove packets from the different priority queues and transmit them out of the node.

3. Traffic monitoring. The network must have a method of identifying and handling users whose traffic does not conform with the SLA. Traffic monitoring or policing

algorithms are used to identify non-conforming packets. These packets can be dropped immediately, or marked for lower priority service by setting a priority bit in the header. The network is not required to meet the SLA performance metrics for marked packets.

2.3.2 Asynchronous Transfer Mode

ATM is a connection oriented packet switching protocol that is capable of providing reliable end-to-end quality of service. ATM utilizes a 53 byte packet format, termed the ATM cell, which has 5 bytes of header information and 48 bytes of payload. This small, fixed cell size allows for fast packet switching in hardware, and minimizes packetization delay for real-time applications. ATM can be viewed as an attempt to provide the QoS benefits of a circuit switched network in a packet switch environment. This is accomplished through the use of a signalling protocol to establish a Virtual Circuit (VC) between the source and destination nodes.

ATM Virtual Connection Service

ATM guarantees QoS by establishing a virtual path between the source and destination, much like a connection oriented circuit switch network establishes a physical path. This logical path can be either a Permanent Virtual Circuit (PVC), or a Switched Virtual Circuit (SVC), depending upon its desired duration. All packets between the source and the destination will travel over the same virtual circuit. The ATM signalling protocol establishes the virtual path by reserving resources at each switch along the route. ATM uses label switching to route packets along the virtual path based upon the Virtual Path Identifier (VPI) and Virtual Channel Identifier (VCI) fields in the cell header. Label switching potentially changes the address field of a packet at each node. Incoming packets are identified by the current VPI/VCI, switched to the correct destination port, and have their VPI/VCI fields updated for the next hop.

ATM Traffic Parameters

ATM is designed to support mixed traffic sources with differing QoS requirements. Traffic parameters are used to characterize these different traffic sources for both CAC decisions, and SLA establishment. The ATM protocol defines the following major traffic parameters.

- *Peak Cell Rate (PCR)* The PCR is the maximum data rate in cells per second.
- *Sustainable Cell Rate (SCR)* The SCR is the mean cell rate in cells per second, where $SCR \leq PCR$.
- *Maximum Burst Size (MBS)* The MBS is the maximum number of consecutive cells that the source can send at the PCR.

ATM QoS Parameters

ATM also defines QoS parameters that are used to quantify the required network performance. These parameters are agreed upon as part of the SLA. The ATM protocol defines the following major traffic parameters.

- *Cell Transfer Delay (CTD)* is the end-to-end latency a packet experiences in the network. CTD has both a fixed component that is a function of propagation delay,and a variable component that is a function of queueing delay. The SLA typically specifies a maximum CTD.
- *Cell Delay Variation (CDV)* is a measure of the variation in delay experienced by packets in the network. CDV is primarily the result of packets experiencing variable queueing delays.
- *Cell Loss Ratio (CLR)* is a measure of the percentage of lost or dropped cells. The CLR is the result of queue or buffer overflows that occur with severe congestion.

ATM Service Categories

ATM classifies traffic into five broad service categories, based upon both the source traffic parameters and the required network performance guarantees. These service categories are used in the negotiation of a SLA and provide a method to aggregate numerous types of user traffic into a manageable number of categories. ATM defines the following service categories.

- *Constant Bit Rate (CBR)* The CBR service category supports real-time traffic with a constant bit rate requirement. CBR traffic is characterized by the PCR traffic parameter. The CBR service category provides both CTD and CDV performance guarantees. This service category is suitable for voice or constant bit rate video traffic.

- *Real-Time Variable Bit Rate (rt-VBR)* The rt-VBR service category supports bursty traffic sources with real-time QoS requirements. A rt-VBR traffic source is characterized by the PCR, SCR, and MBS traffic parameters. The rt-VBR service category provides both CTD and CDV performance guarantees. This service category is suitable for variable bit rate voice or video applications.

- *Non-real-time Variable Bit Rate (nrt-VBR)* The nrt-VBR service category supports bursty traffic sources that do not have real-time delay or jitter requirements. Like rt-VBR, nrt-VBR traffic is classified using the PCR, SCR, and MBS traffic parameters. The nrt-VBR service category provides CLR performance guarantees, but does not provide CTD or CDV guarantees. This service category is suitable for bursty data traffic such as file transfers.

- *Available Bit Rate (ABR)* The ABR service category supports traffic sources that can adjust their transmission rate based upon feedback from the network. This service category is designed to dynamically allocate bandwidth that is not assigned to other service categories. ABR traffic sources must specify both a PCR and a Minimum Cell

Rate (MCR). The network provides CLR performance guarantees, but makes no CTD or CDV guarantees. This service category is not suitable for real-time traffic. Possible applications include LAN interconnections and web browsing.

- *Unspecified Bit Rate (UBR)* The UBR service category is a "best effort" service class, which makes no performance guarantees. This service category is suitable for traffic with no specific service guarantees such as LAN traffic.

A summary of the traffic parameters and QoS guarantees used for each ATM service category is given in Table 2.3.

Table 2.3: Summary of ATM Service Categories

Service Category	**Traffic Parameters**	**QoS Guarantees**	**Reserved Capacity**
CBR	PCR	CTD, CDV, CLR	Yes
rt-VBR	PCR, SCR, MBS	CTD, CDV, CLR	Yes
nrt-VBR	PCR, SCR, MBS	CLR	Yes
ABR	PCR, MCR	CLR	Yes
UBR	None	None	No

2.3.3 IP Quality of Service

IP packet switched networks currently offer a single, best effort service model that makes no performance guarantees. Packets traversing an IP network may be lost, corrupted, duplicated, or delivered out of sequence. The IP protocol depends on higher network protocol layers to provide reliable end-to-end service. The best effort IP service model has proven useful as an interface between a wide variety of link layer technologies and transport layer protocols. However, the existing IP protocol is unable to handle the demands of time-sensitive traffic such as voice and video.

The Internet Engineering Task Force (IETF) has published Request for Comments (RFCs) that add QoS capability to the current Internet infrastructure. The IETF RFCs propose two new traffic models termed Controlled Load [25] and Guaranteed Service [26].

They also describe two different methods of enhancing IP networks with QoS capabilities, Integrated Services (IntServ) [27] and Differentiated Services (DiffServ) [28].

IP Traffic Classes

The IETF has developed two service models which offer improved performance guarantees in comparison to the best effort model.

- *Controlled Load (CL)* The CL service model is designed to provide traffic applications with performance equivalent to that of a lightly loaded best effort network, regardless of the current load on the network. RFC 2211 provides the following definition of CL [25].
  - A large percentage of transmitted packets will be successfully delivered by the network.
  - The transit delay experienced by a high percentage of delivered packets will not greatly exceed the minimum transmit delay experienced by any successfully delivered packets

The CL service model can be described as a private best effort network. This service model is suitable for real-time applications such as streaming audio or video, which can use playback buffers to overcome some absolute latency. It is also suitable for applications that do not have strict delay requirements, but which will generally perform better with lower latency. This includes applications such as TCP and FTP.

- *Guaranteed Service (GS)* The GS service model supports real-time traffic applications with strict bandwidth and delay performance requirements. RFC 2212 provides the following description of the GS [26].
  - GS provides mathematically provable bounds on end-to-end queuing delay.

- Packets will be delivered within the guaranteed time provided the traffic source complies with agreed upon parameters. Packets will not be lost due to queuing overflows in the network.

The GS service model provides performance guarantees for both bandwidth and end-to-end delay. This service model is suitable for real-time applications with strict delay requirements, such as voice and video.

Integrated Services

IntServ is a connection oriented model that provides reliable end-to-end QoS in IP networks. IntServ provides QoS by reserving network resources for individual packet flows. For this purpose, a flow is defined as a stream of IP packets between applications on end hosts which have the same source and destination addresses, Transmission Control Protocol/User Datagram Protocol (TCP/UDP) port numbers, and protocol field [29]. The performance requirements of a specific flow are known as the flowspec, and can include requirements like bandwidth and delay. Unlike a CS voice network where each call has the same bandwidth and delay requirements, each flow in the IntServ model can have a unique flowspec. The IntServ model is intended to allow both time-sensitive and non-time-sensitive traffic flows to be serviced by the same IP layer.

The three main components of the IntServ model are a signaling protocol, an admission control capability, and a packet forwarding mechanism [29]. IntServ uses the Resource Reservation Protocol (RSVP) to allocate network resources to each flow [27]. If RSVP can not reserve sufficient network resources to meet the flowspec, then admission control is used to prevent the flow from entering the network. In addition, IntServ requires a packet forwarding mechanism that can classify packets, manage buffers, and schedule service in the network routers. Thus, to implement IntServ, a router must be able to maintain flow state information for each QoS-enabled traffic flow that it services.

The IntServ model suffers from scalability issues based upon providing QoS on a per flow basis [30]. IntServ requires a router to maintain state information for all flows that pass through it. This becomes a significant problem as link speeds increase to gigabit-per-second or terabit-per-second rates with a corresponding increase in the number of simultaneous flows. In addition, every router along a path must use the IntServ model of traffic service to provide end-to-end resource reservation and QoS packet forwarding. While it is possible to upgrade all routers within a small corporate network to provide IntServ based QoS, this can not be easily accomplished in the global Internet. DiffServ is a newer approach to providing QoS enhancements to IP that addresses the scalability problem.

Differentiated Services

DiffServ is a connectionless approach to providing QoS in an IP network. DiffServ addresses the scalability issues of IntServ by defining a small number of differentiated forwarding treatments that are termed per-hop-behaviors (PHBs). DiffServ also utilizes a redefined IP header field now known as the DS field to mark each packet with a DiffServ codepoint (DSCP) [31]. Packets from flows with similar QoS requirements are marked with the same DSCP. Routers treat packets as aggregate flows by assigning the same PHB to all packets with the same DSCP. The scalability of DiffServ is, therefore, limited by the number of PHBs, as opposed to the number of flows in IntServ.

The two primary components of a DiffServ network are core routers and edge routers. DiffServ simplifies the requirements of core routers by treating traffic as aggregate flows [30]. Core routers must be capable of forwarding packets using an appropriate PHB based upon the DS field value. DiffServ does not specify the use of a particular scheduling or queueing mechanism within core routers. DiffServ utilizes a small number of edge routers to perform more complex traffic control functions. Edge routers are responsible for classifying, marking, shaping or dropping packets on aggregate flows entering or leaving a DiffServ network.

The DiffServ model can be deployed across domains under the control of different network managers. Since DiffServ provides QoS on a per-hop basis, network managers in different domains are able to use different PHBs on their core routers. Service level agreements can be made between DiffServ domains regarding the amount and type of traffic offered from one domain to another. The DiffServ edge routers are responsible for policing traffic and enforcing the service level agreements. The DiffServ per-hop approach to QoS, therefore, supports scalability from a network management perspective. However, it also makes end-to-end performance guarantees more difficult.

The DiffServ model allows for both absolute and relative QoS performance [29]. Absolute DiffServ provides bounds on guaranteed performance much like IntServ. To guarantee end-to-end QoS, the DiffServ model requires a new component, termed a bandwidth broker [29]. The bandwidth broker negotiates between DiffServ domain edge routers to ensure that each domain along the route can support the QoS requirement. In contrast, relative DiffServ only guarantees the relative QoS levels between aggregate traffic flows. Packets with higher priority DSCP codes will receive better performance than those with lower priority. The relative DiffServ approach is easier to deploy incrementally than absolute DiffServ. By relaxing the performance guarantees, relative DiffServ allows DiffServ capable pockets to exist within a larger network.

2.4 Self-Similar Traffic Sources

Self-similar is a term used to describe a specific property of an object, which is preserved if the object is scaled with respect to time or space. Data traffic exhibits self-similarity in that it appears bursty across a wide range of time scales. This holds true across networks of different sizes ranging from Local Area Networks (LANs) to Wide Area Networks (WANs), as well as across different network protocols such as Ethernet, TCP, and HTTP [32][33][34]. The self-similar nature of data traffic is in sharp contrast to voice

traffic, which appears bursty at smaller time scales and gradually becomes smooth as the time scale is increased [32]. Because of this difference, the Poisson models typically used for voice traffic do not work well for data traffic. Self-similar traffic models complicate network simulation due to both long simulation times required to reach steady state, and a high degree of variability at steady state.

2.4.1 Basic Principles of Self-Similarity

Self-similarity refers to the preservation of a certain property when an object is scaled in time or space. The concept of self-similarity can be explained graphically by showing that the magnified parts of an object resemble the shape of the whole object. Park and Willinger illustrate this concept using Cantor sets as follows [35]. A two-dimensional Cantor set is drawn beginning with a solid square. Each edge of the square is scaled in size by $\frac{1}{3}$ and placed on the corners of the original square. This scaling process is done recursively to create the two-dimensional Cantor set. As shown in Figure 2.13(a), the magnified parts of the two-dimensional Cantor set resemble the whole. The one-dimensional Cantor set is constructed in a manner similar to the two-dimensional set, as shown in Figure 2.13(b). The one-dimensional Cantor set can be used to represent a traffic series $x(t)$ that is bursty at different time scales. While this deterministic approach serves well to illustrate the concept of self-similarity, a statistical measure of self-similarity is necessary for stochastic traffic sources.

Stochastic self-similarity does not require exact similarity of a time sequence at different scales. Instead, stochastic self-similarity is determined based upon the similarity of certain statistical properties at different time scales. Second-order statistics are well suited for this purpose since they capture the variability or burstiness of a random process. In particular, an autocorrelation function that decays polynomially, as opposed to exponentially, is indicative of a self-similar series [35]. The existence of non-trivial correlation at a significant

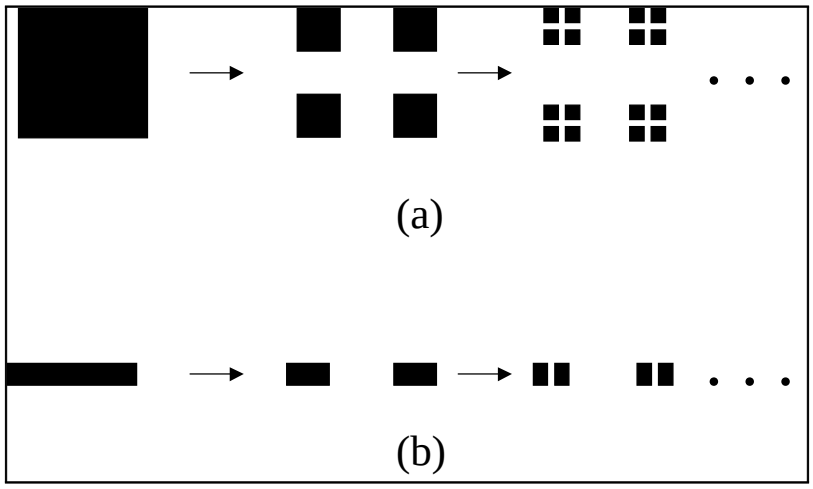

Figure 2.13: Two Dimensional (a) and One Dimensional (b) Cantor Sets

time distance is referred to as *long-range dependence*. Park and Willinger give the following mathematical definition of stochastic self-similarity [35].

Consider a wide-sense stationary random process $X(t)$ with mean $\mu = E[X(t)]$ and variance $\sigma^2 = E[(X(t) - \mu)^2]$. The autocovariance of $X(t)$ is given by Equation 2.14.

$$\gamma(r, s) = E[(X(r) - \mu)(X(s) - \mu)] \tag{2.14}$$

This wide-sense stationary process must satisfy translational invariance as expressed in Equation 2.15.

$$\gamma(r, s) = \gamma(r + k, s + k) \tag{2.15}$$

If we assume that $X(t)$ is zero mean, then the autocovariance $\gamma(r, s) = \gamma(r - s, 0)$, which is denoted as $\gamma(k)$. The random process $X(t)$ is self-similar with a Hurst parameter H, where $\frac{1}{2} < H < 1$, if Equation 2.16 is satisfied.

$$\gamma(k) = \frac{\sigma^2}{2}((k + 1)^{2H} - 2k^{2H} + (k - 1)^{2H} \tag{2.16}$$

The Hurst parameter H is often used as a measure of burstiness for a self-similar series. A complete mathematical discussion of stochastic self-similarity is presented in Park and Willinger [35].

2.4.2 Self-Similar Traffic in Data Networks

Network data traffic has been shown to exhibit self-similar properties across a variety of network scales and protocols. As a result, the concept of self-similar traffic has become important in the study of data networks. The self-similar properties of data traffic were first published in the seminal paper by Leland et. al. in 1994 [32]. Their study demonstrated that actual Ethernet LAN traffic exhibits burstiness across time scales on the order of milliseconds to hours. In contrast, exponential traffic sources became indistinguishable from white noise after aggregating packet arrivals over a few hundred milliseconds. Leland et. al. also found that increasing the number of Ethernet traffic sources did not smooth out the aggregate traffic, but actually increased the burstiness. Their study concluded that Ethernet traffic was self-similar with a Hurst parameter of approximately 0.80 over a 27-hour observation period, and a Hurst parameter ranging between 0.85 and 0.95 over a four-year observation period. These results demonstrated that aggregated LAN traffic is very different from both telephone traffic and traditional Poisson models for data traffic.

The self-similar properties exhibited by Ethernet LAN traffic have also been shown to exist for Wide Area Network (WAN) traffic. Paxson and Floyd studied traces of WAN traffic including TCP, FTP, and TELNET [33]. They found that WAN traffic was much burstier than Poisson models across a wide range of time scales. Paxson and Floyd concluded that Poisson processes are valid for modelling the arrival of user sessions, but self-similar models should be used to model the arrival of WAN packets. In particular, they found that WAN traffic fit well to a Pareto distribution with a shape parameter between 0.9 and 0.95. Crovella and Bestavros conducted a similar study of World Wide Web (WWW) traffic using the HTTP

protocol [34]. They demonstrated that HTTP traffic was self-similar on time scales of one second and above with a Hurst parameter between 0.76 and 0.83.

Self-similar data network traffic has a significant impact on network design and management. The fact that aggregated network traffic becomes more bursty, rather than smoothing out like Poisson traffic, is particularly important. The use of traditional Poisson models to determine buffer and bandwidth requirements will underestimate the actual requirements. It is therefore desirable to use self-similar traffic models to accurately provision network resources for QoS in data networks.

2.4.3 Self-Similar Traffic and Network Simulation

The simulation of self-similar network traffic satisfying the definition of Equation 2.16 is not a trivial process. A widely accepted method of generating traffic with "self-similar-like" properties is to aggregate a number of on-off traffic sources, which have on and off periods following a heavy-tailed distribution [32][33][35][36]. The resulting traffic is *asymptotically self-similar*, meaning that the autocorrelation function declines according to the power law and there is a positive correlation between widely separated events [35]. The Pareto distribution is a commonly used heavy-tailed distribution. The probability density function of the Pareto Distribution is given in Equation 2.17, where α is referred to as the shape parameter and β as the location parameter. The resulting distribution will have infinite variance if $\alpha < 2$, and infinite mean if $\alpha \leq 1$. For the case of $\alpha > 1$, the mean value of the Pareto distribution is given by Equation 2.18. The location parameter β represents the minimum value of the Pareto distribution.

$$p(x) = \alpha\beta^{\alpha}x^{-\alpha-1}; 0 < \beta < x, 0 < \alpha \tag{2.17}$$

$$E[x] = \frac{\alpha}{\alpha - 1}\beta \tag{2.18}$$

A heavy tailed distribution like the Pareto will produce many small observations and an occasional extremely large observation. The Hurst parameter, reflecting the burstiness of the Pareto distribution, can be calculated with Equation 2.19 [32]. As discussed in Section 2.4.2, Ethernet traffic is characterized by a Hurst parameter of approximately $H = 0.9$. Substituting this value of H into Equation 2.19 gives a shape parameter of $\alpha = 1.2$, which corresponds to the case of infinite variance. In practical terms, infinite variance means that the overwhelming majority of the observations are small, but the occasional large observation contributes most to the sample mean and variance.

$$H = \frac{(3 - \alpha)}{2} \tag{2.19}$$

The occasional large observations associated with heavy-tailed distributions create unique challenges for network simulation. The first challenge is a slow convergence to steady state. This happens because the infrequent, very large observations contribute heavily to the sample mean. Therefore, insufficient simulation time will result in an underestimate of the sample mean. Park and Willinger give Equation 2.20 for the magnitude of error between the sample mean A_n and the distribution mean μ as a function of the number of samples n and shape parameter α [35]. It is clear from this equation that as $\alpha \rightarrow 1$ the sample mean and distribution mean will not converge. Park and Willinger state that simulation convergence becomes impractical when α is somewhere between 1.7 and 1.5, and that, given sufficient computing resources, it may be feasible to reach simulation steady state consistently if $\alpha > 1.7$.

$$|A_n - \mu| \sim n^{\frac{1}{\alpha - 1}} \tag{2.20}$$

The second challenge associated with heavy-tailed distributions is high variability at steady state. Park and Willinger explain this as a result of the occasional very large observation as follows [35]. First, they define a *swamping observation* as one that causes the sample mean A_n to be at least twice as large as the distribution mean μ. For this to occur in a simulation with n sample observations, the swamping observation must have a value of at least $n\mu$. Assuming a Pareto distribution, they give Equation 2.21 for the probability of observing at least one swamping observation in n trials. To illustrate the high variability at steady state cased by swamping observations, consider the case where $\alpha = 1.2$ and $n = 10^5$. Substituting these values into Equation 2.21 gives a probability of observing at least one swamping observation in 10^5 samples greater than 0.01. A thorough mathematical discussion of both underestimating the sample mean and variability at steady state is given in [35].

$$p = 1 - [1 - (\frac{\alpha - 1}{n\alpha})^{\alpha}]^n \tag{2.21}$$

2.5 Review of Existing OVSF Code Assignment Algorithms

This research is motivated by the lack of public domain algorithms for the assignment of OVSF codes. One factor contributing to the lack of published algorithms is that OVSF codes are not widely used in existing CDMA systems. The IS-95 CDMA standard uses spreading codes of the same length, rather than the newer OVSF codes that were first proposed in 1997 [4]. A second, probably more significant factor is the importance of such algorithms to the performance of 3G WCDMA systems. The 3GPP technical specifications require the use of OVSF channelization codes in the WCDMA forward link, but leave the implementation details up to the equipment vendors. The ability of code assignment algorithms to efficiently support data traffic will differentiate 3G equipment from competing vendors.

While it is very likely that equipment vendors are researching OVSF code assignment, it is equally likely that commercial interests will prevent the timely publication of their results.

One of the goals of 3G wireless is to support users with different data rates in a DS CDMA system. In [37], Dahlman and Jamal present two methods of implementing variable bit rate services in a DS CDMA system. The first method, *multi-code transmission*, assigns each user a number of spreading codes to achieve a higher data rate. All of the spreading codes have the same SF, and provide the same nominal data rate $R_b = R_c/SF$. A user is assigned M different spreading codes for an overall data rate of R_bM. The second method, *single-code transmission*, assigns each user a single spreading code. Multiple data rates in this case are achieved by using codes with different SFs. The possible data rates under this scheme are constrained to $R_c/2^n$, where n is an integer. The authors conclude that multi-code transmission is preferable in the forward link, due primarily to the availability of orthogonal spreading codes. They also conclude that single-code transmission is preferable from a receiver complexity point of view. However, this work was published prior to the proposal of OVSF spreading codes.

Adachi, Sawahashi, and Okawa propose a tree-structured method of generating orthogonal spreading codes with different lengths [4]. This key paper provides a method of supporting different data rates in a CDMA system with only one spreading code per user. This is possible due to the orthogonal properties of the variable length codes, addressing the problems with single code transmission described in [37]. The tree structured codes are orthogonal, except for the case where one code is in the sub-tree of another code. This scenario is described in detail in Section 2.1.3. The 3GPP WCDMA standards utilize OVSF codes for channel separation in the forward link [5].

Cheng and Lin present an assignment scheme for OVSF codes [1]. They state that it is advantageous to assign codes to low data rate users in a manner that maximizes the available number of low SF, or high data rate, codes. Cheng and Lin define two criteria that

can be used to evaluate a code assignment scheme.

- Utilization is defined as the ratio of assigned bandwidth to overall bandwidth. A code allocation scheme that preserves more low SF codes has a better chance of providing higher utilization.
- Complexity is defined as increasing with the number of codes assigned to a single user.

The authors assume that a user can be assigned multiple OVSF codes to achieve different data rates. Their code assignment algorithm assigns as many users as possible, while minimizing the number of codes per user. This algorithm addresses the problem of initial code assignment, but does not address the problem of efficiently re-assigning codes after users disconnect.

Minn and Siu propose a dynamic OVSF code assignment algorithm that addresses the problem of *code blocking* [2]. Code blocking occurs when the capacity exists to admit a user, but no vacant orthogonal codes are available. Code blocking can occur when a number of users disconnect from the network. The authors address this problem by reassigning existing users to new codes in a manner that maximizes the available number of low SF codes. This algorithm addresses the problem of efficiently re-assigning codes, but does not address supporting different types of traffic.

Fantacci and Nannicini present an algorithm that uses dynamic assignment of OVSF codes to support both real-time and non-real-time traffic [3]. They consider real-time CBR and VBR traffic, which is represented by voice and video terminals. They model non-real-time traffic as both CBR and VBR. First, the algorithm assigns codes to real-time VBR and CBR users. Then, non-real-time users are assigned one of the remaining codes. Non-real-time users are required to either release their code, or switch to a lower data rate when another real-time user requests access to the system. Fantacci and Nannicini's algorithm provides a method for non-real-time traffic to utilize codes that are not currently assigned to

real-time users. However, it does not address the problem of efficiently sharing OVSF codes between a number of bursty traffic users.

This goal of this research is to develop an OVSF code assignment algorithm that allows bursty data sources to efficiently share WCDMA wireless resources. This algorithm will improve upon those proposed in [1][2][3] by supporting the sharing of OVSF spreading codes between bursty traffic sources. Two methods of sharing OVSF codes will be evaluated to support data sources with different QoS requirements. First, bursty traffic sources will be statistically multiplexed on a single OVSF code channel. Second, a set of OVSF codes will be dynamically reassigned between a number of bursty traffic sources. The research questions posed in Section 1.4 will be used to evaluate the performance of both methods.

2.6 Summary

This chapter has presented the theoretical background and a review of related research in the area of OVSF code assignment algorithms. The design of an algorithm to share codes efficiently between bursty data users brings together aspects of several research areas. Spread spectrum communications and the generation of orthogonal spreading codes provide a means of sharing wireless resources between variable bit rate users. Cellular communications, and the evolution of commercial cellular standards, give an understanding of 3G wireless goals and capabilities. Methods of providing QoS in data networks, including ATM, IntServ, and DiffServ, offer insight to the requirements of real-time traffic in data networks. Self-similar traffic models represent actual network traffic more realistically than Poisson models, but present new challenges in network simulation.

This research is motivated by the current lack of public-domain OVSF code assignment algorithms. This can be attributed primarily to two factors. First, the generation of OVSF codes is a relatively new idea that was first presented in 1997 [4]. Second, commercial

interests of 3G equipment vendors will likely prevent the timely publication of code assignment algorithms. The existing algorithms published in [1][2][3] address problems such as initial code assignment, code blocking, and supporting both real-time and best effort traffic sources. This research will focus on developing an improved OVSF code assignment algorithm that supports sharing spreading codes between bursty data users to improve channel utilization.

Chapter 3

Methodology and Simulation Design

This chapter presents the methodology used throughout this research. As discussed in Section 1.5, the purpose of this research effort is to develop an OVSF code assignment algorithm that supports sharing spreading codes for bursty data users. The key contribution of this research is the implementation and testing of two code sharing techniques which are not implemented in existing OVSF code assignment algorithms [1][2][3]. The new OVSF code sharing techniques investigated in this research, termed statistical multiplexing and dynamic code sharing, are described in Section 3.2.

Selecting an appropriate, proven methodology is a critical step in any research endeavor. Resource constraints prohibited implementing and testing a dynamic code assignment algorithm in prototype 3G wireless equipment. Thus, simulation was the primary tool used throughout this research. The simulation model was developed using the commercial package OPNET Modeler 7.0, by OPNET Technologies. Where feasible, analysis was used to validate simulation results. The sections in this chapter describe the simulation model developed for this research effort. Jain presents a ten-step method of systematic performance evaluation, which is well suited for evaluating the performance of a communications system through simulation [7]. I followed this systematic approach in developing and testing an

OVSF code assignment algorithm.

1. State goals and define the system
2. List services and outcomes
3. Select metrics
4. List parameters
5. Select factors to study
6. Select evaluation technique
7. Select workload
8. Design experiments
9. Analyze and interpret data
10. Present results

3.1 Problem Definition and Assumptions

The purpose of this research was to develop an OVSF code assignment algorithm that supports multiplexing of bursty data users in a 3G WCDMA system. This research problem was defined in Chapter 1. Several assumptions were necessary to limit the scope of the problem. The intent of these limiting assumptions was to keep the simulation complexity manageable, while still meeting the research goals. This section describes assumptions made in modelling the data network, wireless link, call admission, and call handoff.

3.1.1 Data Network

The simulation modeled the data network as a source or server connected to the Node Base station (Node B). This assumption limited the scope of the simulation to the wireless link between the Node B and the User Equipment (UE). Various types of data

networks could exist between the server and the Node B in an actual 3G wireless network. The performance of these networks would vary widely based upon factors such as physical location of the server, network congestion, and network QoS capabilities. Modelling the data network between the server and the Node B would require assumptions that would limit the applicability of the results. Therefore, the performance of the data network was not modelled.

3.1.2 Wireless Link

The simulation modeled the wireless link as the forward link between the Node B and the UE. Algorithms to assign OVSF codes will only be necessary on the forward link of 3GPP WCDMA systems. On the reverse link between the UE and the Node B, unique PN sequences will separate the signals from different users. Since the focus of this work is on assignment algorithms for OVSF codes, the reverse link was not modelled.

3.1.3 Call Admission

The simulation model assumed that data users are already admitted to the network and assigned either a single OVSF code, or a set of OVSF codes. This assumption limited the scope of the simulation to the data plane. The signalling functions required to model the control plane would significantly increase the complexity of the simulation, without improving the accuracy of results after a call is established. Therefore, control plane functions such as signalling and call admission were not modelled.

3.1.4 Call Handoff

The simulation did not model call handoff, limiting the scope to a cell covered by one Node B. Modelling call handoff for data users in a wireless, “always on” network requires

the implementation of a complex routing protocol such as mobile IP. It would additionally require modelling of both the data network between cells, and the signalling for handoff between cells. The complexity required to model call handoff would improve the simulation from a systems perspective, but would not improve the model of the Node B to UE forward link. Since the focus of this work is on improving the performance of the wireless link, call handoff was not modelled.

3.2 Code Sharing Techniques

The primary contribution of this research effort was the implementation and testing of OVSF code sharing techniques which are not used in currently proposed OVSF code assignment algorithms [1][2][3]. These techniques are termed *statistical multiplexing* and *dynamic code sharing*. The statistical multiplexing technique uses a shared channel to support multiple bursty traffic sources. The dynamic code sharing technique supports multiple data users by temporarily granting access to dedicated channels. These techniques differ in terms of both complexity and performance guarantees. The two different code sharing techniques developed during this research could be implemented within the 3GPP WCDMA specifications. The 3GPP specifications discuss resource management strategies using both shared and dedicated channel assignment [6], but do not discuss specific techniques of resource sharing.

3.2.1 Statistical Multiplexing

The statistical multiplexing technique allowed a number of bursty data users to share a single downlink channel. Packets arriving from bursty traffic sources were queued for transmission on a shared channel. Packets were transmitted from the queue at the channel data rate, which was determined by the OVSF code. The statistical multiplexing method

of code sharing was low in complexity, but provided no hard performance guarantees. It could therefore be used to support data traffic without specific QoS requirements, such as IP best-effort traffic. Statistical multiplexing served as a baseline of comparison for the dynamic code sharing technique.

3.2.2 Dynamic Code Sharing

The dynamic OVSF code assignment algorithm efficiently shared bandwidth between bursty traffic sources with different QoS requirements. This algorithm addressed the issue that bursty traffic sources do not make efficient use of a dedicated code or channel assignment. High QoS traffic requires a bandwidth reservation that can support its peak data rate. However, this bandwidth is not used when the bursty source is transmitting at less than its peak data rate. This algorithm addressed the problem by dynamically changing the spreading code and bandwidth assigned to high QoS traffic. Best effort traffic was allowed to utilize an orthogonal spreading code when high QoS traffic was not transmitting at peak data rate. The dynamic code sharing algorithm functioned as follows:

1. Assign a high QoS traffic source a set of variable spreading factor codes consisting of a sub-section of the OVSF tree. The highest data rate code in the set must support the peak source data rate.

2. Begin transmitting high QoS traffic on a code from the assigned set that supports the source's lowest data rate.

3. Allow best effort traffic to transmit on a code in the assigned set, which is orthogonal to the code in use by the QoS traffic source.

4. If the delay for high QoS traffic exceeds a threshold:

    - Block best effort traffic.

- Switch QoS traffic to the parent code of the code currently in use to increase the data rate.
- Go to step 3.

5. If the channel utilization for the code currently used by QoS traffic falls below a threshold:

   - Block best effort traffic.
   - Switch QoS traffic to a child code of the code currently in use to reduce the data rate.
   - Go to step 3.

The operation of the dynamic code assignment algorithm can be illustrated with a two-level OVSF tree containing three spreading codes. For example, consider the highlighted section of Figure 3.1 containing the codes $C_2(1)$, $C_4(1)$, and $C_4(2)$. These three codes have spreading factors that support data rates of 30 kbps, 15 kbps, and 15 kbps respectively. The set of codes can be used to reserve bandwidth for a high QoS traffic source with a peak data rate of 30 kbps. High QoS traffic is initially assigned to the 15 kbps channel corresponding to code $C_4(1)$. Likewise, best effort traffic is assigned to the 15 kbps channel corresponding to code C4(2). Since these two codes are orthogonal, both codes can be used simultaneously. If the delay on the high QoS 15 kbps channel exceeds a threshold, then high QoS traffic is assigned to the 30 kbps channel corresponding to code $C_2(1)$. Since code $C_4(2)$ is not orthogonal to code $C_2(1)$, best effort traffic is blocked while high QoS traffic is on the 30 kbps channel. If the utilization on the 30 kbps channel drops below a threshold, high QoS traffic is switched back to the 15 kbps code $C_4(1)$, and best effort traffic is unblocked on code $C_4(2)$. Thus, the algorithm dynamically assigns the spreading codes to both limit high QoS packet delay, and allow best effort traffic to utilize available bandwidth. This scenario is illustrated in Figure 3.2. The dynamic code sharing method can provide performance guarantees at the cost of increased complexity.

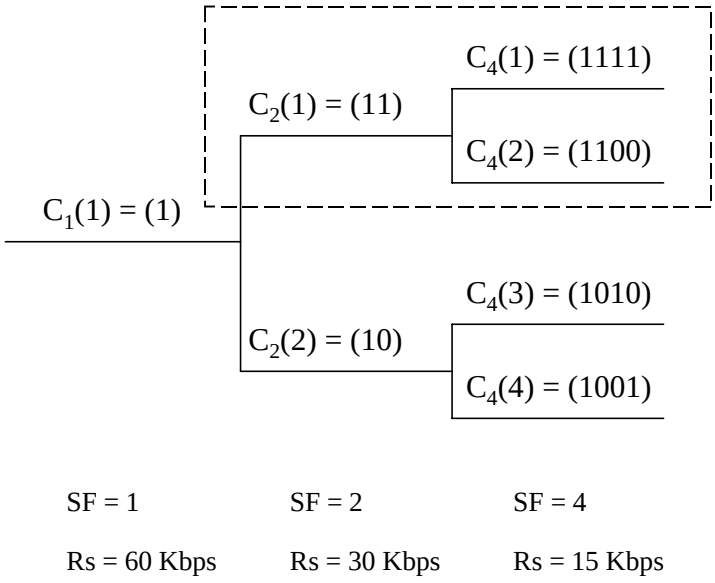

Figure 3.1: Example of OVSF Code Sharing

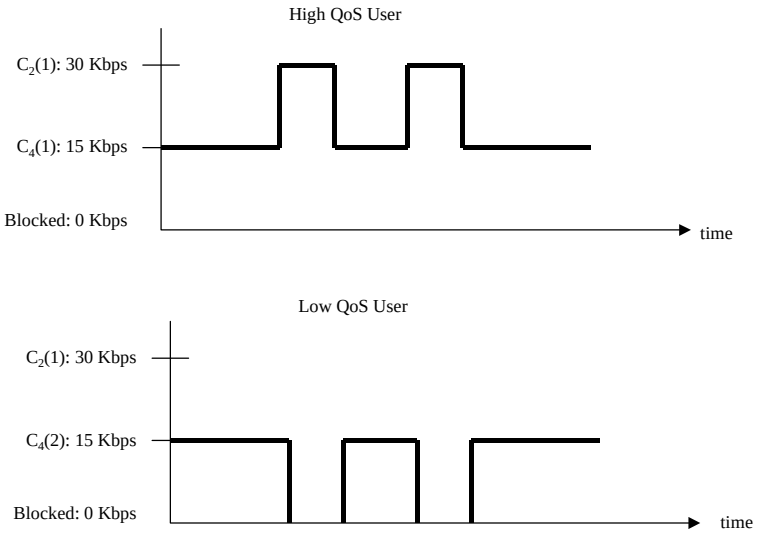

Figure 3.2: Dynamic Code Sharing Illustration

3.3 Performance Metrics

The code sharing algorithms were evaluated based upon *channel utilization,* and *end-to-end delay.* These performance metrics were defined as follows.

3.3.1 Channel Utilization

Channel utilization was defined as the ratio of channel throughput to channel capacity. Channel utilization was expressed as a percentage, and a higher value was considered better. As described in Section 3.8.1, the Baseline Scenario measured channel utilization for a single bursty traffic source. The results of the Baseline Scenario served as a benchmark for channel utilization.

3.3.2 End-to-End Delay

End-to-end delay was defined as the average time that elapses between source transmission and sink reception of a packet. End-to-end delay was measured in seconds, and a smaller value was considered better. As described in Section 3.8.1, the Baseline Scenario measured end-to-end delay for a single bursty traffic source. The results of the Baseline Scenario served as a benchmark for end-to-end delay.

3.4 Simulation Parameters

Inputs to the simulation model that were not varied during different simulation runs were termed *simulation parameters.* The values selected for these parameters affected how accurately the simulation modeled the actual system. The simulation parameters are given in Table 3.1, and discussed below.

Table 3.1: Prototype Simulation Parameters

Simulation Parameter	**Value**
Packet Size	1000 bits
Transmit Frequency	2.0 GHz
Channel Bandwidth	5.0 MHz
Channel Data Rates	120, 240, 480, 960 kbps
Transmitter Power (Node B)	100 W

3.4.1 Packet Size

The packet size for the simulation model was fixed at 1000 bits for all packets. This included a 40 bit header and 960 bits of payload. The packet size was used to calculate the transmission delay, and therefore contributed to the average end-to-end packet delay. The selected value of 1000 bits resulted in a maximum packet transmission delay of 8.33 ms which allowed the packet to be transmitted during a single 10 ms WCDMA frame.

3.4.2 Transmit Frequency

The transmit frequency for the simulation model was set at 2.0 GHz. This transmit frequency was used to calculate path loss in the simulation. The selected value is in the center of the 3G wireless frequency band.

3.4.3 Channel Bandwidth

The channel bandwidth for the simulation model was set at 5 MHz. The simulation used channel bandwidth, in conjunction with channel data rate, to calculate coding gain. The selected value is the nominal spread bandwidth for 3GPP WCDMA.

3.4.4 Channel Data Rates

The simulation model used channel data rates of 120, 240, 480 and 960 kbps. This parameter represented the maximum physical data rate of a particular wireless channel. In an actual wireless system the channel data rate would be used to carry both data and control packets, resulting in lower user data rates. The channel data rate was used in the simulation to calculate transmission delay, and coding gain. The 3GPP WCDMA standard specifies channel data rates of 15, 30, 60, 120, 240, 480, 960, and 1920 kbps. The simulation channel data rates of 120, 240, 480 and 960 kbps were selected to provide typical user data rates while keeping simulation run times reasonable. The selected values were comparable to user data rates provided on wired data networks using ISDN, DSL, or cable modems. A data rate of 1920 kbps was not used as it would have significantly increased simulation run time without improving the accuracy of the simulation results.

3.4.5 Transmitter Power

The simulation model used a transmitter power of 100 Watts per channel. This parameter was used to calculate the Signal to Noise Ratio (SNR) and the resulting Bit Error Rate (BER) for the wireless channel. The simulation used a relatively high value for transmitter power to produce an "error-free" wireless link. The use of an "error-free" wireless link allows the performance metrics of channel utilization and end-to-end delay (Section 3.3) to be measured without the impact of packet retransmissions. This was desirable since the dynamic code assignment algorithm under study did not implement packet retransmission.

3.5 Simulation Factors

Inputs to the simulation that were varied during different simulation runs are termed *simulation factors*. The simulation was run with different combinations of these factors as

described in Section 3.8. The simulation factors are summarized in Table 3.2 and discussed below.

Table 3.2: Proposed Simulation Factors

Simulation Factor	**Values**
Code Sharing Algorithm	Statistical Multiplexing, Dynamic Code Sharing
Source Data Rates	Varied between 40 kbps and 480 kbps
Number of Sources	1, 3, 5, 10, 20
Traffic Distribution	Deterministic, Exponential, On-Off

3.5.1 Code Sharing Algorithm

The simulation was run using two different code sharing algorithms. These algorithms used both the statistical multiplexing technique described in Section 3.2.1, and the dynamic code sharing technique described in Section 3.2.2.

3.5.2 Source Data Rates

The simulation was run with different source data rates between 40 kbps and 480 kbps. These values determined the packet arrival rate for a source representing a single user. The combination of source data rate and the number of sources determined the system traffic load.

3.5.3 Number of Sources

The simulation was run with 1, 3, 5, 10 and 20 traffic sources. These values represented the number of traffic sources sharing access to the wireless channel. The combination of source data rate and the number of sources determined the system traffic load.

3.5.4 Traffic Distribution

The simulation was run using deterministic, exponential, and on-off traffic distributions. These traffic distributions are described in Sections 3.7.1, 3.7.2, and 3.7.3 respectively. In addition, several on-off traffic sources were aggregated together to simulate self-similar traffic as described in Section 2.4.

3.6 Evaluation Technique

The selection of a particular evaluation technique can significantly impact the outcome of a performance evaluation. Three possible techniques of performance evaluation are analytic, simulation, and measurement [7]. These methods differ in terms of accuracy, cost, and required time. Based upon these factors, simulation was the most appropriate technique for this research effort. Measurement was easily ruled out as a feasible technique based upon both cost and required time. Commercial 3G equipment was not yet available, and the development of prototype 3G hardware was not possible within the financial and time constraints of this project. Analytic solutions typically offer less accuracy than simulation, but are also less costly and time consuming. In this case, the cost of simulation was negligible because the hardware and software required was already on-hand. Therefore, simulation was used to conduct this performance analysis and, where feasible, the results were validated analytically.

3.7 Traffic Distributions

The simulation model used *deterministic*, *exponential,* and *on-off* traffic distributions. These traffic distributions differed in terms of both analytic complexity and realism. The deterministic traffic distribution was useful in validating correct operation of the simulation,

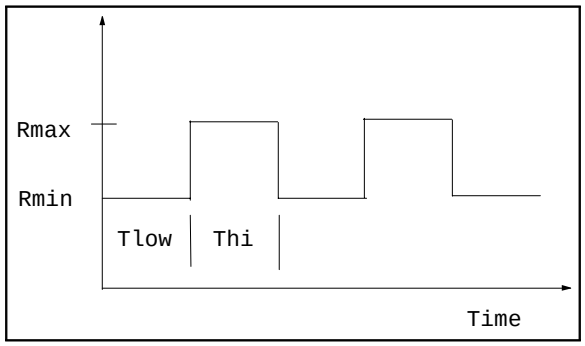

Figure 3.3: Deterministic Traffic Source

but did not accurately model data traffic. The exponential traffic distribution provided a more realistic model of bursty data traffic, which could be analyzed using traditional queuing theory. Aggregated on-off traffic sources were used to simulate self-similar traffic as described in Section 2.4. This offered the most realistic model of bursty data traffic, but could not be analyzed with traditional queuing theory.

3.7.1 Deterministic Source

The deterministic traffic source used both the Constant Bit Rate (CBR) and on-off traffic distributions provided in OPNET. These traffic distributions were combined to produce a periodic waveform. An example of a two-level deterministic source is shown in Figure 3.3. Deterministic signals with more than two levels were constructed in a similar manner.

3.7.2 Exponential Source

The exponential traffic source generated packets using the simple source node provided in OPNET. The simple source was configured with a constant packet size and exponential packet inter-arrival rate. The packet size was fixed at 1000 bits as discussed in Section 3.4. The mean data rate of the source was adjusted by varying the packet inter-arrival rate.

3.7.3 On-Off Bursty Source

The on-off traffic source generated packets using the bursty source node provided in OPNET. The bursty source was configured with a constant packet size and Pareto distributed on-off periods. The packet size was fixed at 1000 bits as discussed in Section 3.4. The on-off source generated packets with a constant inter-arrival time in the on state, and did not generate packets in the off state. The mean data rate of the source was adjusted by varying the packet interarrival rate in the on state. The burstiness of the source was adjusted by varying the parameters of the Pareto distribution for the on-off periods. Self-similar traffic was simulated by aggregating a number of on-off traffic sources.

3.8 Simulation Scenarios

The simulation was run under four different scenarios termed *baseline, statistical multiplexing, dynamic code assignment,* and *hybrid.* The primary difference between these scenarios was the code sharing technique used. All three scenarios varied the simulation factors *traffic loading* and *traffic distribution* defined in Section 3.5. The performance metrics defined in Section 3.3 were used to measure performance in each scenario.

3.8.1 Baseline Scenario

The baseline scenario simulated a single bursty data user with a dedicated wireless channel. The purpose of this scenario was to determine the maximum channel utilization possible without causing unbounded end-to-end delay. The traffic loading was increased by gradually increasing the mean data rate of the user. The highest mean data rate possible without unbounded end-to-end delay was termed the *soft channel capacity.* The ratio of soft channel capacity to physical channel capacity was the maximum channel utilization for a single bursty data user. This value served as a baseline of comparison for the statistical

multiplexing and dynamic code sharing techniques.

3.8.2 Statistical Multiplexing Scenario

The statistical multiplexing scenario simulated a number of bursty data users sharing a single wireless channel. This scenario implemented the statistical multiplexing code sharing technique described in Section 3.2.1. Both the mean data rate of each user and the number of users were varied. The purpose of this scenario was to determine the improvement in channel utilization offered by statistical multiplexing in comparison to dedicated channel assignment.

3.8.3 Dynamic Code Assignment Scenario

The dynamic code assignment scenario simulated a number of bursty data users sharing a set of OVSF spreading codes. This scenario implemented the dynamic code sharing technique described in Section 3.2.2. One improved QoS user was initially assigned a set of spreading codes that could support the user's peak data rate. When the improved QoS user was not transmitting at the peak data rate, lower QoS users were allowed to transmit on orthogonal codes in the set. Both the mean data rate of each user and the number of users were varied. The purpose of this scenario was to determine the improvement in channel utilization offered by dynamic code assignment in comparison to dedicated channel assignment.

3.8.4 Hybrid Scenario

The hybrid scenario simulated a number of bursty users sharing access to a wireless channel using both statistical multiplexing and dynamic code sharing. This scenario implemented the dynamic code sharing technique described in Section 3.2.2. Ten high QoS

on-off traffic sources were statistically multiplexed and the resulting traffic flow was assigned a set of spreading codes that could support the aggregate, peak data rate. Ten low QoS on-off traffic sources were likewise statistically multiplexed. The aggregate traffic flow of low QoS sources was allowed to transmit on an orthogonal spreading code when the multiplexed high QoS sources were not transmitting at the peak data rate. The data rate of the high QoS users was varied to measure performance under different traffic loads. The purpose of this scenario was to determine the improvement in channel utilization offered by combining statistical multiplexing and dynamic code sharing in comparison to dedicated channel assignment. The statistical multiplexing of on-off traffic sources in this scenario resulted in the most realistic model of bursty Internet traffic used in this research effort. As discussed in Section 2.4, aggregating a number of on-off traffic sources with heavy-tailed distributions is a widely accepted method of simulating self-similar Internet traffic.

3.9 Simulation Design

The simulation was designed in OPNET Modeler 7.0 using a top-down approach. OPNET uses a hierarchical structure of network scenarios, nodes and processes. The top level of the simulation graphically depicted the network containing the Node Base Station (Node B) and the mobile User Equipment (UE) nodes. The Node B and UE were designed at the node level using both built-in OPNET processes and user-defined processes. User defined processes, such as the dynamic code assignment algorithm, were defined at the process level using state diagrams. The simulation actions in each state of the user-defined process were defined with a combination of C code and built-in OPNET functions. This section presents an overview of the simulation design at the network, node, and process levels.

Figure 3.4: Simulation Top Level Design

3.9.1 Top Level

The simulation top-level consisted of a base station node and up to twenty mobile UE nodes. Each mobile UE node collected statistics for a single traffic source. It was decided that twenty mobile UE nodes were sufficient to model the number of simultaneous data users in a single cell coverage area. The base station generated deterministic, exponential, and on-off traffic as described in Section 3.7. Packets were transmitted from the base station over a WCDMA wireless link using a different spreading code for each mobile user. Mobile UE nodes received packets on the channel defined by their assigned spreading code. Statistics were collected for both channel utilization and end-to-end packet delay as described in Section 3.3. The simulation top level design for a scenario with six mobile UE nodes is shown in Figure 3.4.

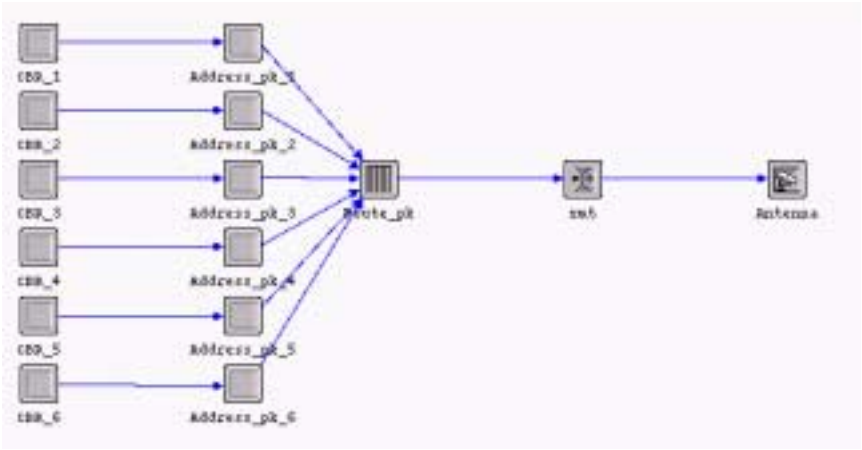

Figure 3.5: Base Station Node for Statistical Multiplexing

3.9.2 Base Station Node (Node B)

The Node B consisted of traffic generators, user-defined address processes, a user-defined code assignment process, a radio transmitter, and an antenna. Two different Node B base stations were developed for the two code sharing techniques described in Section 3.2. A node diagram of the Node B for the statistical multiplexing technique is shown in Figure 3.5. The source processes generated deterministic, exponential, and on-off traffic as described in Section 3.7. Each packet had a 32-bit address field, an 8-bit Type of Service (TOS) field, and 960 bits of data. The Address_pk process was a user-defined process that assigned a destination address and TOS to each packet. The Address_pk state diagram is shown in Figure 3.6. The route_pk process for the statistical multiplexing code sharing technique is a First In First Out (FIFO) queue. The xmt process is a transmitter with a single CDMA channel, and the antenna is an omnidirectional antenna.

The Node B for the dynamic code assignment case is shown in Figure 3.7. The primary differences between this BS node and the statistical multiplexing BS in Figure 3.5 are in the route_pk and xmt processes. The route_pk process contained the user-defined dynamic code assignment process, which is described in detail in Section 3.9.3. The xmt node defined several channels with associated data rates and spreading codes as described

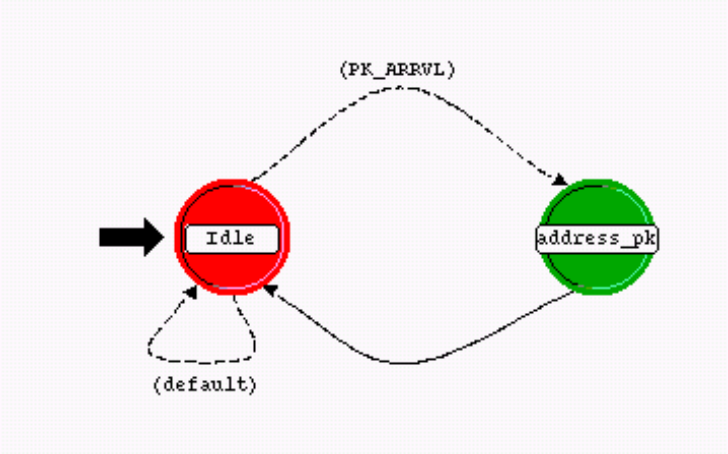

Figure 3.6: Address_pk Process State Diagram

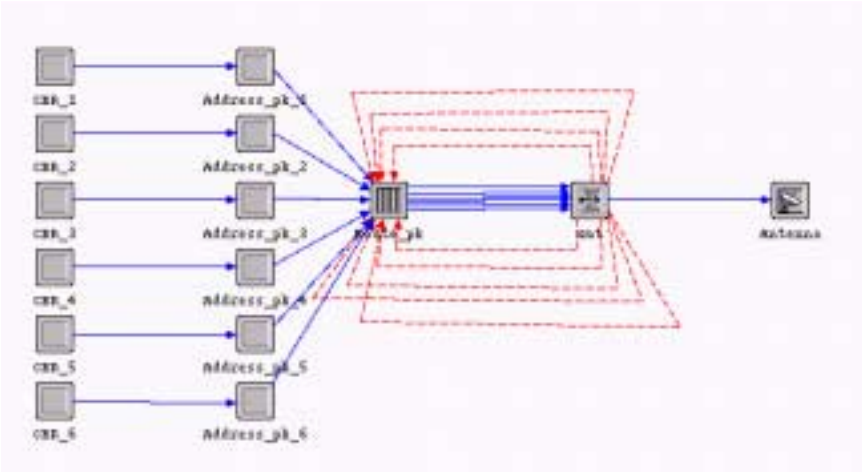

Figure 3.7: Base Station Node for Dynamic Code Assignment

in Section 3.9.4. The Node B used OPNET statistic wires to pass transmitter queue sizes, and channel busy statistics to the route_pk process. These feedback statistics enabled the dynamic performance of the algorithm. The route_pk dynamic code assignment process is described in detail in Section 3.9.3.

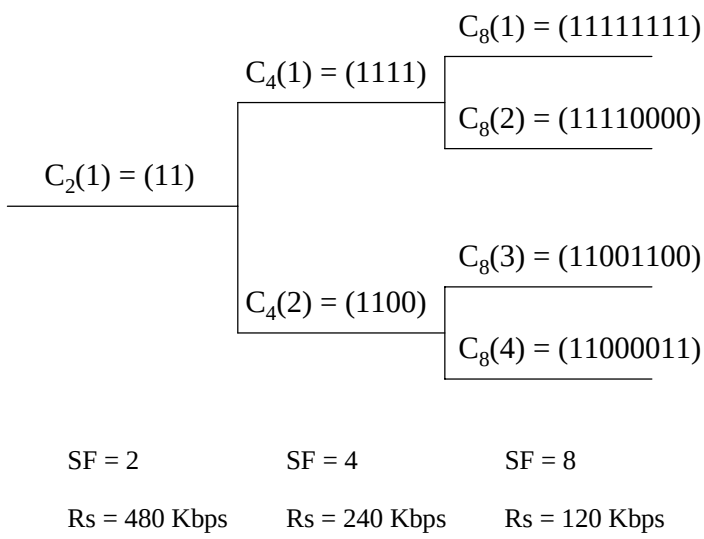

Figure 3.8: OVSF Code Tree for Dynamic Code Assignment Algorithm

3.9.3 Dynamic Code Assignment Process

The route_pk dynamic code assignment process routed packets to output ports, which have a one-to-one mapping to transmitter channels. This process implemented the algorithm given in Section 3.2.2. The transmitter channels corresponded to a three-level OVSF code tree with data rates of 120 kbps, 240 kbps, and 480 kbps as illustrated in Figure 3.8. Figure 3.9 shows the state diagram for the user-defined route_pk process. The function of each state in Figure 3.9 is as follows:

Init_rte: This state initializes the code assignment table. High QoS traffic is initially assigned to code $C_8(1)$, medium QoS traffic to code $C_4(2)$, and low QoS traffic to code $C_8(2)$. Codes $C_2(1)$, $C_4(1)$, $C_8(3)$, and $C_8(4)$ was initially blocked. After initializing the code assignment table, the process entered the idle state.

Route_pk: This state is entered when an interrupt corresponding to a packet arrival occurs. The route_pk state performs the following functions.

- Get pointer to packet

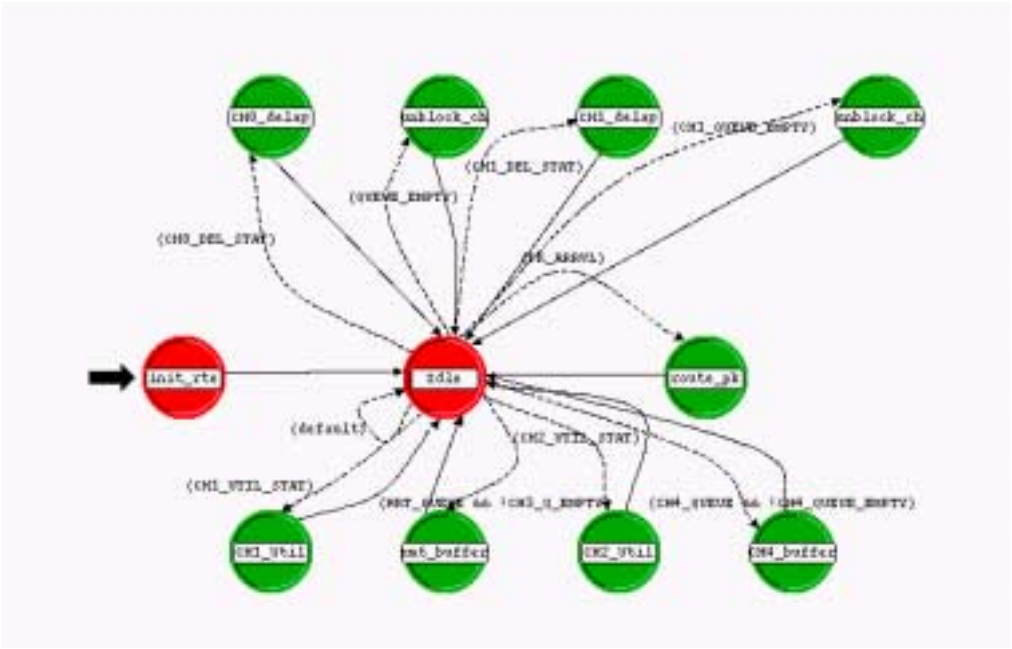

Figure 3.9: Dynamic Code Assignment Process State Diagram

- Read packet address
- Look up output code channel
- If channel not blocked, transmit packet
- Else, queue packet

CH0_ delay: This state is entered when the average queuing delay for the transmitter channel corresponding to code $C_8(1)$ exceeds a threshold. The CH0_delay state updates the code assignment table as follows.

- High QoS traffic assigned to code $C_4(1)$
- Block codes $C_4(1)$, $C_8(1)$, and $C_8(2)$

Note that it is necessary to block code $C_4(1)$, the 240 kbps channel, because the 120 kbps high QoS channel will still have packets in the transmitter queue. Since codes $C_4(1)$

and $C_8(1)$ are not orthogonal, the 240 kbps channel must be blocked until the queue on the 120 kbps channel empties. Blocking code $C_8(1)$ prevents any additional packets from queuing on the 120 kbps channel.

Unblock_ch_1: This state is entered when the transmit buffer is empty on the $C_8(1)$ 120 kbps channel. The unblock_ch_1 state performs the following functions.

- Unblock code $C_4(1)$
- Transmit packets in code $C_4(1)$ buffer

CH1_util: This state is entered each time the channel busy statistic is received for the code $C_4(1)$ 240 kbps channel. The state uses the channel busy statistic to calculate a windowed time average of channel utilization. After an initial warm-up period, the CH1_util state performs the following functions:

- Calculate channel utilization for code $C_4(1)$ 240 kbps channel
- If utilization < 40 % update code assignment table as follows
  - High QoS traffic assigned to code $C_8(1)$
  - Unblock codes $C_8(1)$, and $C_8(2)$
  - Block code $C_4(1)$

Xmt_buffer: This state is entered when the code $C_8(2)$ 120 kbps non-real-time channel statistic is received, and the queue for code $C_8(2)$ is not empty. The xmt_buffer state prevents the queuing of packets at the transmitter queue for the low QoS 120 kbps channel. Non-real-time packets are queued in this process and are only passed to the transmitter channel if the transmitter queue size is less than 1. This function allows the non-real-time channel to be blocked instantaneously when real-time traffic switches to the 240 kbps channel. The xmt_buffer state performs the following function.

- If code $C_8(2)$ transmitter queue size < 1, then send packet at head of code $C_8(2)$ queue to transmitter

CH1_delay: This state is entered when the average queuing delay for the transmitter channel corresponding to code $C_4(1)$ exceeds a threshold. The CH1_delay state updates the code assignment table as follows.

- High QoS traffic assigned to code $C_2(1)$
- Block codes $C_2(1)$, $C_4(1)$, $C_4(2)$, $C_8(1)$, and $C_8(2)$

Note that it is necessary to block code $C_2(1)$, the 480 kbps channel, because the 240 kbps high QoS channel will still have packets in the transmitter queue. Since codes $C_2(1)$ and $C_4(1)$ are not orthogonal, the 480 kbps channel must be blocked until the queue on the 240 kbps channel empties. Blocking code $C_4(1)$ prevents any additional packets from queuing on the 240 kbps channel.

Unblock_ch_4: This state is entered when the transmit buffer is empty on the $C_4(1)$ 240 kbps channel. The unblock_ch_4 state performs the following functions.

- Unblock code $C_2(1)$
- Transmit packets in code $C_2(1)$ buffer

CH4_util: This state is entered each time the channel busy statistic is received for the code $C_2(1)$ 480 kbps channel. The state uses the channel busy statistic to calculate a windowed time average of channel utilization. After an initial warm-up period, the CH4_util state performs the following functions:

- Calculate channel utilization for code $C_2(1)$ 480 kbps channel

- If utilization < 40 % update code assignment table as follows
  - High QoS traffic assigned to code $C_4(1)$
  - Unblock codes $C_4(1)$, and $C_4(2)$
  - Block code $C_2(1)$

Xmt_buffer_2: This state is entered when the code $C_4(2)$ 240 kbps medium QoS channel statistic is received, and the queue for code $C_4(2)$ is not empty. The xmt_buffer state prevents the queuing of packets at the transmitter queue for the medium QoS 240 kbps channel. Non-real-time packets are queued in this process and are only passed to the transmitter channel if the transmitter queue size is less than 1. This function allows the medium QoS channel to be blocked instantaneously when real-time traffic switches to the 480 kbps channel. The xmt_buffer_2 state performs the following function.

- If code $C_4(2)$ transmitter queue size < 1, then send packet at head of code $C_4(2)$ queue to transmitter

3.9.4 Transmitter Channels

The xmt node defined two 120 kbps channels, two 240 kbps channels, and one 480 kbps channel. These channels represent codes $C_8(1)$, $C_8(2)$, $C_4(1)$, $C_4(2)$ and $C_2(1)$ in Figure 3.8. The transmitter channels are defined in OPNET with a channel table. Each row in the table corresponds to one physical channel. The OPNET channel table for the Node B transmitter is shown in Figure 3.10.

3.9.5 Mobile User Equipment Node

The mobile UE node consists of an antenna, a radio receiver, a custom process to filter out packets by address, and a sink. The block diagram of the UE node is shown in Figure

(channel) Table

data rate (bps)	packet formats	bandwidth (kHz)	min frequency (I	spreading code	power (W)
120,000	all formatted, un	5,000	2,000	1.0	100
120,000	all formatted, un	5,000	2,000	2.0	100
240,000	all formatted, un	5,000	2,000	3.0	100
240,000	all formatted, un	5,000	2,000	4.0	100
480,000	all formatted, un	5,000	2,000	5.0	100

5 Rows Delete Insert Duplicate Move Up Move Down

Details Promote Cancel OK

Figure 3.10: Transmitter Channel Table

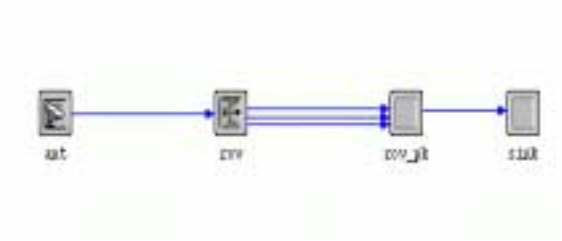

Figure 3.11: Mobile User Equipment Node

3.11. The antenna uses the default omnidirectional antenna pattern. The radio receiver has the same channels described in the BS transmitter node. The rcv_pk process reads the destination address of incoming packets on the shared channel. Packets addressed to the specific UE are passed to the sink, and those addressed to other UE nodes are destroyed. The sink collects statistics for traffic received and end-to-end delay.

3.10 Summary

This chapter has presented the methodology used to design and test an OVSF code assignment algorithm. The significant contribution of this research was the implementation

and testing of two code sharing techniques which are not implemented in existing OVSF code assignment algorithms [1][2][3]. The new OVSF code sharing techniques investigated in this research, termed statistical multiplexing and dynamic code sharing, were described in Section 3.2. The performance of these new code sharing techniques was measured in terms of channel utilization and end-to-end delay.

Simulation was the primary tool used throughout this research. The statistical multiplexing and dynamic code sharing techniques were implemented and tested in the commercial simulation package OPNET Modeler by OPNET Technologies, Inc. The simulation model was developed using the ten step methodology presented in [7]. Simulation parameters were selected to accurately model the WCDMA forward link. Factors that were varied in the simulation included the code sharing algorithm and traffic loading levels.

Chapter 4

Simulation Validation and Verification

This chapter describes the methods used to ensure the simulation model is both correctly implemented and representative of the real system. These two steps are termed *model verification* and *model validation* [7]. Section 4.1 discusses the use of simple deterministic input signals to verify correct operation of the simulation. Section 4.2 presents a theoretical analysis that is used to validate the simulation model. The analysis compares theoretical and simulation results for both deterministic and exponential traffic sources. Section 4.3 presents the results of simulation pilot runs to determine both stopping criteria and the effects of random number generator seeds.

4.1 Model Verification

Model verification is the process of determining if a simulation model functions correctly. This includes such tasks as debugging the computer code, testing for logic errors, and testing the functionality of different modules. As discussed in Section 3.9, the simulation was designed in OPNET Modeler 7.0 using a top-down, modular approach. This approach simplified the task of model verification since each module can be tested independently.

Each node and process in the simulation was tested to verify that it functioned correctly. This was accomplished by running short simulations and collecting statistics at various points in the model. Standard OPNET processes were tested to ensure they functioned as described in the software documentation. The user-defined dynamic code assignment process was tested to verify that it correctly implemented the algorithm presented in Section 3.2.2.

4.1.1 Deterministic Traffic Scenario

The simulation model was run with deterministic traffic sources to verify correct operation. In this scenario, the network consisted of a base station and three mobile users. This number of mobile users was sufficient to test performance of the algorithm without incurring long simulation run times. The base station transmitter had five channels corresponding to the variable spreading factor codes $C_1(1)$, $C_2(1)$, $C_2(2)$, $C_4(1)$, and $C_4(2)$ in Figure 4.1. These five codes supported a total bandwidth of 60 kbps. The bandwidth could be allocated to a single 60 kbps user on $C_1(1)$, to two 30 kbps users on codes $C_2(1)$ and $C_2(2)$, or to a 30 kbps user and two 15 kbps users on codes $C_2(2)$, $C_4(1)$, and $C_4(2)$. These data rates represent the lowest data rates supported in WCDMA and were selected to reduce simulation run times while testing the algorithm.

Traffic was generated in the base station in the form of packets with 32-bit address fields, 8-bit Type of Service (TOS) fields, and 1000 bits of payload. The BS transmitted traffic to three mobile users identified as *High QoS One*, and *Medium QoS One,* and *Best Effort One*. The parameters of the traffic generated for each mobile user are summarized in Table 4.1. The traffic for High QoS One was a combination of Constant Bit Rate (CBR) and on-off traffic sources as shown in Figure 4.2. The traffic for Medium QoS One and Best Effort One were CBR at 15 kbps and 10 kbps respectively.

The following scenario describes the desired operation of the dynamic code assignment

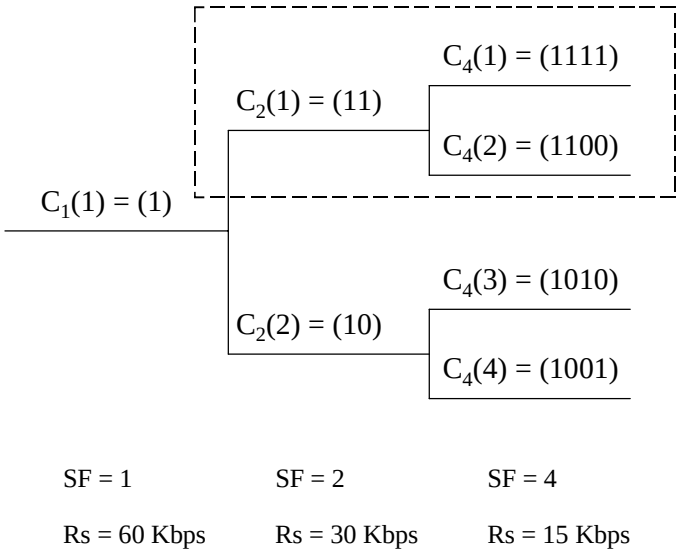

Figure 4.1: BS OVSF Code Tree for Verification Simulation

Table 4.1: Deterministic Traffic Parameters

Source	**Destination**	**TOS**	**Traffic Type**
A	High QoS One	QoS	CBR 10 kbps
B	High QoS One	QoS	On-off, t=5-16, 10 kbps
C	High QoS One	QoS	On-off, t=8-10, 20 kbps
D	High QoS One	QoS	On-off; t=12-14, 20 kbps
E	Medium QoS	Medium QoS	CBR 15 kbps
F	Best Effort	Best Effort	CBR 10 kbps

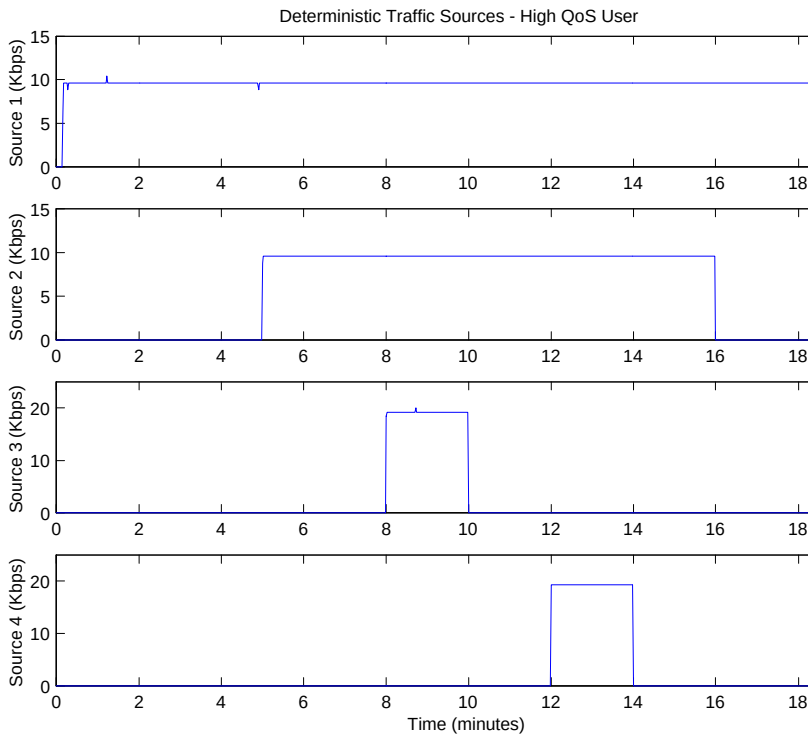

Figure 4.2: Traffic Generated for High QoS User

algorithm under the deterministic traffic load.

- *Time t=0 minutes:* When the simulation begins, the BS is transmitting CBR traffic to mobile High QoS One at 10 kbps on a code $C_4(1)$. The BS is also initially transmitting 15 kbps CBR traffic to mobile Medium QoS One on code $C_2(2)$, and 10 kbps CBR traffic to mobile Best Effort One on code $C_4(2)$.

- *Time t=5 minutes:* The base station begins transmitting an additional 10 kbps of on-off traffic to mobile High QoS One on code $C_4(1)$. The aggregate 20 kbps of high QoS traffic should flood the 15 kbps channel, causing the algorithm to switch High QoS One to code $C_2(1)$. At the same time, Best Effort One traffic should be blocked since code $C_4(2)$ is not orthogonal to code $C_2(1)$.

- *Time t=8 minutes:* The base station begins transmitting an additional 20 kbps of on-off traffic to mobile High QoS One on code $C_2(1)$. The aggregate 40 kbps of high

QoS traffic should flood the 30 kbps channel, causing the algorithm to switch High QoS One to code $C_1(1)$. At the same time, Medium QoS One traffic should be blocked since code $C_2(2)$ is not orthogonal to code $C_1(1)$.

- *Time t=10 minutes:* The 20 kbps on-off traffic to High QoS One stops and the 60 kbps channel utilization should drop below 40 percent. This should trigger the algorithm to switch High QoS traffic to the code $C_2(1)$ and unblock the Medium QoS traffic on code $C_2(2)$. The queued traffic for mobile Medium QoS One should saturate the 30 kbps channel.

- *Time t=12 minutes:* The base station begins transmitting an additional 20 kbps of on-off traffic to mobile High QoS One on code $C_2(1)$. The aggregate 40 kbps of high QoS traffic should flood the 30 kbps channel, causing the algorithm to switch High QoS One to code $C_1(1)$. At the same time, Medium QoS One traffic should be blocked since code $C_2(2)$ is not orthogonal to code $C_1(1)$.

- *Time t=14 minutes:* The 20 kbps on-off traffic to High QoS One stops and the 60 kbps channel utilization should drop below 40 percent. This should trigger the algorithm to switch High QoS traffic to code $C_2(1)$ and unblock the Medium QoS traffic on code $C_2(2)$. The queued traffic for mobile Medium QoS One should saturate the 30 kbps channel.

- *Time t=16 minutes:* The 10 kbps on-off traffic to High QoS One stops and the 30 kbps channel utilization should drop below 40 percent. This should trigger the algorithm to switch High QoS traffic to code $C_4(1)$ and unblock the Best Effort traffic on code $C_4(2)$. The queued traffic for mobile Best Effort One should saturate the 15 kbps channel. Finally, the utilization of the best effort channel should drop to 66 percent when the queued packets for mobile Best Effort One have all been transmitted.

4.1.2 Deterministic Traffic Results

The correct operation of the code assignment algorithm can be shown by observing the transmitter channel utilization. Figure 4.3 shows the transmitter channel utilization for each of the five code channels and demonstrates the correct operation of the algorithm. From top to bottom, the graphs in Figure 4.3 show the channel utilization for codes $C_4(1)$, $C_2(1)$, $C_1(1)$, $C_4(2)$, and $C_2(2)$ respectively. The BS is initially transmitting to High QoS One on code $C_4(1)$, Medium QoS One on code $C_2(2)$, and Best Effort One on code $C_4(2)$. At time t=5 minutes, the BS switches the High QoS One to code $C_2(1)$ and blocks Best Effort One on code $C_4(2)$. At time t=8 minutes, the BS switches High QoS One to code $C_1(1)$, and blocks Medium QoS One on code $C_2(2)$. At time t=10 minutes, the BS switches High QoS One back to code $C_2(1)$ and unblocks Medium QoS on code $C_2(2)$. Queued traffic for Medium QoS One floods code $C_2(2)$. The cycle of switching High QoS One to $C_1(1)$ and blocking Medium QoS on $C_2(2)$ repeats from time t=12 minutes to 14 minutes. Finally, High QoS One is switched to code $C_4(1)$ at time t=16 minutes and Best Effort is unblocked on code $C_4(2)$. These results match the desired operation of the dynamic code assignment algorithm described in Section 4.1.1, verifying that the algorithm is performing correctly.

4.2 Model Validation

Model validation is the process of determining if a simulation model is representative of the real system. A simulation can be validated using expert intuition, real system measurements, or theoretical results [7]. Comparing simulation outputs and measurements from a real system is the most reliable way of validating a simulation model. Real system measurements were not available in this research since resources were not available for prototype 3G wireless equipment. Comparing simulation and theoretical results was the primary method used to validate the simulation model. Theoretical analysis of the system was conducted

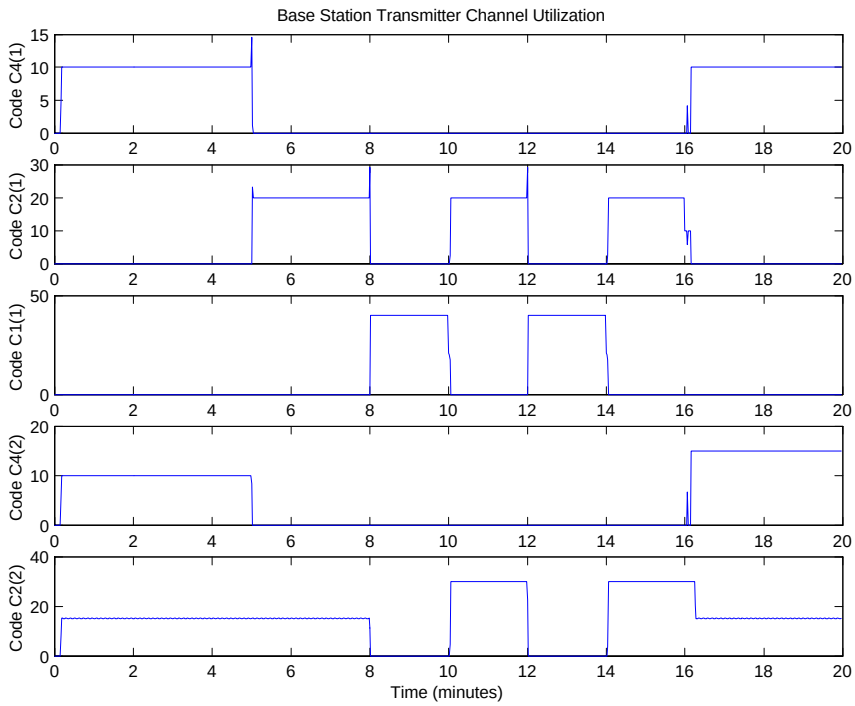

Figure 4.3: Base Station Transmitter Channel Utilization

using both deterministic and exponential traffic sources.

4.2.1 Deterministic Analysis

A theoretical analysis using deterministic traffic sources was conducted to validate correct operation of the simulation model. This analysis determined the theoretical improvement in channel utilization offered by the dynamic code assignment algorithm in comparison with a dedicated code assignment scheme. The improvement in channel utilization is a function of the percent of time that best effort traffic is allowed to transmit. This can be determined analytically for any two adjacent levels of an OVSF code tree and a two-level, bursty QoS source. The best effort source can transmit during time periods in which the QoS source data rate is less than or equal to that of the lower data rate code. Since the dynamic algorithm transitions between any two adjacent levels of the tree in the same manner, the analytic results for a two-level tree will scale to a multiple-level code tree.

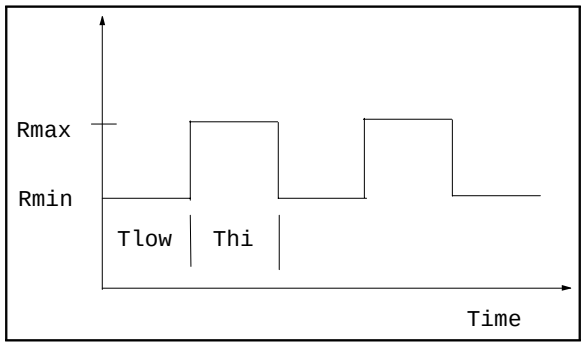

Figure 4.4: Deterministic Signal for QoS Traffic

We begin the analysis by considering a two-level section of an OVSF tree, consisting of three codes, such as the highlighted section of Figure 4.1. A two-level deterministic source is used to model QoS traffic, as shown in Figure 4.4. The QoS source is a periodic square wave with $R_{Max} \leq R_c$ and $R_{Min} \leq R_c/2$, where R_c is the highest data rate of the three codes in the code tree section. The values for $R_{Max}, R_{Min}, T_{hi,}$ and T_{low} are constants. The best effort traffic source is analytically modelled as Constant Bit Rate (CBR) traffic with data rate $R_{be} = R_c/2$, under the assumption that best-effort traffic is always available.

The channel utilization for the dedicated assignment of the signal in Figure 4.4 to a two-level section of the OVSF code tree is given by Equation 4.1, where D is the duty cycle defined in Equation 4.2. The channel utilization with a dedicated code assignment scheme serves as a baseline of comparison for the dynamic code assignment algorithm.

$$U_{dedicated} = D\frac{R_{Max}}{R_c} + (1 - D)\frac{R_{Min}}{R_c} \tag{4.1}$$

$$D = \frac{T_{hi}}{T_{hi} + T_{low}} \tag{4.2}$$

The dynamic code assignment algorithm improves channel utilization by allowing best effort traffic to transmit on an orthogonal code when QoS traffic is not transmitting on the high data rate code. The improvement in channel utilization offered by the dynamic code

assignment algorithm is equal to the channel utilization of the best-effort traffic as given by Equation 4.3. The total channel utilization for the dynamic algorithm is given by Equation 4.4.

$$U_{be} = (1 - D)\frac{R_{be}}{R_c} \tag{4.3}$$

$$U_{dynamic} = D\frac{R_{Max}}{R_c} + (1 - D)\frac{R_{Min} + R_{be}}{R_c} \tag{4.4}$$

The improvement in channel utilization offered by the dynamic code assignment algorithm is a function of the QoS traffic source duty cycle. As the duty cycle approaches a maximum value of one, the improvement offered by dynamic code assignment approaches zero. In this extreme case, the QoS source is no longer bursty and therefore makes efficient use of the assigned bandwidth. To illustrate the improvement offered by dynamic code assignment for different duty cycles, consider the case where $R_c = R_{Max} = 30$ kbps, $R_{Min} = 10$ kbps, and $R_{be} = 15$ kbps. Substituting these values into Equations 4.1 and 4.4 results in Equations 4.5 and 4.6 respectively. Equation 4.5 represents the channel utilization for dedicated code assignment, and Equation 4.6 is the channel utilization for the dynamic code assignment algorithm.

$$U_{dedicated} = D + \frac{1}{3}(1 - D) \tag{4.5}$$

$$U_{dynamic} = \frac{5D + 25}{30} \tag{4.6}$$

Equations 4.5 and 4.6 are plotted in Figure 4.5, illustrating the effect of duty cycle on the improvement in channel utilization. The theoretical improvement in channel utilization as a function of the duty cycle is given by Equation 4.7. The duty cycle can take on

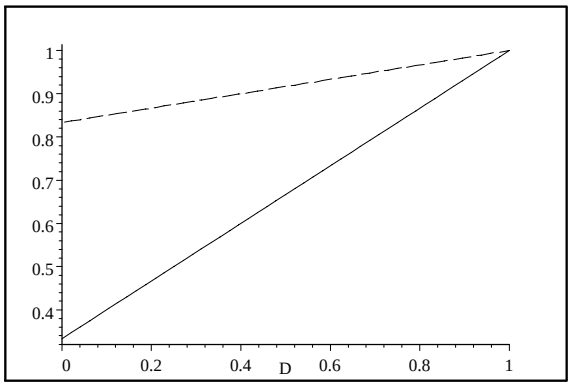

Figure 4.5: Comparison of Channel Utilization as a Function of Duty Cycle (D) Using Dedicated Code Assignment (Eqn.4.5 solid) and Dynamic Code Assignment Algorithm (Eqn. 4.6 dashed)

values between zero and one, resulting in maximum and minimum improvements in channel utilization of 50% and 0% respectively.

$$U_{improvement} = \frac{1 - D}{2} \tag{4.7}$$

The simulation was run using deterministic traffic sources in order to compare the theoretical and simulation results. The QoS traffic was a deterministic source as shown in Figure 4.4 with values of $R_{Max} = 30$ kbps, and $R_{Min} = 10$ kbps. The best-effort traffic was CBR with $R_{be} = 15$ kbps. The duty cycle of the QoS traffic was varied between 0.1 and 0.9. The simulation modeled the wireless forward link between a base station and two mobile users. This number of mobile users was sufficient to test performance of the algorithm without incurring long simulation run times. The base station transmitter had three channels corresponding to the variable spreading factor codes $C_2(1)$, $C_4(1)$, and $C_4(2)$ in Figure 4.1. Code $C_2(1)$ supports a data rate of 30 kbps while codes $C_4(1)$ and $C_4(2)$ each support 15 kbps. These data rates represent the lowest data rates supported in WCDMA, and were selected to reduce simulation run times while testing the algorithm. Traffic is generated in the base station in the form of packets with a 32 bit address field, an 8 bit Type of Service

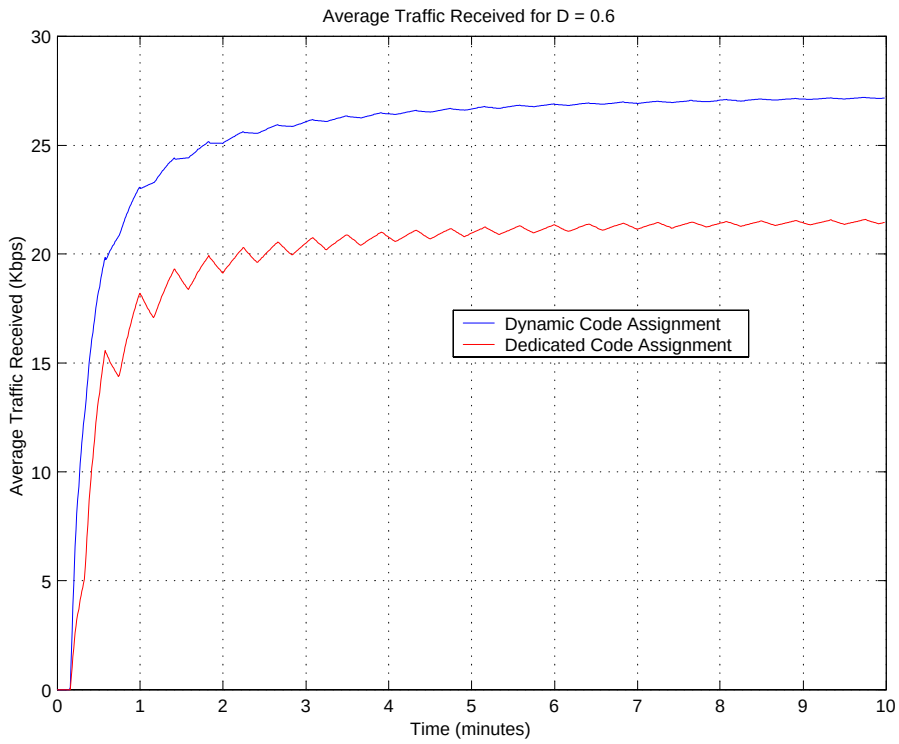

Figure 4.6: Comparison of Average Traffic Received for D=0.6

(TOS) field, and 960 bits of payload.

The average traffic received using both dedicated code assignment and the dynamic code assignment algorithm was collected for each run of the simulation. Figure 4.6 shows the simulation results for average traffic received for a duty cycle of $D = 0.6$. With this duty cycle the dynamic algorithm achieved an average throughput of approximately 27.3 kbps, and the dedicated assignment achieved an average throughput of 21.3 kbps. The corresponding improvement in channel utilization is calculated as $\frac{27.3-21.3}{30} \times 100 = 20\%$. This matches the theoretical improvement calculated by Equation 4.7 as $\frac{1-0.6}{2} \times 100 = 20\%$.

Figure 4.7 shows a comparison between the theoretical and simulation results for improvement in channel utilization as the duty cycle is varied. In each case, the simulation results are very close to the theoretical results. The difference between the theoretical and simulation results can be accounted for by the algorithm's inability to instantly switch between the high and low data rate codes. The agreement between analytic and simulation

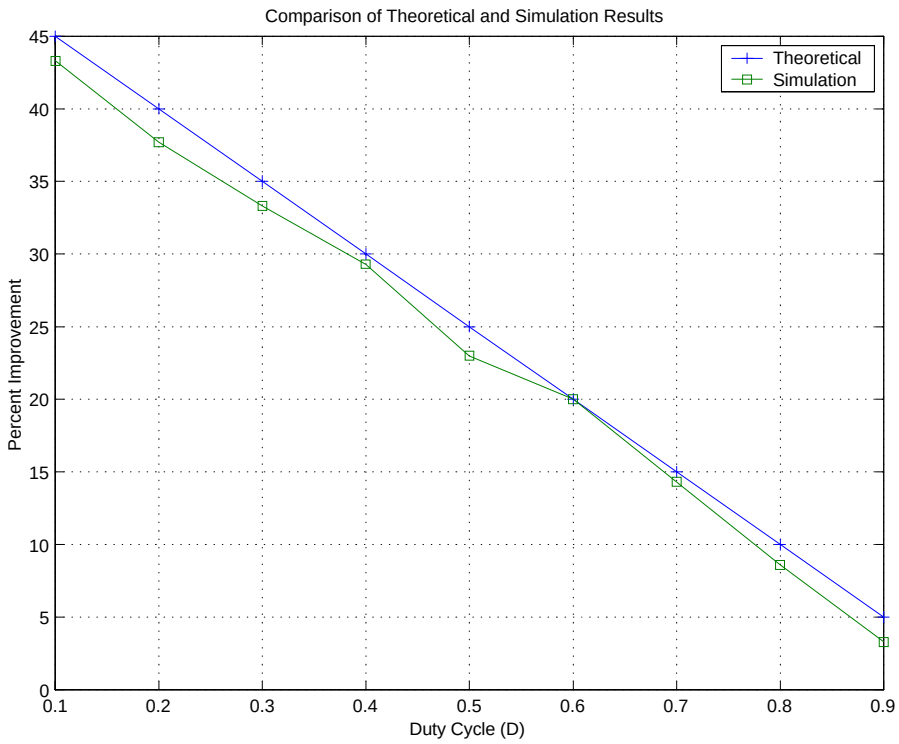

Figure 4.7: Comparison of Theoretical and Simulation Results for Improvement in Channel Utilization

results verify the correct operation of the dynamic code assignment algorithm for a two-level code tree.

The deterministic analysis scales easily to a dynamic code assignment algorithm using a multiple-level code tree. Consider the case of the three-level code tree shown in Figure 4.1. The dynamic code assignment algorithm can share this set of codes between three users, which we will call High QoS, Medium QoS, and Best Effort. The High QoS user is modeled with a periodic pulse train as shown in Figure 4.8. The High QoS user transmits at 10 kbps, 20 kbps, and 40 kbps with $Tlow = Tmed = Thi$. This deterministic signal will cause the High QoS user to cycle between codes $C_4(1)$, $C_2(1)$, $C_4(1)$, $C(1)$, $C_4(1)$ etc. The improvement in channel utilization offered by the dynamic code assignment algorithm is a function of the amount of time the Medium QoS and Best Effort users can transmit. The Medium QoS user can only transmit when the High QoS user is transmitting on codes $C_2(1)$

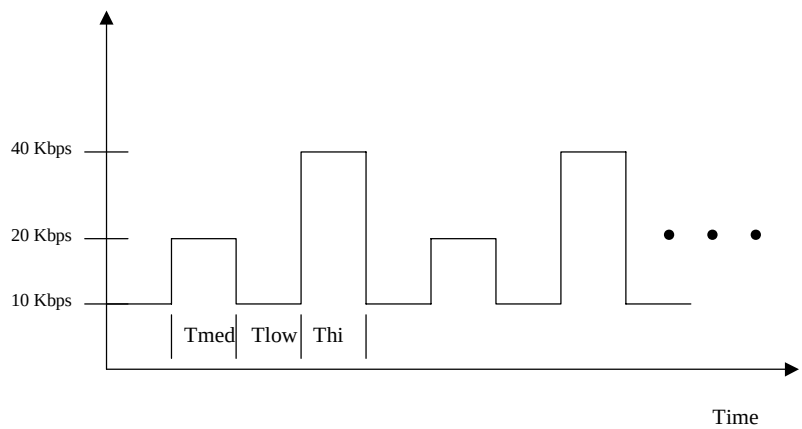

Figure 4.8: Three-level Determinstic Signal for Bursty High QoS Source

or $C_4(1)$. The Best Effort user can only transmit when the High QoS user is transmitting on code $C_4(1)$. The Medium QoS user is modelled as CBR traffic at 20 kbps. The Best Effort user is modelled as CBR traffic at 15 kbps.

The channel utilization for a dedicated code assignment scheme with a three-level tree is given by Equation 4.8, where R_c is the maximum data rate of the code tree (60 kbps) and D_{low}, D_{med}, and D_{hi} are given by Equations 4.9, 4.10, and 4.11 respectively.

$$U_{dedicated} = \frac{R_{low}D_{low} + R_{med}D_{med} + R_{hi}D_{hi}}{R_c} \tag{4.8}$$

$$D_{low} = \frac{T_{low}}{T_{low} + T_{med} + T_{hi}} \tag{4.9}$$

$$D_{med} = \frac{T_{med}}{T_{low} + T_{med} + T_{hi}} \tag{4.10}$$

$$D_{hi} = \frac{T_{hi}}{T_{low} + T_{med} + T_{hi}} \tag{4.11}$$

The channel utilization for the Medium QoS user is given by Equation 4.12, and that

for the Best Effort user is given by Equation 4.13.

$$U_{medQoS} = \frac{R_{medQoS} D_{med}}{R_c} \tag{4.12}$$

$$U_{be} = \frac{R_{be} D_{low}}{R_c} \tag{4.13}$$

The channel utilization for the dynamic code assignment algorithm is the sum of $U_{dedicated}$, U_{medQoS} and U_{be} as given in Equation 4.14. The theoretical improvement in channel utilization is the sum of U_{medQoS} and U_{be} as given in Equation 4.15.

$$U_{dynamic} = \frac{(R_{low} + R_{be})D_{low} + (R_{med} + R_{medQoS})D_{med} + R_{hi} D_{hi}}{R_c} \tag{4.14}$$

$$U_{improvement} = \frac{R_{be} D_{low} + R_{medQoS} D_{med}}{R_c} \tag{4.15}$$

The simulation model was run with the three-level deterministic signal to compare theoretical and simulation results for a multi-level code tree. The High QoS traffic was a deterministic source as shown in Figure 4.8 with values of $R_{hi} = 40$ kbps, $R_{med} = 20$ kbps, and $R_{low} = 10$ kbps. The Medium QoS traffic was CBR at 20 kbps, and the Best Effort traffic was CBR at 15 kbps. The base station transmitter had five channels corresponding to the variable spreading factor codes $C_1(1)$, $C_2(1)$, $C_2(2)$, $C_4(1)$, and $C_4(2)$ in Figure 4.1. Code $C_1(1)$ supports a data rate of 60 kbps, codes $C_2(1)$ and $C_2(2)$ each support 30 kbps, and codes $C_4(1)$ and $C_4(2)$ each support 15 kbps. Traffic was generated in the base station in the form of packets with 32-bit address fields, an 8-bit Type of Service (TOS) fields, and 960 bits of payload.

The simulation model output the average traffic received for each of the three users.

Table 4.2: Channel Utilization for Three-level Code Tree

	Theoretical	Simulation
High QoS	33.3%	29.7%
Medium QoS	25%	24.5%
Best Effort	12.5%	5.5%
Improvement	37.5%	36.4%

In each case, the average traffic received was divided by the maximum channel capacity of 60 kbps to calculate the average channel utilization. The theoretical values of channel utilization and improvement in channel utilization were calculated using Equations 4.8, 4.12, 4.13 and 4.15. A comparison of the theoretical and simulation results given in Table 4.2 show that they match well. The differences between theoretical and simulation results can be attributed to the inability of the simulation model to transition instantly between states. The agreement between analytic and simulation results validate the correct operation of the dynamic code assignment algorithm for a three-level code tree.

This section has presented an analysis of the dynamic code assignment algorithm using deterministic traffic sources. This analysis began with a two-level code tree and was extended to a multiple level code tree. The agreement between theoretical and simulation results has served to validate the simulation model under a load of deterministic traffic. This analysis is extended once again in the following section to include exponential traffic sources.

4.2.2 Exponential Analysis

A theoretical analysis using exponential traffic sources was conducted to validate the simulation model under a stochastic traffic load. This analysis expands upon the deterministic analysis presented in the previous section. We begin the analysis by considering the three-level code tree in Figure 4.1, which supports data rates of 15, 30 and 60 kbps. The High QoS packet arrival rate follows the exponential probability density function in Equation 4.16, where α is the mean arrival rate. The average throughput for the High QoS user is

equal to α. The average channel utilization for a dedicated code assignment scheme is given by Equation 4.17, where R_c is the maximum data rate of the code tree.

$$e^{-\alpha}\frac{\alpha^k}{k!} \tag{4.16}$$

$$U_{dedicated} = \frac{\alpha}{R_c} \tag{4.17}$$

The Medium QoS user is able to transmit when the High QoS user is transmitting at less than 30 kbps. The probability that the Medium QoS user can transmit is given by Equation 4.18. The average throughput for the Medium QoS user, with mean arrival rateR_{medQoS}, is given by Equation 4.19.

$$P_{medQoS} = \int_0^{30} e^{-\alpha}\frac{\alpha^k}{k!}dk \tag{4.18}$$

$$T_{medQoS} = R_{medQoS}P_{medQoS} \tag{4.19}$$

The Best Effort user is able to transmit when the High QoS user is transmitting at less than 15 kbps. The probability that the Best Effort user can transmit is given by Equation 4.20. The average throughput for the Best Effort user, with mean arrival rateR_{be}, is given by Equation 4.21.

$$P_{be} = \int_0^{15} e^{-\alpha}\frac{\alpha^k}{k!}dk \tag{4.20}$$

$$T_{be} = R_{be}P_{be} \tag{4.21}$$

The simulation was run with exponential traffic sources to compare theoretical and

simulation results. The High QoS source was exponential with a mean arrival rate that varied between 10 kbps and 40 kbps for different runs of the simulation. The Medium QoS user was exponential with a mean arrival rate of 30 kbps. The Best Effort source was exponential with a mean arrival rate of 15 kbps. The base station transmitter had five channels corresponding to the variable spreading factor codes $C_1(1)$, $C_2(1)$, $C_2(2)$, $C_4(1)$, and $C_4(2)$ in Figure 4.1. Code $C_1(1)$ supports a data rate of 60 kbps, codes $C_2(1)$ and $C_2(2)$ each support 30 kbps, and codes $C_4(1)$ and $C_4(2)$ each support 15 kbps. Traffic is generated in the base station in the form of packets with 32-bit address fields, 8-bit Type of Service (TOS) fields, and 960 bits of payload.

A comparison of the theoretical and simulation results for exponential traffic sources validates correct operation of the simulation. Figure 4.9 shows that the theoretical and simulation results match closely for all three traffic source. The results are also intuitively correct. The High QoS user is almost always on code $C_4(1)$ when transmitting at a mean data rate of 10 to 15 kbps. This allows the Medium QoS user to transmit on code $C_2(1)$ and the Best Effort user to transmit on code $C_4(2)$. The High QoS user is mostly on code $C_2(1)$ when transmitting at a mean data rate of 20 to 30 kbps. This allow the Medium QoS user to transmit on code $C_2(2)$, but blocks the Best Effort user on code $C_4(2)$. Likewise, the High QoS user blocks the Medium QoS user when transmitting at a mean data rate of 35 to 40 kbps. Figure 4.10 compares the theoretical and simulated improvement in channel utilization for exponential traffic sources. The theoretical and simulation results in Figure 4.10 also match closely. This analysis has validated correct operation of the simulation under a stochastic traffic load.

4.3 Simulation Pilot Runs

Simulation pilot runs were made to check dependence upon random number generator seeds and to determine appropriate run times. These pilot runs were conducted with both

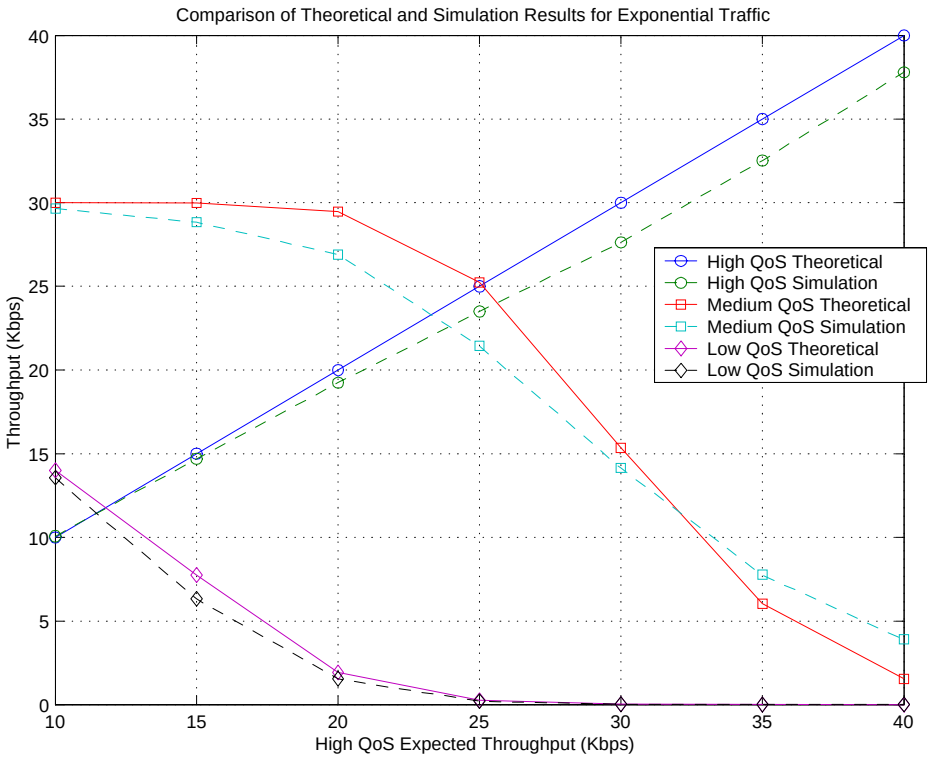

Figure 4.9: Comparison of Theoretical and Simulation Results for Exponential Traffic

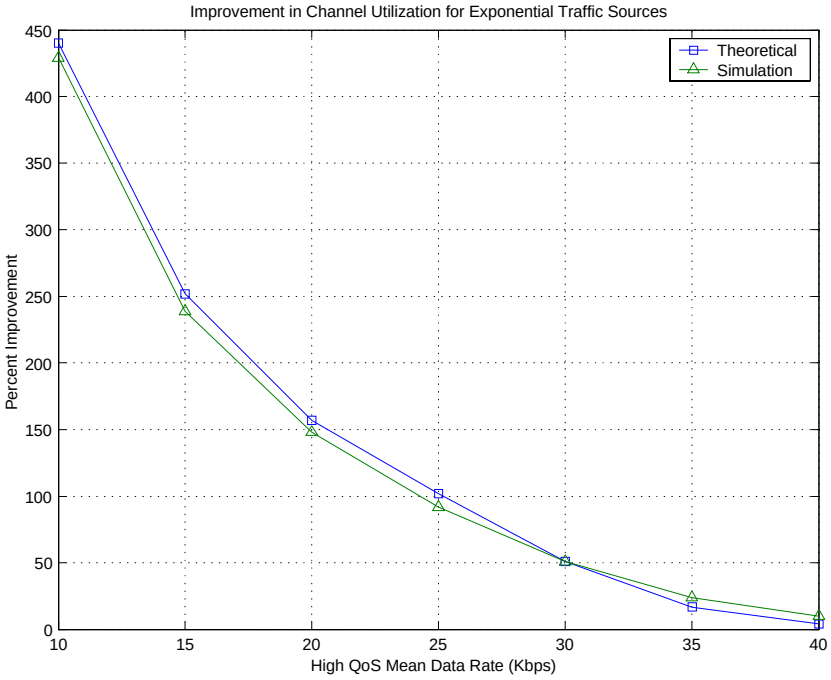

Figure 4.10: Theoretical and Simulation Improvement in Channel Utilization for Exponential Traffic

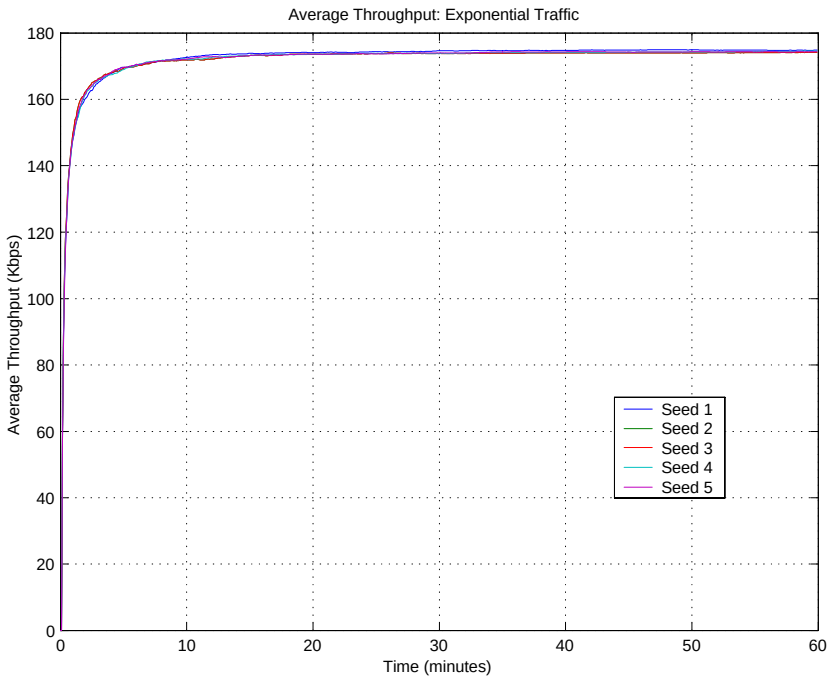

Figure 4.11: Average Throughput for Exponential Traffic Source

exponential and self-similar traffic source, as described in Section 3.7. Figure 4.11 shows the average throughput results for exponential traffic using five different seeds. The plots with different seed values are not distinguishable from each other, illustrating that the simulation model is not affected by random number generator seeds under exponential traffic loads. Figure 4.11 also shows that the average throughput has reached a steady state value after approximately 30 minutes. A simulation run time of 60 minutes will therefore be used with exponential traffic sources.

Simulation using self-similar traffic sources presents unique challenges in comparison with exponential traffic sources. Park and Willinger discuss problems with long simulation times required to reach steady state, and high variability between simulation runs [35]. As discussed in Section 2.4, Internet traffic fits a model with a shape parameter of $\alpha = 1.2$. Park and Willinger state that the simulation time required to reach steady state becomes impractical as the shape parameter decreased to the range of $1.5 < \alpha < 1.7$. To test this statement, the simulation was run with self-similar traffic sources and the shape parameter

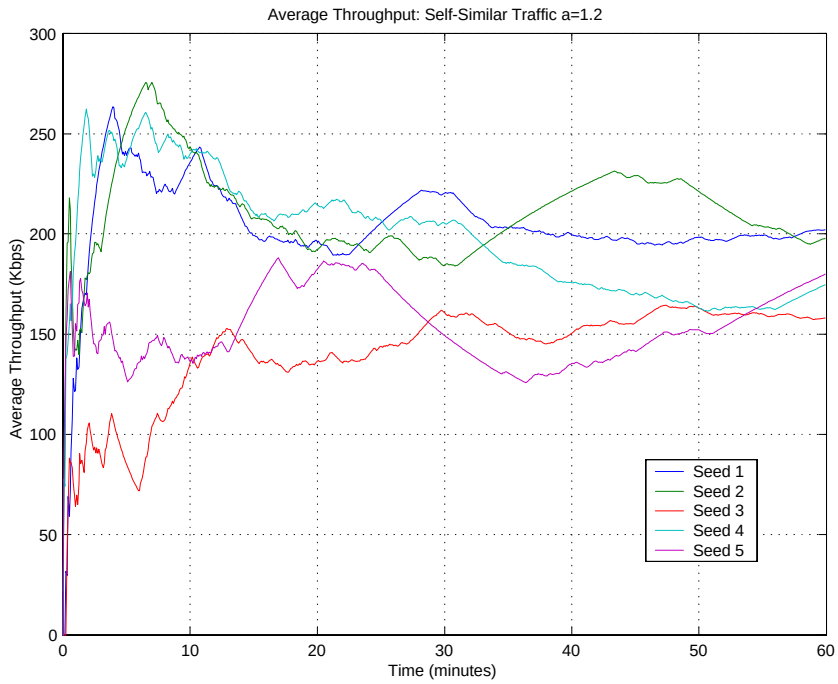

Figure 4.12: Average Throughput for Self-Similar Source $\alpha = 1.2$

was varied between $\alpha = 1.2$ and $\alpha = 1.9$. Figure 4.12 shows the average throughput for $\alpha = 1.2$ using five different seed values. The graph displays both a lack of steady state and a high variance between runs with different seed values. Figure 4.13 shows the same results for $\alpha = 1.8$ and five seed values. While there is still some variance between runs with different seeds, the simulation appears to be reaching steady state.

Figure 4.14 summarizes the simulation results for all values of α that were tested. The simulation was run with five seeds for each value of the shape parameter α. The simulation output the average throughput for each seed value. The mean and standard deviation of these five values were calculated. Figure 4.14 plots the ratio of the standard deviation to the mean as the shape parameter α is varied. This ratio serves as a measure of burstiness, which affects both the ability to reach steady state and variance between simulation runs. These results support the claim by Park and Willinger that it may not be practical to reach steady state with a shape parameter below $\alpha = 1.7$.

The result of these pilot runs present a dilemma between accurately modelling bursty

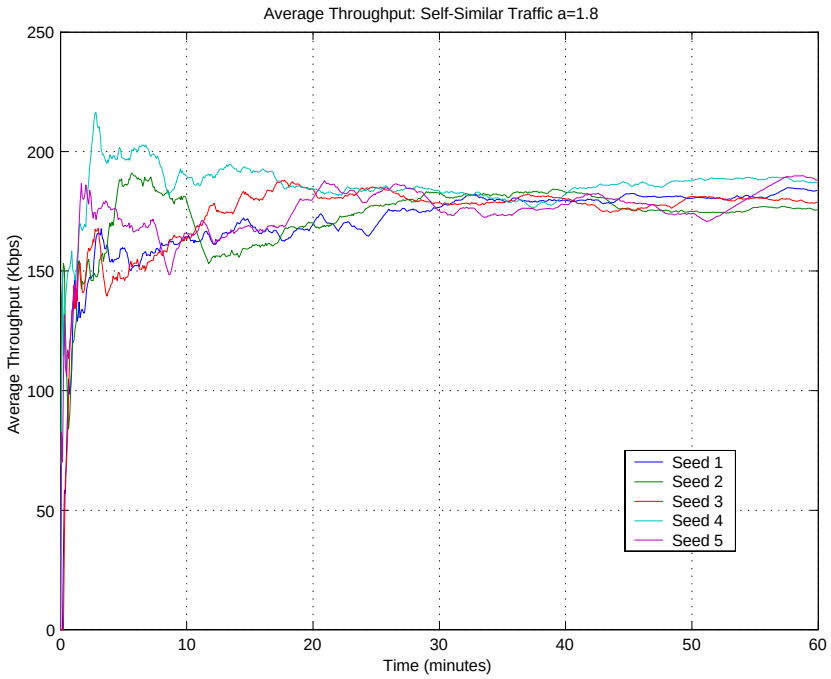

Figure 4.13: Average Throughput for Self-Similar Source α = 1.8

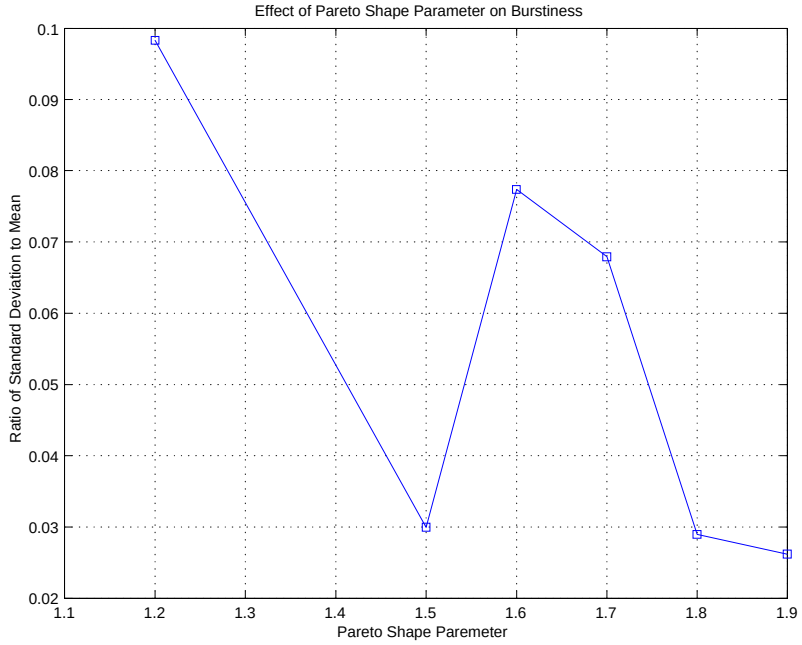

Figure 4.14: Burstiness of Self-Similar Source

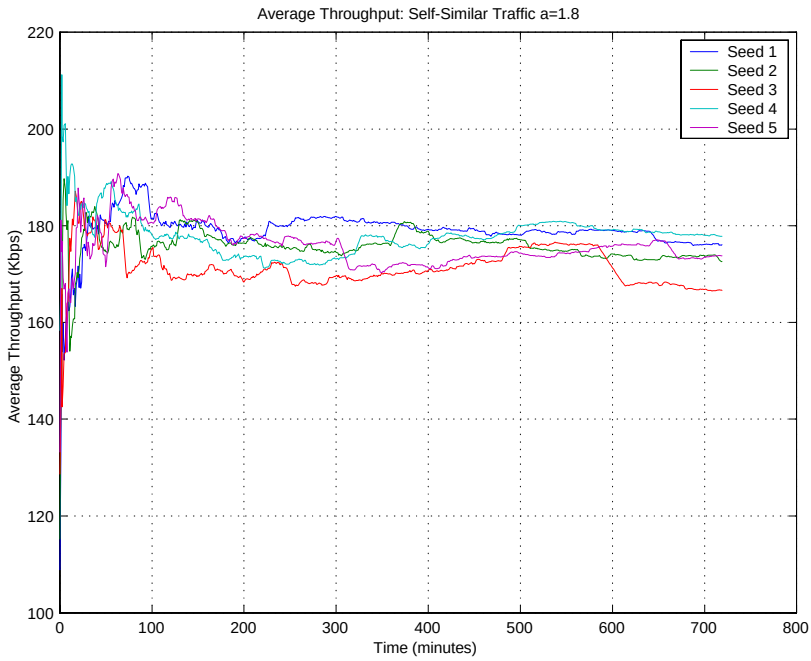

Figure 4.15: Average Traffic Recieved for a 12-hour Simulation Run $\alpha = 1.8$

Internet traffic and generating meaningful simulation results. A shape parameter of $\alpha = 1.2$ most accurately models Internet traffic. As shown in Figure 4.12, a shape parameter of $\alpha = 1.2$ produces simulation results that are highly variable and will not reach steady state in a practical amount of time. A shape parameter of $\alpha = 1.8$ produces self-similar traffic that is not as bursty as Internet traffic. However, as shown in Figure 4.13, the simulation can reach steady state and the variance between runs is much smaller. Although this traffic is not as bursty as actual Internet traffic, a comparison of Figure 4.13 and Figure 4.11 show that it is much burstier than exponential traffic. Based upon these tests, the simulation used a shape parameter of $\alpha = 1.8$.

The simulation was next run for 12 hours with self-similar traffic and a shape parameter of $\alpha = 1.8$. The results of the simulation runs with five seed values are shown in Figure 4.15. The simulation appears to reach steady state after about two hours, with some variance between runs, as expected. Therefore, a simulation run time of three hours was selected for self-similar traffic sources.

4.4 Summary

This chapter has presented the methods used to validate and verify the simulation model. Section 4.1 discussed the use of deterministic input signals to verify correct operation of the simulation. Next, Section 4.2 presented analysis using both deterministic and exponential traffic sources to validate the simulation model. Finally, Section 4.3 gave the result of simulation pilot runs to determine both the effects of seed variables and appropriate simulation run times.

Chapter 5

Results and Analysis

This chapter presents the simulation results and analysis. The chapter begins with a discussion of the simulation results' statistical accuracy in Section 5.1. The next five sections present results and analysis focused on answering the five research questions posed in Section 1.4. First, Section 5.2 presents the baseline scenario results and addresses research Question 1. The statistical multiplexing scenario results and analysis of research Question 2 are given in Section 5.3. Next, Section 5.4 presents the results for the dynamic code assignment scenario and answers research Question 3. Analysis of supporting data traffic with different QoS requirements, including a hybrid scenario, are given in Section 5.5 addressing research Question 4. Finally, Section 5.6 discusses the effect of exponential and self-similar traffic models on the simulation results answering research Question 5. The research questions from Section 1.4 are repeated here for ease of reference:

1. What is the baseline performance that can be expected for a single bursty data source on a single OVSF code?

2. How does the performance of bursty data traffic compare to the baseline performance when multiple bursty sources are statistically multiplexed on a single OVSF code?

(a) At what traffic load does the delay become unacceptable?

(b) Is the statistical multiplexing of bursty data sources suitable for non-real-time data traffic?

(c) Is the statistical multiplexing of bursty data sources suitable for real-time data traffic?

3. How does the performance of bursty data traffic compare to the baseline performance when multiple bursty sources are "over booked" by dynamically assigning OVSF codes to share bandwidth?

   (a) At what traffic load does the delay become unacceptable?

   (b) Is the "over-booking" of bursty data sources suitable for non-real-time data traffic?

   (c) Is the "over-booking" of bursty data sources suitable for real-time data traffic?

4. Should a Radio Link Control (RLC) or Medium Access Control (MAC) algorithm use statistical multiplexing, dynamic code assignment, or a combination of both methods to assign OVSF codes for bursty data traffic with different QoS requirements?

5. How does the use of different traffic models, specifically exponential and self-similar traffic models, affect the performance of a code assignment algorithm using statistical multiplexing or dynamic code assignment?

5.1 Statistical Accuracy

The use of stochastic traffic sources introduces a measure of uncertainty in simulation results. Running a simulation with different random number generator seeds will produce different results. In this research, the simulation model was run with five different seed values for each set of input parameters. The statistical accuracy of the results was measured as

Table 5.1: 95-Percent CI - Baseline Exponential Traffic

Expected Data Rate	**175000**	**350000**	**400000**	**450000**	**470000**
Seed 1	174861	348966	399296	449748	468745
Seed 2	174259	348628	398912	449299	468278
Seed 3	174259	348157	398313	448833	467978
Seed 4	174579	348237	398465	448703	467675
Seed 5	174416	348785	398862	449300	468207
Mean	174475	348555	398465	449177	468177
95% CI	174160	348122	398286	448658	467686
	174789	348998	399253	449656	468667

a confidence interval. The 95-percent confidence interval was calculated with Equation 5.1, where $\overline{x}$ is the mean, s is the standard deviation, n is the number of samples, and $t_{[1-\alpha/2;n-1]}$ is the $(1-\alpha/2)$ -quantile of the Student t distribution with $n-1$ degrees of freedom [7].

$$100(1-\alpha)\%CI = (\overline{x} - t_{[1-\alpha/2;n-1]}\frac{s}{\sqrt{n}}, \overline{x} + t_{[1-\alpha/2;n-1]}\frac{s}{\sqrt{n}} \tag{5.1}$$

The simulation results for exponential traffic sources had very tight confidence intervals. Table 5.1 gives the throughput results for the Baseline Scenario with exponential traffic. As an example, consider the right-most column for a source mean data rate of 470 kbps. The average throughput for all five seeds was 468,177 bps with a 95-percent confidence interval of 467,686 to 468,667 bps.

The simulation results for on-off traffic sources had reasonably tight confidence intervals considering the use of Pareto distributed on and off times. As discussed in Section 2.4, simulation using heavy-tailed distributions causes increased variance at steady state. Table 5.2 gives the throughput results for the Baseline Scenario with on-off traffic. As an example, consider the right-most column for a source peak data rate of 500 kbps. The average throughput for all five seeds was 252,356 bps with a 95-percent confidence interval of 242,600 to 262,112 bps.

The results for all simulation runs exhibited similar 95-percent confidence intervals.

Table 5.2: 95-Percent CI - Baseline On-Off Traffic

Peak Data Rate	**350000**	**400000**	**450000**	**470000**	**480000**	**500000**
Seed 1	177634	199686	222967	245632	248681	249358
Seed 2	176155	196958	219457	255666	247791	241424
Seed 3	170159	202343	238954	250619	232154	261217
Seed 4	172840	193610	216899	235436	246628	251205
Seed 5	178995	204665	215278	236808	252171	258576
Mean	175157	199452	222711	244832	245485	252356
95% CI	170671	194043	210876	234006	235884	242600
	179642	204862	234546	255658	255086	262112

Exponential traffic sources resulted in very tight confidence intervals, while on-off traffic sources produced reasonably tight confidence intervals. Therefore, five simulation runs with different seed values was determined to be sufficient for 95-percent confidence intervals. All data points graphed in the rest of this chapter represent the mean value of five simulation runs with different seed values. Complete tables with the results of individual simulation runs and the 95-percent confidence intervals are given in the Appendix.

5.2 Baseline Scenario Results

The intent of the baseline scenario was to measure the maximum channel utilization for a single traffic source without excessive end-to-end delay. The base station used a single wireless channel with a data rate of 480 kbps. The source data rate was gradually increased until the average end-to-end packet delay increased exponentially. The scenario was run for both exponential and on-off traffic sources. These results serve as a basis of comparison for the Statistical Multiplexing and Dynamic Code Assignment scenarios.

5.2.1 Exponential Traffic

The baseline scenario demonstrated that a single exponential traffic source makes efficient use of a dedicated channel. The single exponential source with a mean data rate

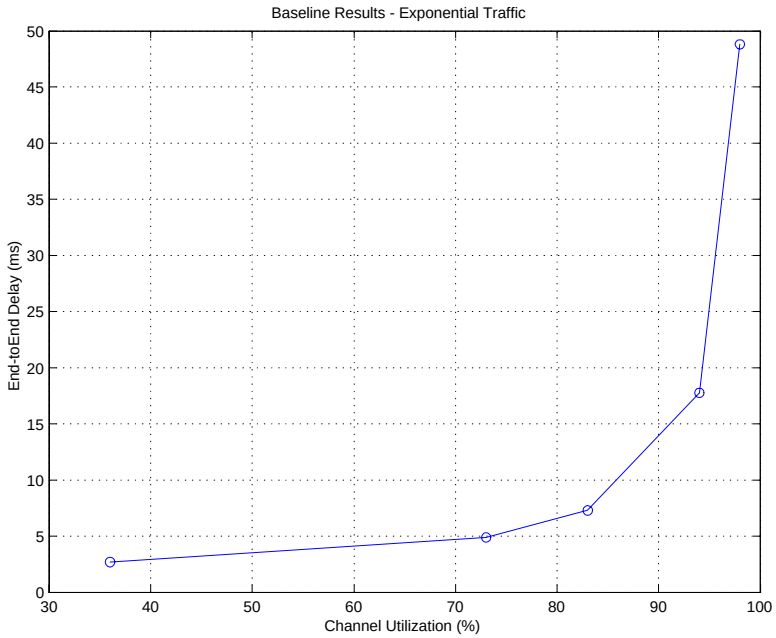

Figure 5.1: Baseline Results Exponential Traffic

of 400 kbps achieved over 80 percent channel utilization with an average end-to-end delay less than 10 ms. The mean data rate of the exponential source was varied between 175 kbps and 470 kbps. Figure 5.1 plots the average end-to-end packet delay as a function of channel utilization. The average end-to-end delay began to increase exponentially when the mean data rate was increased to 450 and 470 kbps.

5.2.2 On-Off Traffic

The baseline scenario demonstrated that a single on-off traffic source makes inefficient use of a dedicated channel. The single on-off traffic source only achieved a channel utilization of approximately 50 percent without excessive delay. The peak data rate of the on-off traffic source was varied between 300 kbps and 500 kbps, and the dwell times in the on and off states were Pareto distributed. Figure 5.2 plots the average end-to-end packet delay as a function of channel utilization. The average end-to-end packet delay was constant, consisting of only transmission delay, up to a peak data rate of 470 kbps. The average end-to-end delay

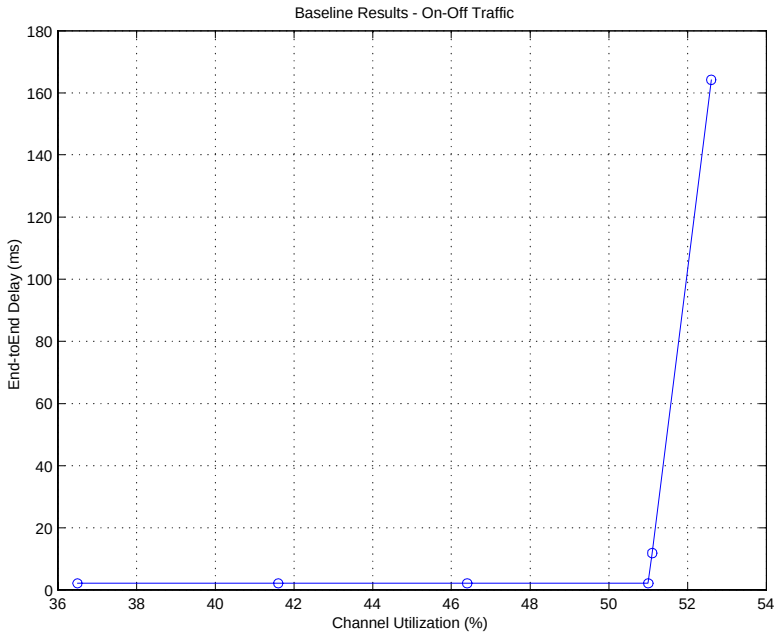

Figure 5.2: Baseline Results On-Off Traffic

began to increase exponentially when the peak data rate was increased to 480 and 500 kbps.

5.2.3 Analysis

The baseline performance for a single bursty data source on a dedicated channel varies significantly for different traffic models. One of the basic premises of this research was that a bursty data source would inefficiently utilize dedicated bandwidth. If the bursty data source was allocated sufficient bandwidth for low delay at peak data rates, it would result in unused bandwidth at mean data rates. The baseline results for on-off traffic illustrated this concept. Figure 5.2 shows that the on-off traffic source has almost no delay with up to 50 percent channel utilization. However, when the channel utilization exceeded 50 percent the end-to-end delay increased exponentially. This result makes sense based upon the on-off traffic model used in the simulation. The on-off source transmitted at a constant data rate in the on state, and did not transmit in the off state. The length of the on and off states followed a Pareto distribution with equal likelihood of being in each state. Therefore, the

mean data rate of the on-off source is one half of the peak data rate. The on-off data source experienced only transmission delay as long as the peak data rate was less than the maximum channel data rate. If the peak data rate exceeded the maximum channel capacity the queue size grew and the on-off source experienced exponentially increasing delay. Consequently, the on-off traffic source achieved only 50 percent channel utilization without experiencing excessive delay.

A single exponential traffic source made relatively efficient use of a dedicated channel in comparison to the on-off source. Figure 5.1 shows that an exponential traffic source utilized 83 percent of the channel bandwidth with and end-to-end delay below 10 ms. As the channel utilization increased above 90 percent, the end-to-end delay increased exponentially. This result is intuitive for the exponential traffic model used in the simulation. The exponential source generated fixed length packets with an exponentially distributed inter-arrival time. With a mean inter-arrival time corresponding to an average data rate of 400 kbps, the channel was able to service bursts of arriving packets without significant queuing delay. However, when the average data rate increased to 450 kbps or higher the bursts of arriving packets caused excessive queuing delay.

The baseline results demonstrated that channel utilization is highly dependent upon the traffic model. It is therefore expected that the improvement in channel utilization resulting from statistical multiplexing and dynamic code sharing will also be dependent upon the traffic model. A detailed discussion of the effect of these traffic models on the statistical multiplexing and dynamic code sharing techniques is given in Section 5.6.

5.3 Statistical Multiplexing Results

The intent of the statistical multiplexing scenario was to measure the maximum channel utilization for a shared channel without excessive delay. The base station used a

single wireless channel with a data rate of 480 kbps. The number of sources sharing the channel was gradually increased. The scenario was run for both exponential and on-off traffic sources.

5.3.1 Exponential Traffic

The statistical multiplexing scenario demonstrated that multiplexing up to ten exponential traffic sources did not result in any improvement in channel utilization. The starting point for this scenario was a single traffic source with a mean data rate of 400 kbps and an average end-to-end delay of approximately 7 ms. This corresponded to the baseline scenario results for maximum channel utilization without excessive delay (Figure 5.1). The number of exponential traffic sources was varied between 1 and 10. In each case, the mean data rate of each source was $400/n$ kbps where n is the number of sources. Figure 5.3 shows the channel utilization as the number of users increases. In each case, the channel utilization was 83% corresponding to an average throughput of 400 kbps. Figure 5.4 plots the average end-to-end packet delay as the number of users increases. Once again the results did not differ significantly from the case of a single exponential traffic source. Finally, Figure 5.5 compares the peak and mean throughput as the number of users is increased. The fact that the peak data rate for ten users was not lower than that for one user indicates that the aggregate traffic flow has not become less bursty. Therefore, statistically multiplexing ten exponential traffic sources did not result in any improvement in channel utilization.

5.3.2 On-Off Traffic

The statistical multiplexing scenario demonstrated that multiplexing on-off traffic sources provided a 26 percent improvement in channel utilization over the baseline scenario. The starting point for this scenario was a single on-off traffic source with a peak data rate of 470 kbps and an average end-to-end delay of approximately 2 ms. This corresponds to the

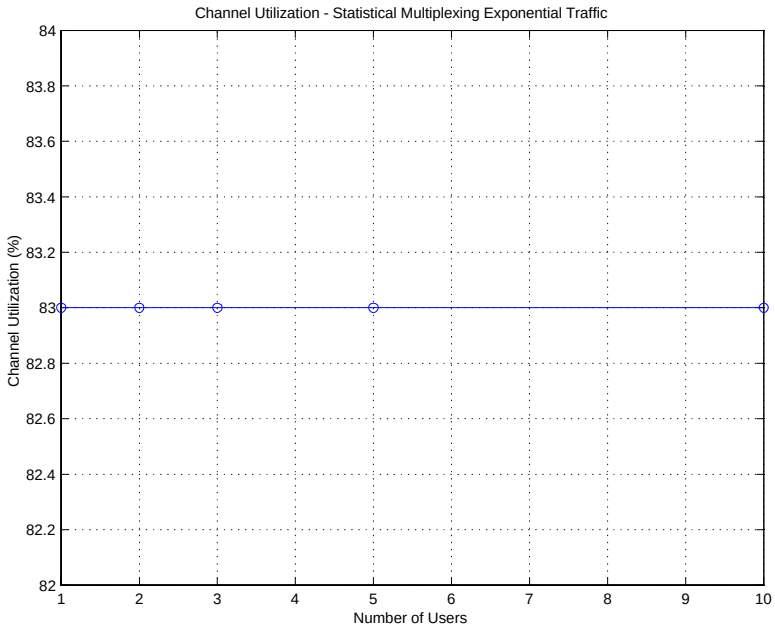

Figure 5.3: Channel Utilization - Statistical Multiplexing of Exponential Traffic

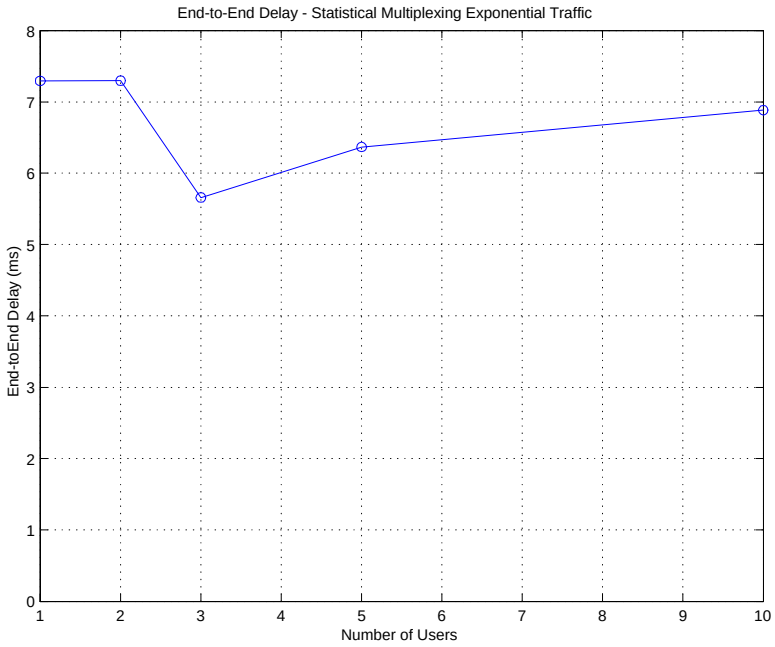

Figure 5.4: End-to-End Delay - Statistical Multiplexing of Exponential Traffic

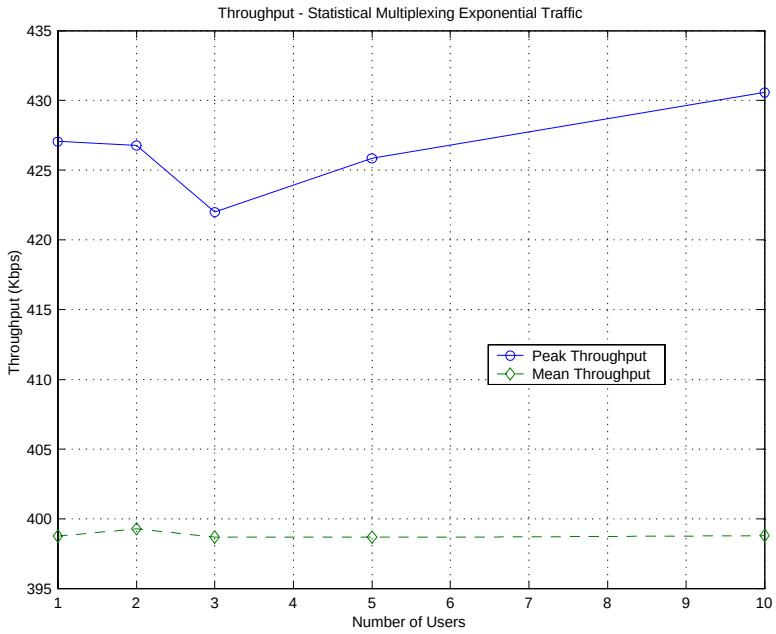

Figure 5.5: Peak vs. Mean Throughput - Statistical Multiplexing of Exponential Traffic

baseline scenario results for maximum channel utilization without excessive delay (Figure 5.2). The number of on-off traffic sources was varied between 1 and 10. In each case, the peak data rate of each source was $470/n$ kbps where n is the number of sources. Figure 5.6 and Figure 5.7 show the channel utilization and average end-to-end delay as the number of users is increased. For both channel utilization and end-to-end delay, the results for 10 users did not differ significantly from the results for one user. Figure 5.8 gives a comparison of peak and mean data rates as the number of users is increased. The mean data rate remained almost constant, but the peak data rate decreased from about 430 kbps to 340 kbps. The aggregate traffic from 10 users was significantly less bursty than the traffic from a single on-off source. Therefore, statistically multiplexing ten on-off sources resulted in the potential for increased channel utilization. To determine how much more data the channel could support, the peak data rate of each user was gradually increased from 47 kbps to 70 kbps. Figure 5.9 shows the average end-to-end delay as channel utilization is increased. The results show that a channel utilization of about 63 percent was achieved with an end-to-end delay less than 10 ms. This corresponds to a peak data rate of 60 kbps for each of the ten

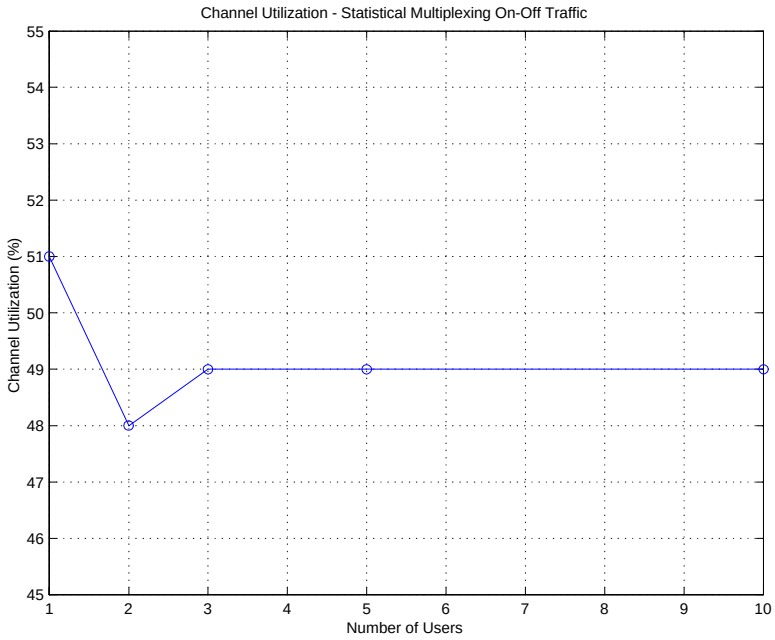

Figure 5.6: Channel Utilization - Statistical Multiplexing of On-Off Traffic

on-off sources. The end-to-end delay began to increase exponentially when the peak data rate was increased to 65 and 70 kbps. Statistical multiplexing of on-off sources gave an improvement in channel utilization of 26 percent in comparison with the baseline case of 50 percent channel utilization.

5.3.3 Analysis

The statistical multiplexing results varied significantly for the different traffic models. Sharing a channel between ten exponential traffic sources resulted in almost exactly the same channel utilization and end-to-end delay as the baseline scenario for a single source. This indicated that an exponential traffic source is not bursty enough to result in any statistical multiplexing gain with up to ten users sharing a link. It would be expected that multiplexing many more exponential sources would result in some smoothing out of the aggregate traffic flow. Since the simulation modeled the coverage area of a single base station, more than ten simultaneous data users on a single forward link were not modeled. It was deemed

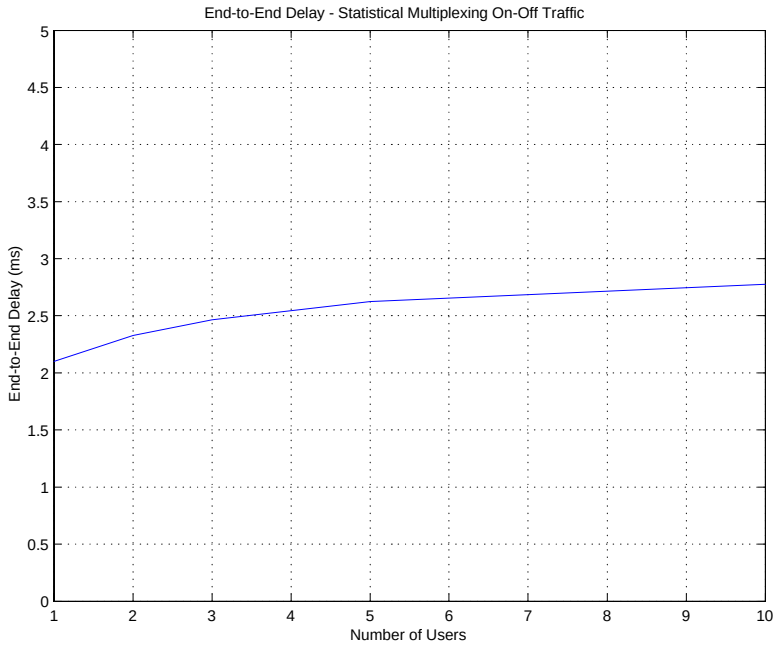

Figure 5.7: End-to-End Delay - Statistical Multiplexing of On-Off Traffic

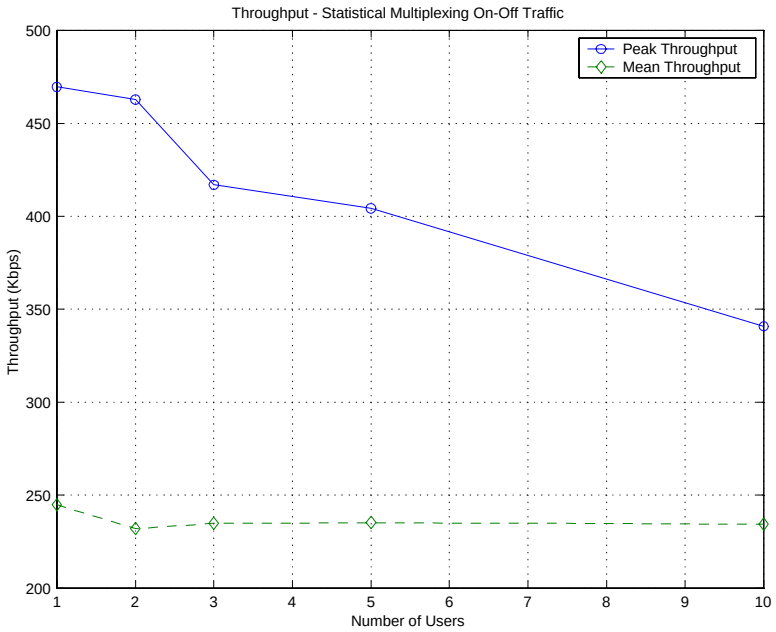

Figure 5.8: Peak vs. Mean Throughput - Statistical Multiplexing of On-Off Traffic

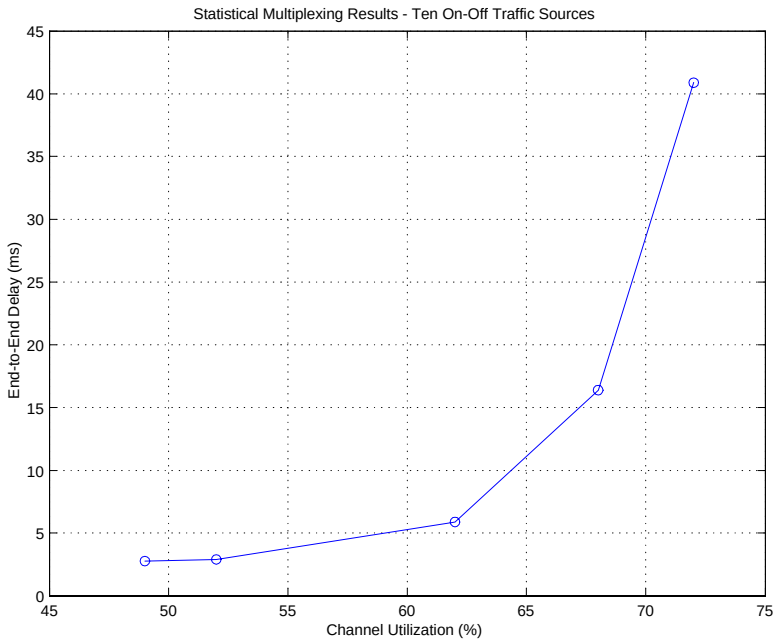

Figure 5.9: Statistical Multiplexing Results - 10 On-Off Traffic Sources

unlikely that more than ten simultaneous data users would be available to share a single channel within a single cell. While multiplexing exponential traffic sources did not result in an increase in channel utilization, this method could still be useful for source data rates that fall between possible channel data rates. For example, the WCDMA standard offers channel data rates of 15, 30, 60, 120, 240, 480, 960 and 1920 kbps. A single source with a mean data rate of 300 kbps would require a 480 kbps channel. Statistical multiplexing could be used to share the 480 kbps channel with other exponential traffic sources resulting in improved channel utilization.

Sharing a channel between on-off sources resulted in significant improvement in channel utilization in comparison with the baseline scenario. Figure 5.9 shows that multiplexing ten on-off users gave a channel utilization of 63 percent with an average end-to-end delay of about 6 ms. This corresponds to a 26 percent increase in channel utilization with only a 4 ms increase in end-to-end delay. Therefore for bursty traffic sources, statistically multiplexing even a small number of traffic flows on a shared channel resulted in much improved channel

utilization.

The statistical multiplexing of a number of sources on a shared channel is suitable for both real-time and non-real-time data traffic. The average end-to-end delay for up to ten users was below 10 ms for both exponential and on-off traffic sources. A benchmark of 200 to 400 ms is commonly used as acceptable end-to-end delay for real-time traffic. Voice traffic, for example, typically requires an end-to-end delay below 250 ms [38]. Therefore, a delay of 10 ms on a single link is reasonable for real-time traffic. Since the delay increased exponentially above certain traffic loads, both a traffic admission policy and traffic policing would be necessary to guarantee QoS. Statistical multiplexing does not reserve bandwidth for an individual traffic flow. It is therefore suitable for QoS schemes such as DiffServ, which offer different QoS levels to aggregate traffic flows. Without traffic admission or policing, statistical multiplexing could still be used for non-real-time traffic.

5.4 Dynamic Code Assignment Results

The intent of the dynamic code assignment scenario was to measure the maximum channel utilization using dynamic access to dedicated channels. The base station used five channels corresponding to a set of OVSF codes. A high QoS user was given priority access to three channels of 480, 240, and 120 kbps each. A medium QoS user had access to an orthogonal 240 kbps channel when it was not blocked. Likewise, a low QoS user had access to an orthogonal 120 kbps channel when it was not blocked. The data rate of the high QoS user was gradually increased. The scenario was run for both exponential and on-off traffic sources.

5.4.1 Exponential Traffic

The dynamic code assignment scenario results for exponential traffic showed significant improvements in channel utilization without causing excessive delay for the high QoS user. The mean data rate of the high QoS user was varied between 100 kbps and 400 kbps. The maximum data rate of 400 kbps was selected because this was the maximum data rate for a single user in the baseline scenario. The medium QoS source transmitted at a mean data rate of 200 kbps, and the low QoS source transmitted at a mean data rate of 100 kbps. Figure 5.10 shows the average throughput and end-to-end delay for high QoS traffic. The results show throughput exactly matching the mean data rate, and end-to-end delay varying between 10 and 30 ms. The high QoS user experiences more delay than in the baseline case, but the values are still acceptable for real-time traffic. Figure 5.11 shows the average throughput and delay for medium QoS traffic. The results show throughput above 100 kbps when the high QoS mean data rate is 200 kbps or lower. The end-to-end delay increases significantly with a high QoS data rate between 100 and 150 kbps, but drops off at higher data rates. The somewhat counter-intuitive decrease in delay at high QoS data rates above 150 kbps is caused because most of the medium QoS packets are never delivered and are still in queue when the simulation ends. Figure 5.12 shows the throughput and delay for the low QoS source. The results show that the low QoS throughput drops below 30 kbps when the high QoS data rate is 150 kbps. Likewise, the low QoS user experiences end-to-end delays over 400 seconds when the high QoS user transmits at 150 kbps or higher. As with the medium QoS user, delay eventually decreases for the low QoS user indicating that most packets are not being delivered by simulation end time.

Figure 5.13 compares the total throughput, high QoS throughput, medium QoS throughput, and low QoS throughput. The results show that the dynamic code assignment algorithm makes efficient use of available bandwidth. The medium and low QoS sources are given access to the channel when the high QoS user transmits below 400 kbps. Finally,

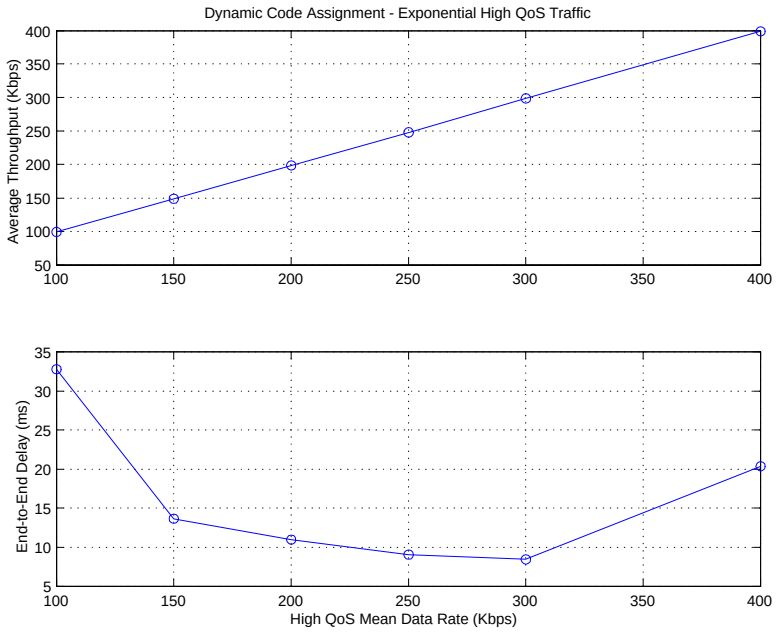

Figure 5.10: Throughput and Delay for Dynamic Code Assignment High QoS Traffic

Figure 5.14 plots the improvement in channel utilization as a function of the high QoS mean data rate.

5.4.2 On-Off Traffic

The dynamic code assignment scenario results for on-off traffic showed significant improvements in channel utilization without causing excessive delay for the high QoS user. The peak data rate of the high QoS user was varied between 100 kbps and 470 kbps. The maximum data rate of 470 kbps was selected because this was the maximum data rate for a single user in the baseline scenario. The medium QoS source transmitted at a peak data rate of 230 kbps, and the low QoS source transmitted at a mean data rate of 110 kbps. Figure 5.15 shows the average throughput and delay results for high QoS traffic. The throughput results are approximately half of the peak data rate, which is consistent for on-off traffic with a 50 percent duty cycle. The end-to-end delay increases to almost 200 ms as the high QoS peak data rate approaches 470 kbps. While this is a significant increase in delay over the

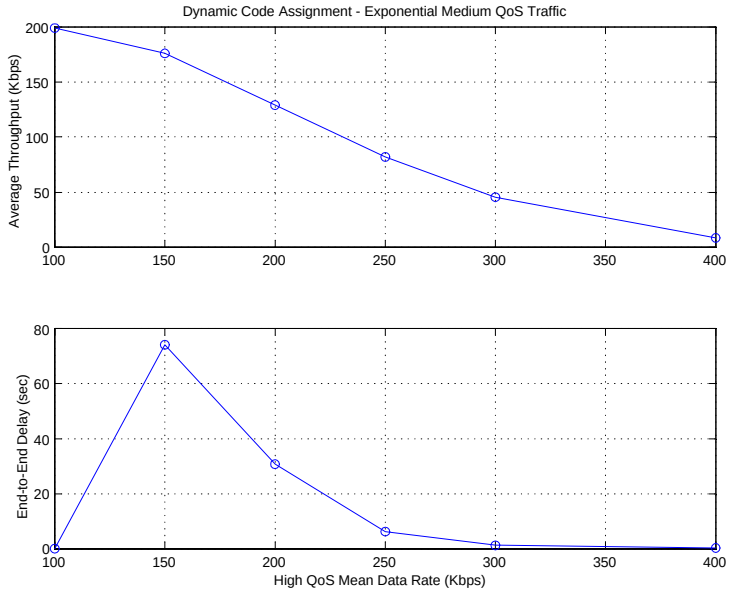

Figure 5.11: Throughput and Delay for Dynamic Code Assignment Medium QoS Traffic

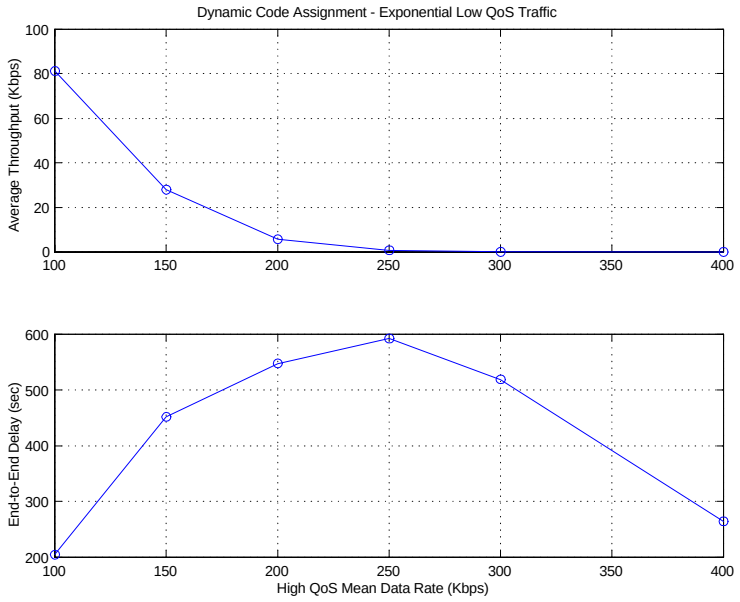

Figure 5.12: Throughput and Delay for Dynamic Code Assignment Low QoS Traffic

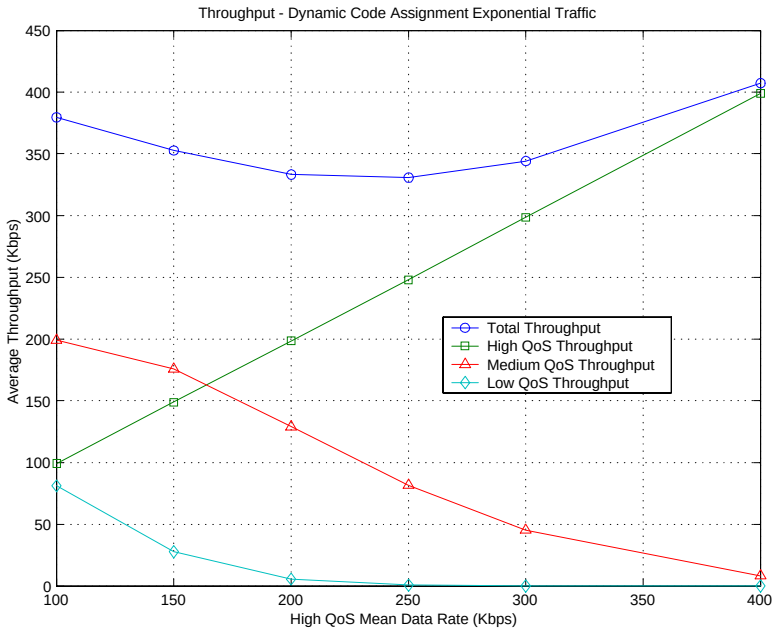

Figure 5.13: Throughput for Dynamic Code Assignment

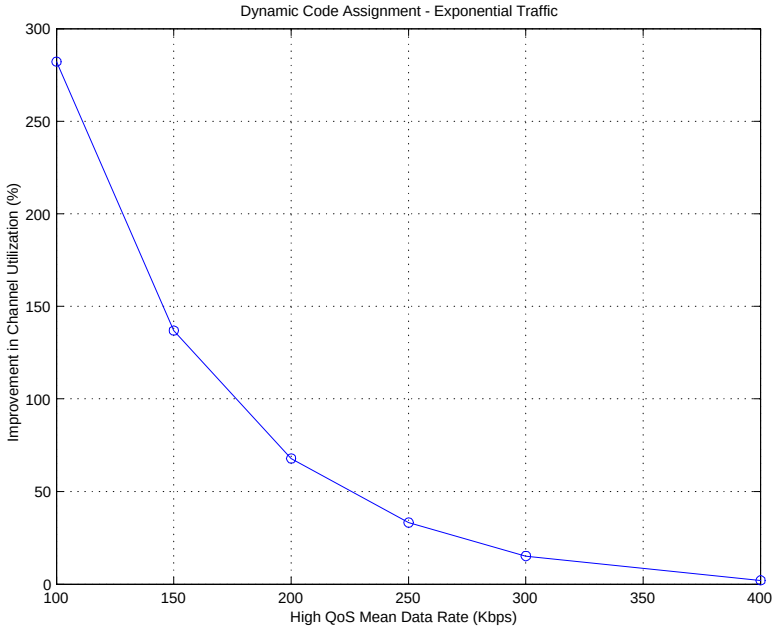

Figure 5.14: Improvement in Channel Utilization for Dynamic Code Assignment Exponential Traffic

baseline scenario, it is still acceptable for real-time data. This is particularly true for high QoS peak data rates of 400 kbps or less when the end-to-end delay is below 50 ms. Figure 5.16 gives the throughput and delay results for medium QoS traffic. The medium QoS source has an average throughput over 100 kbps when the high QoS data rate is 175 kbps or less. The throughput for medium QoS traffic drops off quickly for increased high QoS data rates. Likewise, the end-to-end delay for medium QoS traffic is below 200 ms when the high QoS data rate is 175 kbps or less. For increased high QoS data rates the medium QoS end-to-end delay increases to over 400 seconds. Figure 5.17 shows the throughput and delay results for low QoS traffic. The low QoS source has throughput over 50 kbps and delay below 200 ms when the high QoS source transmits at 100 kbps. For all increased high QoS data rates the low QoS source has throughput below 20 kbps and end-to-end delay above 1000 seconds.

Figure 5.18 compares the total, high QoS, medium QoS, and low QoS throughput. The results show that the dynamic code assignment algorithm makes efficient use of available bandwidth for on-off traffic sources. The medium QoS and low QoS sources are able to utilize available bandwidth at all high QoS source data rates. The improvement in channel utilization is greatest when the high QoS source transmits at low data rates, as illustrated by Figure 5.19.

5.4.3 Analysis

The dynamic code assignment results showed significant improvement in channel utilization without causing excessive delay for either exponential or on-off traffic sources. The dynamic code assignment algorithm was designed to allow medium or low QoS traffic to utilize available bandwidth when high QoS traffic was not transmitting at peak data rates. Figures 5.13 and 5.18 illustrate that the algorithm achieves this goal for both exponential and on-off traffic sources. Dynamic code sharing for exponential traffic resulted in a total throughput between 330 and 400 kbps when the high QoS mean data rate is varied between

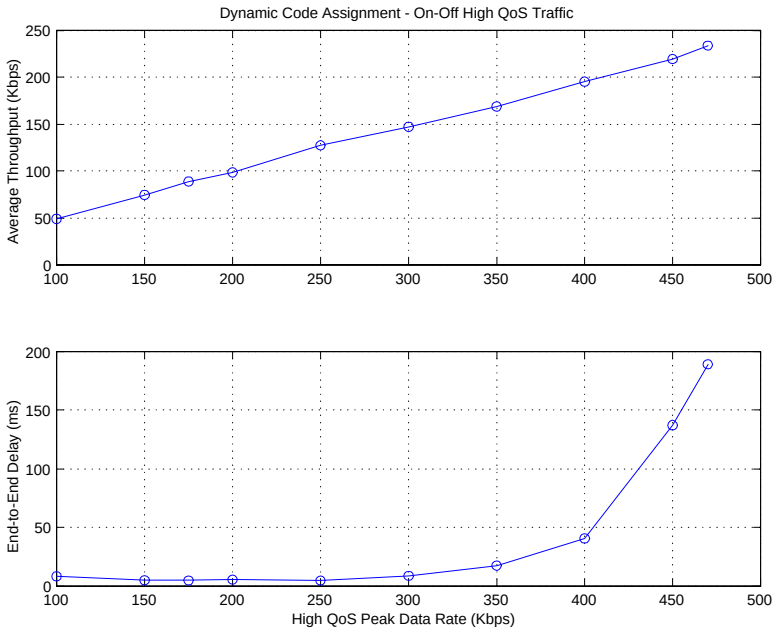

Figure 5.15: Throughput and Delay for Dynamic Code Assignment High Qos On-Off Traffic

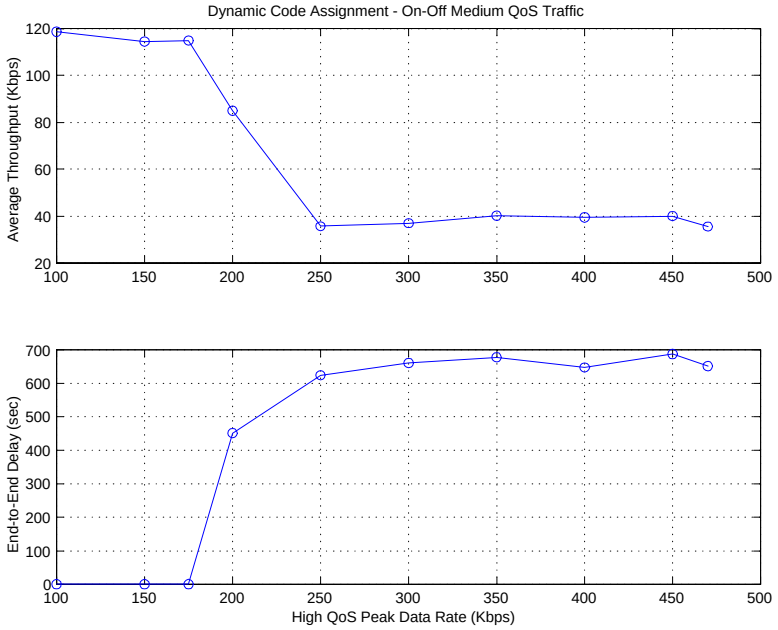

Figure 5.16: Throughput and Delay for Dynamic Code Assignment Medium Qos On-Off Traffic

Figure 5.17: Throughput and Delay for Dynamic Code Assignment Low Qos On-Off Traffic

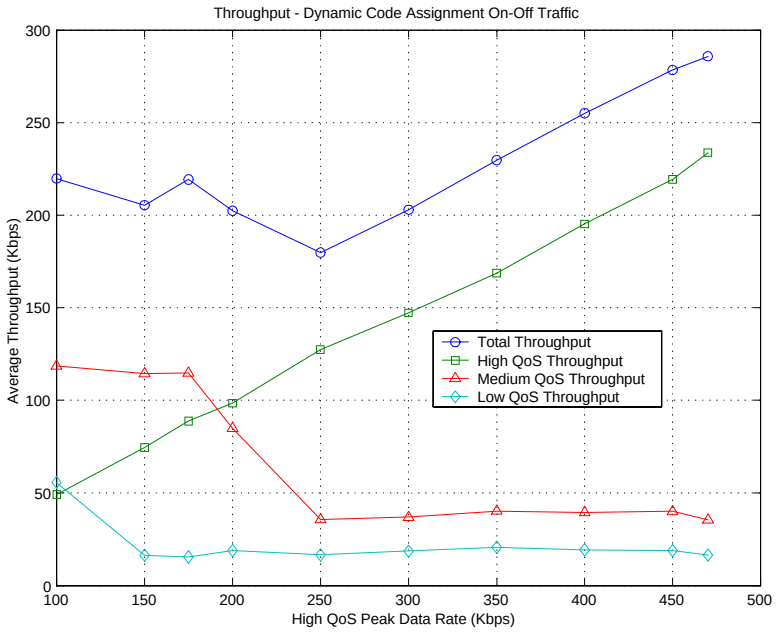

Figure 5.18: Throughput for Dynamic Code Assignment On-Off Traffic

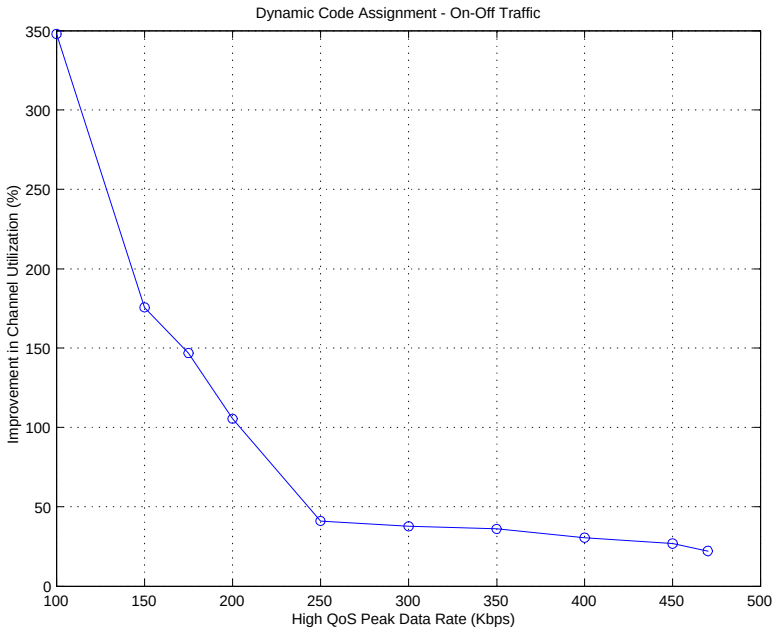

Figure 5.19: Improvement in Channel Utilization for Dynamic Code Assignment On-Off Traffic

100 kbps and 400 kbps. This is achieved with an end-to-end delay below 35 ms for the high QoS user. Similar results were achieved for on-off traffic with a total throughput between 180 and 285 kbps and high QoS delay below 200 ms. In both cases the total channel utilization is comparable with the maximum channel utilization of the baseline scenario. The increases in delay over the baseline scenario come from two sources. First, the transmission delay for each packet is higher when the high QoS source is transmitting on the lower data rate channels. Second, there is some queuing delay caused by the algorithm transitioning between channels. It was necessary to queue packets to prevent simultaneous transmission on non-orthogonal spreading codes.

The dynamic code assignment algorithm offered the most improvement in channel utilization with only two QoS levels for both exponential and on-off traffic sources. Figures 5.12 and 5.17 show that the low QoS source had almost no throughput when the high QoS user transmitted above 150 kbps. However, Figures 5.11 and 5.16 show medium QoS throughput above 100 kbps with high QoS data rates as high as 200 kbps. This can be

explained by the large increments in data rates between the WCDMA spreading codes. Recall that the spreading codes available to the high QoS user supported data rates of 120, 240, and 480 kbps. A high QoS source with a mean data rate of 200 kbps or more would very rarely transmit on the 120 kbps code. Therefore, the medium QoS user would be allowed to transmit on an orthogonal 240 kbps code, while the low QoS user would be blocked on the orthogonal 120 kbps code. Based upon this result it was deemed unnecessary to implement a code tree of greater than three levels.

The dynamic code assignment algorithm is suitable for supporting both real-time and non-real-time traffic. The high QoS source achieved end-to-end delay below 35 ms for exponential traffic and below 200 ms for on-off traffic. Additionally, the end-to-end delay for on-off traffic was below 50 ms for high QoS peak data rates of 400 kbps or less. These delay values are suitable for real-time traffic over a single link. The high QoS user has priority access to the entire channel bandwidth. A traffic admission policy would therefore be necessary to ensure that the system could provide enough bandwidth to guarantee QoS. Traffic policing would not be necessary to guarantee throughput or delay for the high QoS user. Granting the high QoS user priority access to the channel effectively protects this traffic flow from the actions of other traffic flows sharing the channel. The medium QoS user was allowed to transmit only when bandwidth was available. This would be suitable for non-real-time traffic. The dynamic code assignment algorithm can reserve bandwidth for a single traffic flow. It is therefore suitable for QoS schemes such as IntServ that reserve resources on a per-flow basis. The dynamic code assignment algorithm could also be utilized for QoS schemes like DiffServ that reserve bandwidth for aggregate traffic flows. This possibility is examined in detail in Section 5.5.

5.5 Hybrid Scenario Results

The intent of the hybrid scenario was to measure the maximum channel utilization using a combination of both statistical multiplexing and dynamic code assignment. The base station used three channels corresponding to a set of OVSF codes. Ten statistically multiplexed high QoS sources were given priority access to two channels of 960 and 480 kbps each. Ten statistically multiplexed medium QoS sources were given access to an orthogonal 480 kbps channel when it was not blocked. The peak data rates of the ten high QoS sources were varied between 60 and 80 kbps. The ten medium QoS sources transmitted at peak data rates of 40 kbps. All sources in this scenario were on-off traffic sources with Pareto distributed dwell times. As discussed in Section 2.4, the multiplexing of multiple on-off traffic sources is considered a standard method of simulating "self-similar-like" traffic. Therefore, the traffic sources in this scenario can be considered to represent two differentiated QoS levels of aggregated IP traffic flows.

Figure 5.20 shows the throughput and delay results for the ten high QoS sources. The average throughput is between 300 and 400 kbps. This is consistent with ten on-off sources transmitting at peak data rates between 60 and 80 kbps and a 50 percent duty cycle. The end-to-end delay is below 5 ms in all cases, which is comparable to the baseline scenario delay. Therefore, the aggregate flow of ten high QoS sources appears to be uninfluenced by any other data on the channel.

Figure 5.21 shows the throughput and delay results for ten medium QoS sources. The medium QoS average throughput is just below 200 kbps when the high QoS users have a peak data rate of 60 or 70 kbps. This is consistent with ten on-off sources transmitting at peak data rates of 40 kbps and a 50 percent duty cycle. The medium QoS throughput drops to just below 185 kbps when the high QoS peak data rates increase to 80 kbps. The medium QoS delay is below 6 seconds when the high QoS sources transmit at a peak 60 or 70 kbps, but increases to over 200 seconds when the high QoS sources transmit at a peak of 80 kbps.

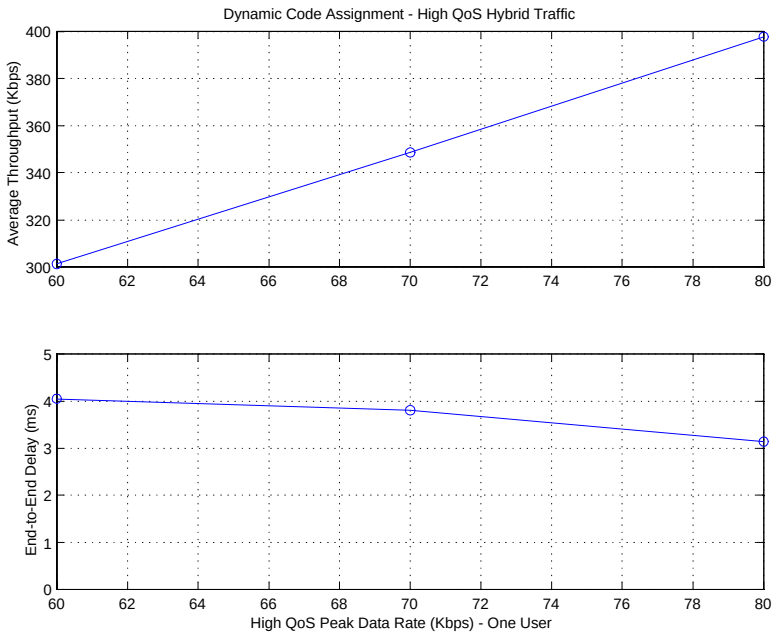

Figure 5.20: Throughput and Delay for Hybrid Scenario High QoS Traffic

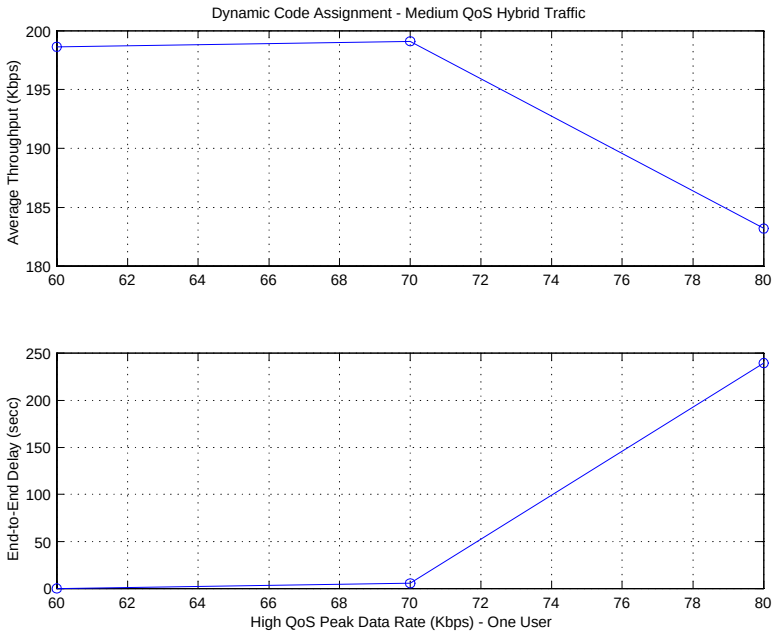

Figure 5.21: Throughput and Delay for Hybrid Scenario Medium QoS Traffic

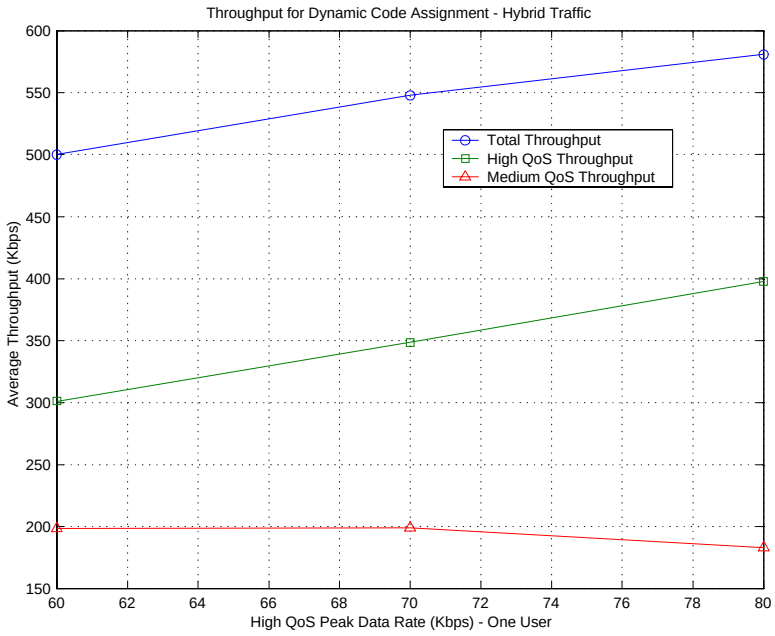

Figure 5.22: Throughput Hybrid Scenario

Figure 5.22 compares the total, high QoS, and medium QoS throughput for the hybrid scenario. The results show that the dynamic code assignment algorithm makes efficient use of available bandwidth for multiplexed traffic sources. The total throughput is between 500 and 575 kbps when the high QoS users transmit at a mean between 300 and 400 kbps. The improvement in channel utilization varies between 45 and 65 percent for the hybrid scenario, as illustrated in Figure 5.23. This improvement is achieved with end-to-end delays less than 5 ms for high QoS sources, and less than 200 ms for medium QoS sources. The results demonstrate that the dynamic code assignment algorithm developed through this research works exceptionally well for aggregated on-off traffic sources, which are representative of self-similar IP traffic.

5.5.1 Analysis

The results from the hybrid scenario demonstrate that the dynamic code assignment algorithm works exceptionally well for aggregated on-off traffic sources. The ten multiplexed

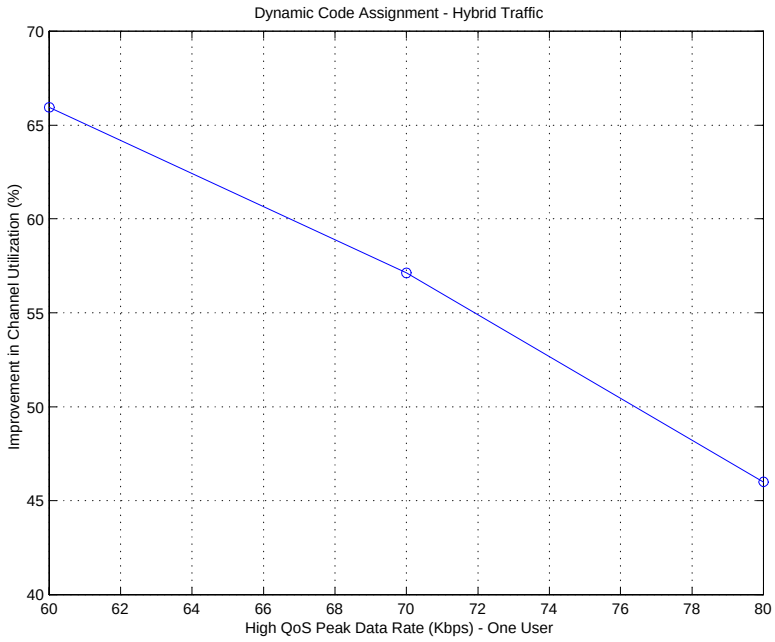

Figure 5.23: Improvement in Channel Utilization for Hybrid Scenario

high QoS users were unaffected by other traffic, and achieved throughput and delay values consistent with the baseline scenario. The QoS for these sources is suitable for the IP Controlled Load (CL) service class, as described in Section 2.3. The CL service class is designed to offer IP traffic performance equivalent to that of a lightly loaded network, regardless of the current network load. The ten multiplexed medium QoS sources achieved throughput no less than 90 percent of the offered load, and average end-to-end delay below 200 seconds. The QoS for the ten medium QoS sources is suitable for the IP Best Effort service class, as described in Section 2.3.

The use of multiplexed on-off traffic sources in the hybrid scenario increases the importance of the results. As discussed in detail in Section 2.4, aggregated on-off traffic sources can be used to simulate self-similar Internet traffic. Therefore, the hybrid scenario results are more representative of how the dynamic code assignment algorithm would function under and actual IP traffic load then the results in the exponential or on-off scenarios. The dynamic code assignment algorithm provided an increase in channel utilization between 45 and 65 percent over that of only high QoS traffic by allowing medium QoS traffic to

transmit on available bandwidth. This was achieved with no increase in high QoS end-to-end delay in comparison to the baseline case of a dedicated channel. Additionally, the total channel utilization increased between 5 percent and 19.6 percent in comparison with the best utilization in the baseline case. The medium QoS traffic was able to transmit at a minimum of 90% of its offered load with average end-to-end delay below 200 seconds.

The hybrid scenario results demonstrate that the dynamic code assignment algorithm is capable of supporting IP QoS traffic. It would require both admission control and traffic policing to guarantee throughput and delay for the high QoS traffic flows. As such, it is suitable for the DiffServ IP QoS standard that provides QoS guarantees to aggregate flows of traffic with the same QoS level. The algorithm could effectively support two IP QoS levels corresponding to the CL service class and the best effort service class.

5.6 Traffic Model Analysis

The performance of both statistical multiplexing and dynamic code assignment is significantly impacted by the traffic model used in the simulation. The different simulation scenarios used exponential, on-off, and "self-similar like" traffic models. The variance in results indicates the importance of selecting a traffic model that accurately represents the actual network load. This research effort was intended to develop an algorithm to make efficient use of a wireless channel under a load of IP traffic with different QoS requirements.

The exponential traffic model is widely used in both queuing theory and simulation to represent bursty data traffic. However, the resulting traffic is relatively smooth and well-behaved in comparison to an on-off traffic source. An on-off traffic model is often used to model an individual real-time data source such as a voice-over IP or video stream. The difference between these two traffic models is exemplified by the baseline scenario results, where exponential traffic had 83 percent channel utilization and on-off traffic had 50 percent

channel utilization.

The statistical multiplexing of up to ten users on a single channel performed much better for on-off traffic than for exponential traffic. Ten exponential traffic sources behaved almost identically to a single higher data rate exponential source. However, ten on-off sources produced significant multiplexing gain that resulted in 26 percent increase in channel utilization. As discussed in Section 2.4, aggregating a number of on-off traffic sources results in self-similar like traffic that can be used to model Internet traffic. The maximum channel utilization without excessive delay for statistically multiplexed exponential traffic was 83 percent, while that for multiplexed on-off traffic was 63 percent. This is consistent with measured results indicating that Internet traffic is much burstier than exponential traffic models, and therefore requires larger buffer or bandwidth allocation for similar performance [33].

The dynamic code assignment algorithm offered the best performance for multiplexed on-off traffic sources. This is illustrated by the hybrid scenario results in which the throughput and delay of high QoS sources was not affected by other traffic sharing the channel. The multiplexed on-off traffic sources are representative of aggregate IP traffic flows with two different QoS levels. The dynamic code assignment algorithm also allowed lower QoS traffic to utilize available bandwidth using exponential and on-off traffic sources. However, the high QoS traffic experienced increases in end-to-end delay ranging between about 30 ms and 195 ms.

5.7 Summary

This chapter has presented the simulation results and analysis. The results demonstrated that exponential traffic sources are able to make efficient use of both dedicated and shared channels. On the contrary, on-off traffic sources make very inefficient use of dedi-

cated channels. The statistical multiplexing and dynamic code assignment scenarios both improved the channel utilization for on-off traffic sources. The hybrid scenario multiplexed on-off sources and used dynamic code assignment. The traffic in the hybrid scenario is the most representative of IP traffic. The results for the hybrid scenario demonstrated that the dynamic code assignment algorithm developed through this research makes efficient use of available bandwidth. It is also provides QoS differentiation that is suitable for IP DiffServ traffic.

Chapter 6

Conclusions

The telecommunications industry in North America underwent significant growth in the 1990's for both data communications and wireless voice communications. As a result, both Internet access and cellular telephones have become common household commodities. Attempts to integrate voice and data communications have revealed weaknesses in both the Internet and the cellular telephone network. The Internet's best effort service model can not support real-time traffic such as voice or video. Likewise, the cellular telephone network's low data rates are not suitable for applications such as web browsing or FTP.

Research is ongoing to improve both the cellular telephone network and the Internet. Emerging 3G wireless networks will focus on supporting data traffic with increased data rates. IP QoS schemes such as DiffServ are being implemented to support real-time traffic on the Internet. Our research effort presented in this document focused on supporting IP traffic with different QoS requirements in emerging 3G wireless networks.

6.1 Significant Results

The key contribution of this research was the implementation and testing of two code sharing techniques which are not implemented in existing OVSF code assignment algorithms [1][2][3]. The new OVSF code sharing techniques investigated in this research, termed statistical multiplexing and dynamic code sharing, were described in Section 3.2. The performance of these new code sharing techniques was measured in terms of channel utilization and end-to-end delay.

Simulation was the primary tool used to implement and test the statistical multiplexing and dynamic code sharing techniques. The simulation was validated analytically using both deterministic and exponential traffic sources, as described in Section 4.2. The simulation was run under different scenarios, as described in Section 3.8, which addressed the five research questions posed in Section 1.4. A summary of the significant results for each of the five research questions follows.

6.1.1 Baseline Results

The baseline scenario measured the maximum channel utilization without excessive delay for a single source on a dedicated channel. These results addressed Research Question 1. One of the basic premises of this research was that a bursty data source would inefficiently utilize dedicated bandwidth. In order to meet QoS requirements of both data rate and delay, a bursty data source must be allocated enough bandwidth to support its peak data rate. However, when the bursty source transmits below the peak data rate, the dedicated bandwidth is not utilized. The purpose of this scenario was to measure the maximum channel utilization for a single bursty source using both exponential and on-off traffic distributions. This value served as a benchmark against which the improvement in channel utilization from code sharing could be measured.

The baseline performance for a single bursty data source on a dedicated channel varied significantly for different traffic models. The baseline scenario demonstrated that a single on-off traffic source made inefficient use of a dedicated channel. The single on-off traffic source achieved a channel utilization of only 50 percent without excessive delay. A single exponential traffic source made relatively efficient use of a dedicated channel in comparison to the on-off source. The single exponential source with a mean data rate of 400 kbps achieved 83 percent channel utilization with an average end-to-end delay less than 10 ms. The baseline results demonstrated that channel utilization is highly dependent upon the traffic model.

6.1.2 Statistical Multiplexing

The statistical multiplexing scenario measured the improvement in channel utilization offered by sharing a single code channel between a number of bursty data sources. These results addressed Research Question 2. The basic idea of this scenario was that multiplexing a number of bursty sources would result in an aggregate traffic flow that was less bursty than the individual traffic flows. Statistical multiplexing would smooth out the aggregate traffic flow, resulting in a lower ratio of peak to mean data rate. In terms of channel utilization, a higher mean data rate with the same peak data rate would result in increased channel utilization. This was tested via simulation for both exponential and on-off traffic sources.

The statistical multiplexing results varied greatly for the different traffic models. The statistical multiplexing of up to ten exponential traffic sources did not result in any improvement in channel utilization. In the baseline scenario, a single exponential traffic source achieved 83-percent channel utilization with an average end-to-end delay of approximately 7 ms. The statistical multiplexing of ten exponential traffic sources gave 83 percent channel utilization with end-to-end delay below 10 ms. In addition, the ratio of peak to mean data rate did not vary significantly for up to ten multiplexed exponential sources. Therefore,

statistically multiplexing ten exponential traffic sources did not result in any improvement in channel utilization. In contrast, the statistical multiplexing of ten on-off traffic sources provided a 26 percent improvement in channel utilization over the baseline scenario. In the baseline scenario, a single on-off traffic source achieved a channel utilization of 50-percent and an average end-to-end delay of approximately 2 ms. The statistical multiplexing of ten on-off sources resulted in a channel utilization of 63-percent with an average end-to-end delay of about 6 ms.

6.1.3 Dynamic Code Sharing

The dynamic code sharing scenario measured the improvement in channel utilization offered by dynamically sharing access to a set of OVSF spreading codes for a number of bursty users with different QoS requirements. These results addressed Research Question 3. The basic concept of this scenario was to allocate a set of OVSF codes to a bursty data source with high QoS requirements. As the data rate of this source varied, the source was dynamically switched between codes in the assigned set with different spreading factors. Channel utilization was improved by allowing lower QoS data source to transmit on orthogonal codes in the set when the high QoS source was not transmitting at the peak data rate. This was tested via simulation for both exponential and on-off traffic sources.

The dynamic code assignment results showed significant improvement in channel utilization without causing excessive delay for either exponential or on-off traffic sources. For each traffic model, the simulation was first run with the high QoS source transmitting at the data rate that achieved the maximum channel utilization in the baseline scenario. This corresponded to 83-percent channel utilization for exponential traffic and 50-percent channel utilization for on-off traffic. The data rate of the high QoS source was decreased on successive simulation runs to test the ability of the dynamic code assignment algorithm to utilize available bandwidth. Dynamic code sharing for exponential traffic resulted in a total

channel utilization between 68-percent and 83-percent when the high QoS channel utilization was varied between 20-percent and 83-percent. This was achieved with an end-to-end delay below 35 ms for the high QoS user. Therefore, dynamic code assignment for exponential traffic was able to match the baseline channel utilization but not exceed it.

The dynamic code sharing results for on-off traffic demonstrated an improvement in channel utilization of 18-percent in comparison with the baseline scenario results of 50-percent channel utilization. The total channel utilization varied between 37-percent and 59-percent when the high QoS channel utilization was varied between 20-percent and 50-percent. This was achieved with a high QoS end-to-end delay below 200 ms. The increases in delay over the baseline scenario come from two sources. First, the transmission delay for each packet is higher when the high QoS source is transmitting on the lower data rate channels. Second, there is some queuing delay caused by the algorithm transitioning between channels. It was necessary to queue packets to prevent simultaneous transmission on non-orthogonal spreading codes.

6.1.4 Hybrid Scenario

The hybrid scenario measured the improvement in channel utilization offered by combining statistical multiplexing and dynamic code sharing. These results addressed Research Question 4. The basic idea of this scenario was to use the dynamic code sharing algorithm on aggregate flows of traffic with similar QoS requirements, as opposed to the individual traffic flows used in the dynamic code sharing scenario. The statistical multiplexing of a number of on-off traffic sources with pareto distributed on and off cycles is a widely used method of simulating self-similar Internet traffic. Therefore, the results from this scenario could be used to predict performance of the dynamic code sharing algorithm under a load of IP traffic with different QoS constraints.

The hybrid scenario resulted in an improvement in channel utilization of 18-percent in

comparison to the baseline scenario of 50-percent channel utilization with almost no increase in end-to-end delay for the high QoS traffic. The total channel utilization varied between 52-percent and 59-percent when the high QoS channel utilization was varied between 31-percent and 41-percent. This was achieved with a high QoS average end-to-end delay less than 4 ms. In addition, the low QoS end-to-end delay was less than 200 ms. The results of this scenario demonstrated that the combination of statistical multiplexing and dynamic code sharing provided the best results in terms of both improved channel utilization and low end-to-end delay.

6.1.5 Traffic Models

The use of different traffic models had a significant impact on the results in each simulation scenario. This issue was addressed by Research Question 5. One of the basic premises of this research effort was that a bursty data source would make inefficient use of dedicated bandwidth that supported its peak data rate. The baseline scenario results demonstrated that this was true for the on-off traffic model which achieved only 50-percent channel utilization. However, the single exponential traffic source was able to utilize 83-percent of the channel bandwidth in the baseline scenario. This indicated that the exponential traffic source was significantly less bursty than the on-off source. Since the statistical multiplexing and dynamic code sharing techniques were intended to take advantage of the bursty nature of data traffic, the "less-bursty" nature of the exponential traffic source affected the results in those scenarios. The maximum channel utilization for exponential traffic in both the statistical multiplexing and dynamic code sharing scenarios was 83-percent, which is the same as the baseline scenario.

The on-off traffic source produced improvements in channel utilization between 18-percent and 26-percent in comparison to the baseline scenario for the statistical multiplexing, dynamic code sharing, and hybrid scenarios. These results were encouraging with respect to

utilizing the code sharing techniques developed in this research to support Internet traffic. As discussed in Section 2.4, several papers have been published documenting the fact that Internet traffic is better modeled as self-similar traffic than as exponential traffic. Section 2.4 also described the use of multiple on-off traffic sources to simulate self-similar traffic. The hybrid scenario in particular offered the best representation of actual Internet traffic. In this scenario, the dynamic code sharing technique resulted in 18-percent improvement in channel utilization in comparison with the baseline scenario. This was achieved with a high QoS average end-to-end delay below 4 ms.

6.2 Applications of Code Sharing

The focus of this research effort was to develop and implement a code sharing algorithm that supported IP traffic with different QoS requirements in a 3G wireless network. The key contributions of this work were the implementation of two code sharing techniques, termed statistical multiplexing and dynamic code sharing, that are not utilized in existing OVSF code assignment algorithms. The results of this research have demonstrated that both statistical multiplexing and dynamic code sharing offer improvements in channel utilization for bursty data traffic in a 3G wireless network. The implementation of a code sharing algorithm has several potential applications in 3G and future generations of wireless networks. These include integrating wireless voice and data, supporting IP QoS, and providing "always-on" data service. Each of these potential applications is discussed below.

6.2.1 Integrating Wireless Voice and Data

The current 2G cellular networks are primarily voice networks. This is particularly true in terms of revenue generated by voice versus data traffic. Voice is the "killer-application" for 2G wireless networks and generates almost all of the revenue. The move

to 3G and future generations of wireless networks will focus on supporting wireless data traffic. However, it is not yet clear how supporting wireless data will translate into increased revenues for wireless service providers. This problem is similar to that currently faced by wireline telephone service providers with dial-up Internet access. Dial-up Internet access has significantly increased the call holding times and traffic on wired voice networks. The telephone networks must support this increased traffic load, but the users access fees are paid to the Internet Service Providers (ISP). A similar problem is likely to occur for wireless voice service providers as wireless Internet access increases. The use of code sharing for data traffic in 3G wireless networks offers a potential solution to this issue.

The codes in an OVSF tree can be used to support either voice or data users in a WCDMA network. There will be economic incentive for service providers to use as many of the available codes as possible to support "paying" voice subscribers. Code sharing could be used to allow a number of wireless data users to share a small subset of the available codes. Therefore, while there may be a large number of total codes available in a cell there will still be incentive to make efficient use of a small number of codes to support data traffic.

6.2.2 Supporting IP QoS

The current Internet service model supports only best-effort traffic. Therefore, as 3G wireless networks are deployed there may initially be little incentive to support different QoS levels for data traffic. The statistical multiplexing of data sources to share a single wireless channel may be the most appropriate method of code sharing in this scenario. However, as both IP QoS and the 3G wireless network develop there will be increasing requirements to support data traffic with different QoS requirements.

IP QoS is not currently implemented across large areas of the Internet, but works well in small networks. For example, DiffServ can be used within an office to support both Voice Over IP (VOIP) and data traffic over an Ethernet LAN. As data traffic increases on

3G wireless networks it is likely that the network between the Base Station Controller (BSC) and the Node Base station (Node B) will evolve to a mesh IP network. This will allow ISPs to provide data services at various access points without all data traffic flowing through the BSC. There will be some incentive to go to an all-IP access network using VOIP and DiffServ to service voice customers. This will be similar to what is currently done in wired networks to support voice and data over Ethernet using VOIP and DiffServ.

The Dynamic Code Sharing algorithm developed in this research is capable of supporting multiple QoS levels for IP traffic in a 3G wireless network. This type of code sharing could be used to support VOIP and other real-time IP traffic in 3G and future generations of wireless networks.

6.2.3 Providing "Always-on" Data Service

Providing "always-on" data service in a wireless network presents some interesting challenges. It is not practical to dedicate a channel to each data user that is available whether or not he is currently transmitting. However, the user should have the impression that a channel is always available when he attempts to transmit data. Code sharing offers a potential solution for "always-on" data service. Both the statistical multiplexing and dynamic code sharing techniques are capable of supporting a large number of users on a small number of spreading codes. By sharing access to either a single code or a set of codes, data users will have the impression of "always-on" data service.

6.3 Recommendations for Future Research

This research effort has successfully developed a dynamic code assignment algorithm capable of supporting IP QoS in a 3G wireless network. There are several aspects of this research, which offer interesting prospects for future research. These include, but are not lim-

ited to, interaction with higher layer protocols, multi-code transmission, modelling signalling functions, and modelling multiple cells.

6.3.1 Interaction with higher layer protocols

The results of this research effort could be expanded by integrating the dynamic code assignment RLC/MAC algorithm with higher layer protocols and introducing bit errors on the wireless link. Higher layer network protocols, such as TCP, often experience difficulties transmitting over wireless links. This is due in large part to the fact that TCP considers packet loss to be the result of congestion and not link errors. As a result, TCP reduces its transmission window size when packets are lost due to link errors. The performance of the dynamic code assignment algorithm using both TCP and UDP traffic could be studied.

6.3.2 Multi-code transmission

This research effort could be expanded by considering CDMA systems that support multi-code transmission. While the dynamic code assignment algorithm was designed with WCDMA in mind, it is general enough to support other CDMA systems. A multi-code CDMA system supports higher data rates by assigning each user multiple spreading codes of the same length. This results in smaller steps between possible data rates in the system. The dynamic code assignment algorithm could be adapted to support a high QoS user with priority access to multiple spreading codes. Lower QoS users would be allowed to transmit on unused codes when the high QoS user is not transmitting at peak data rates. This would allow the algorithm to be used in systems such as CDMA 2000, which plan to support higher data rates with multiple spreading codes.

6.3.3 Signalling

The results of this research could be expanded by studying the effects of signalling on performance of the dynamic code assignment algorithm. The signalling required for call set-up and changing spreading codes requires some amount of time. This could potentially reduce the effectiveness of the code assignment algorithm if spreading codes are changed too often. This type of research could investigate the optimal thresholds to trigger changing between spreading codes for the dynamic code assignment algorithm. Research in the area of signalling could also investigate the effects of different call admission or traffic policing policies on the performance of the dynamic code assignment algorithm.

6.3.4 Multiple cells

This research effort could be expanded by modelling multiple cells in the 3G wireless network. Implementing IP QoS across multiple cells poses some interesting problems. The cellular system performs hand-off of the radio link when a mobile user moves between cells. However, a scheme such as Mobile IP is also necessary to handle mobile data terminals moving between IP domains. It is not clear if a protocol such as Mobile IP can offer suitable end-to-end delay for real-time traffic.

6.4 Summary

This chapter has presented conclusions based upon the research results, and recommended areas of future research. The goal of this research was to implement an OVSF code assignment algorithm capable of supporting IP QoS in a 3G wireless network. The results have demonstrated that the research goal was achieved. The dynamic code assignment algorithm is capable of supporting IP QoS on both a per flow (IntServ) and an aggregate flow (DiffServ) basis. It can support two QoS levels corresponding to IP controlled load and

best effort service levels. This type of algorithm does not need to be deployed in every 3G base station. It can be implemented to support demand where wired data networks with IP QoS connect to wireless networks. The results of this research effort could be expanded by implementing higher layer protocols, multi-code transmission, signalling, or multiple cells.

References

[1] Ray-Guang Cheng and Phone Lin, "OVSF code channel assignment for IMT-2000," *IEEE Vehicular Technology Conference*, vol. 3, pp. 2188–2192, 2000.

[2] Thit Minn and Kai-Yeung Siu, "Dynamic assignment of orthogonal variable spreading factor codes in w-CDMA," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 8, pp. 1429–1440, August 2000.

[3] Romano Fantacci and Saverio Nannicini, "Multiple access protocol for integration of variable bit rate multimedia traffic in UMTS/IMT-2000 based on wideband CDMA," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 8, pp. 1441–1454, August 2000.

[4] F. Adachi, M. Sawahashi, and K. Okawa, "Tree-structured generation of orthogonal spreading codes with different lengths for the forward link of DS-CDMA mobile radio," *IEE Electronics Letters*, vol. 33, no. 1, pp. 27–28, January 1997.

[5] "Spreading and Modulation," 3GPP 3rd Generation Technical Specification 25.213 (Release 1999).

[6] "Radio Resource Management Strategies," 3GPP 3rd Generation Technical Specification 25.922 (Release 1999).

[7] Raj Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*, Wiley, New York, 1991.

[8] Roger L. Peterson, Rodger E. Ziemer, and David E. Borth, *Introduction to Spread Spectrum Communications*, Prentice Hall, Upper Saddle River, NJ, 1995.

[9] *IEEE Transactions on Communications*, vol. COM-25, August 1977, Special Issue on Spread Spectrum.

[10] Tero Ojanpera and Ramjee Prasad, Eds., *Wideband CDMA for Third Generation Mobile Communications*, Artech House, Boston, MA, 1998.

[11] Bernard Sklar, *Digital Communications Fundamentals and Applications*, Prentice Hall, New Jersey, 1988.

[12] Esmael H. Dinan and Bijan Jabbari, "Spreading Codes for Direct Sequence CDMA and Wideband CDMA Cellular Networks," *IEEE Communications Magazine*, vol. 36, no. 9, pp. 48–54, September 1998.

[13] Theodore S. Rappaport, *Wireless Communications, Principles and Practice*, Prentice Hall PTR, Upper Saddle River, NJ, 1996.

[14] George I. Zysman, Joseph A. Tarallo, Richard E. Howard, John Freidenfelds, Reinaldo A. Valenzuela, and Paul M. Mankiewich, "Technology evolution for mobile and personal communications," *Bell Labs Technical Journal*, vol. 5, no. 1, pp. 107–129, January-March 2000.

[15] Jian Cai and David J. Goodman, "General packet radio service in GSM," *IEEE Communications Magazine*, vol. 35, no. 10, pp. 122–131, October 1997.

[16] Prodip Chaudhury, Werner Mohr, and Seizo Onoe, "The 3GPP Proposal for IMT-2000," *IEEE Communications Magazine*, vol. 37, no. 12, pp. 72–81, December 1999.

[17] Tero Ojanperä and Ramjee Prasad, "An Overview of Third-Generation Wireless Personal Communications: A European Perspective," *IEEE Personal Communications Magazine*, vol. 5., no. 6, pp. 59–65, December 1998.

[18] Ken Buchanan, Rodger Fudge, David McFarlane, Tim Phillips, Akio Sasaki, and Howard Xia, "IMT-2000: Service Provider's Perspective," *IEEE Personal Communications Magazine*, vol. 4, no. 4, pp. 8–13, August 1997.

[19] Richard D. Carsello and Reuven Meidan, "IMT-2000 Standards: Radio Aspects," *IEEE Personal Communications Magazine*, vol. 4, no. 4, pp. 30–40, August 1997.

[20] Raj Pandya, David Grillo, Edgar Lycksell, Phillipe Mieybegue, Hideo Okinaka, and Masami Yabusaki, "IMT-2000 Standards: Network Aspects," *IEEE Personal Communications Magazine*, vol. 4, no. 4, pp. 20–29, August 1997.

[21] Erik Dahlman, Bjorn Gudmunson, Mats Nilsson, and Johan Skold, "UMTS/IMT-2000 Based on Wideband CDMA," *IEEE Communications Magazine*, vol. 36, no. 9, pp. 70–80, September 1998.

[22] Tero Ojanperä and Ramjee Prasad, "An Overview of Air Interface Multiple Access for IMT-2000/UMTS," *IEEE Communications Magazine*, vol. 36, no. 9, pp. 82–95, September 1998.

[23] "Services Provided by the Physical Layer," 3rd Generation Technical Specification 25.302 (Release 1999).

[24] "Radio Interface Protocol Architecture," 3rd Generation Technical Specification 25.301 (Release 1999).

[25] J. Wroclawski, "Specification of the controlled-load network element service," RFC 2211, September 1997.

[26] S. Schenker, C. Partridge, and R. Guerin, "Specification of guaranteed quality of service," RFC 2212, September 1997.

[27] R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: An Overview," RFC 1633, June 1994.

[28] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services," RFC 2475, December 1998.

[29] Constantinos Dovrolis and Parameswaran Ramanathan, "A Case for Relative Differentiated Services and the Proportional Differentiation Model," *IEEE Network Magazine*, vol. 13, no. 5, pp. 27–34, September/October 1999.

[30] Benjamin Teitelbaum, Susan Hares, Larry Dunn, Robert Neilson, Vishy Narayan, and Francis Reichmeyer, "Internet2 QBone: Building a Testbed for Differentiated Services," *IEEE Network Magazine*, vol. 13, no. 5, pp. 8–16, September/October 1999.

[31] K. Nichols, S. Blake, F. Baker, and D. Black, "Definition of the Differentiated Services Field (DS Field) in The IPv4 and IPv6 Headers," RFC 2474, December 1998.

[32] Will E. Leland, Murad S. Taqqu, Walter Willinger, and Daniel V. Wilson, "On the Self-Similar Nature of Ethernet Traffic (Extended Version)," *IEEE/ACM Transactions on Networking*, vol. 2, no. 1, pp. 1–15, February 1994.

[33] Vern Paxson and Sally Floyd, "Wide Area Traffic: The Failure of Poisson Modeling," *IEEE/ACM Transactions on Networking*, vol. 3, no. 3, pp. 226–244, June 1995.

[34] Mark E. Crovella and Azer Bestavros, "Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes," *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, pp. 835–846, December 1997.

[35] Kihong Park and Walter Willinger, *Self-Similar Network Traffic and Performance Evaluation*, John Wiley and Sons, New York, 2000.

[36] David McDysan, *QoS and Traffic Management in IP and ATM Networks*, McGraw-Hill, New York, 2000.

[37] Erik Dahlman and Karim Jamal, "Wide-band services in a DS-CDMA based FPLMTS system," *Proceedings of IEEE Vehicular Technology Conference*, vol. VT-33, no. 3, pp. 1656–1660, August 1996.

[38] Olivier Hersent, David Gurle, and Jean-Pierre Petit, *IP Telephony Packet-Based Multimedia Communications Systems*, Addison Wesley, Great Britain, 2000.

Appendix A

Simulation Data Tables

Table A.1: Baseline Throughput Exponential Traffic

Expected Data Rate (bps)	175000	350000	400000	450000	470000
Seed1 Throughput (bps)	174861	348966	399296	449748	468745
Seed2 Throughput (bps)	174259	348628	398912	449299	468278
Seed3 Throughput (bps)	174259	348157	398313	448833	467978
Seed4 Throughput (bps)	174579	348237	398465	448703	467675
Seed5 Throughput (bps)	174416	348785	398862	449300	468207
Mean	174475	348555	398770	449177	468177
Std Dev	253	349	390	418	395
95% CI	174160	348122	398286	448658	467686
	174789	348988	399253	449696	468667

Table A.2: Baseline Delay Exponential Traffic

Expected Data Rate (bps)	175000	350000	400000	450000	470000
Seed1 Delay (sec)	0.002700	0.004900	0.007330	0.018220	0.051960
Seed2 Delay (sec)	0.002690	0.004890	0.007290	0.017550	0.045040
Seed3 Delay (sec)	0.002690	0.004870	0.007280	0.017590	0.049120
Seed4 Delay (sec)	0.002690	0.004860	0.007250	0.017670	0.047960
Seed5 Delay (sec)	0.002700	0.004890	0.007310	0.017820	0.049920
Mean	0.002694	0.004882	0.007292	0.017770	0.048800
Std Dev	0.000005	0.000016	0.000030	0.000272	0.002559
95% CI	0.002687	0.004862	0.007254	0.017432	0.045624
	0.002701	0.004902	0.007330	0.018108	0.051976

Table A.3: Baseline Throughput On-Off Traffic

Peak Data Rate (bps)	175000	200000	400000	470000	480000	500000
Seed1 Throughput (bps)	177634	199686	222967	245632	248681	249358
Seed2 Throughput (bps)	176155	196958	219457	255666	247791	241424
Seed3 Throughput (bps)	170159	202343	238954	250619	232154	261217
Seed4 Throughput (bps)	172840	193610	216899	235436	246628	251205
Seed5 Throughput (bps)	178995	204665	215278	236808	252171	258576
Mean	175157	199452	222711	244832	245485	252356
Std Dev	3613	4357	9533	8720	7734	7859
95% CI	170671	194043	210876	234006	235884	242600
	179642	204862	234546	255658	255086	262112

Table A.4: Baseline Delay On-Off Traffic

Peak Data Rate (bps)	175000	200000	400000	470000	480000	500000
Seed1 Delay (sec)	0.0021	0.0021	0.0021	0.0021	0.0108	0.1472
Seed2 Delay (sec)	0.0021	0.0021	0.0021	0.0021	0.0124	0.1439
Seed3 Delay (sec)	0.0021	0.0021	0.0021	0.0021	0.0096	0.2023
Seed4 Delay (sec)	0.0021	0.0021	0.0021	0.0021	0.0127	0.1744
Seed5 Delay (sec)	0.0021	0.0021	0.0021	0.0021	0.0140	0.1530
Mean	0.0021	0.0021	0.0021	0.0021	0.0119	0.1642
Std Dev	0.0000	0.0000	0.0000	0.0000	0.0017	0.0244
95% CI	0.0021	0.0021	0.0021	0.0021	0.0098	0.1339
	0.0021	0.0021	0.0021	0.0021	0.0140	0.1945

Table A.5: Statistical Multiplexing Throughput Exponential Traffic

Number of Users	**1**	**2**	**3**	**5**	**10**
Seed1 Throughput (bps)	399296	399211	399060	399109	399018
Seed2 Throughput (bps)	398912	398631	398569	398835	398754
Seed3 Throughput (bps)	398313	399818	398365	398486	398854
Seed4 Throughput (bps)	398465	398959	398573	398348	398834
Seed5 Throughput (bps)	398862	399829	398955	398727	398645
Mean	398770	399290	398704	398701	398821
Std Dev	390	529	292	298	137
95% CI	398286	398633	398342	398331	398650
	399253	399946	399066	399071	398992

Table A.6: Statistical Multiplexing Delay Exponential Traffic

Number of Users	**1**	**2**	**3**	**5**	**10**
Seed1 Delay (sec)	0.00733	0.00725	0.00569	0.00641	0.00682
Seed2 Delay (sec)	0.00729	0.00729	0.00566	0.00635	0.00688
Seed3 Delay (sec)	0.00728	0.00727	0.00562	0.00637	0.00687
Seed4 Delay (sec)	0.00725	0.00734	0.00565	0.00634	0.00694
Seed5 Delay (sec)	0.00731	0.00733	0.00567	0.00636	0.00691
Mean	0.00729	0.00730	0.00566	0.00637	0.00688
Std Dev	0.00003	0.00004	0.00003	0.00003	0.00005
95% CI	0.00725	0.00725	0.00563	0.00633	0.00683
	0.00733	0.00734	0.00569	0.00640	0.00694

Table A.7: Statistical Multiplexing Throughput On-Off Traffic

Number of Users	**1**	**2**	**3**	**5**	**10**
Seed1 Throughput (bps)	245632	229795	231688	232586	238835
Seed2 Throughput (bps)	255666	237587	240788	236516	229994
Seed3 Throughput (bps)	250619	229569	238829	234432	232316
Seed4 Throughput (bps)	235436	232857	232055	234882	236293
Seed5 Throughput (bps)	236808	230798	231253	238408	234686
Mean	244832	232121	234923	235365	234425
Std Dev	8720	3320	4523	2203	3431
95% CI	234006	227999	229308	232630	230165
	255658	236243	240537	238100	238684

Table A.8: Statistical Multiplexing Delay On-Off Traffic

Number of Users	**1**	**2**	**3**	**5**	**10**
Seed1 Delay (sec)	0.0021	0.0023	0.0025	0.0026	0.0028
Seed2 Delay (sec)	0.0021	0.0023	0.0025	0.0026	0.0028
Seed3 Delay (sec)	0.0021	0.0023	0.0025	0.0026	0.0028
Seed4 Delay (sec)	0.0021	0.0023	0.0025	0.0026	0.0028
Seed5 Delay (sec)	0.0021	0.0023	0.0025	0.0027	0.0028
Mean	0.0021	0.0023	0.0025	0.0026	0.0028
Std Dev	0.0000	0.0000	0.0000	0.0000	0.0000
95% CI	0.0021	0.0023	0.0024	0.0026	0.0028
	0.0021	0.0023	0.0025	0.0026	0.0028

Table A.9: Statistical Multiplexing Peak Throughput On-Off Traffic

Number of Users	**1**	**2**	**3**	**5**	**10**
Seed1 Throughput (bps)	469533	469533	415933	418866	338800
Seed2 Throughput (bps)	469533	469466	419666	439066	347200
Seed3 Throughput (bps)	469533	469533	398800	372933	337533
Seed4 Throughput (bps)	469533	469533	423133	409933	325733
Seed5 Throughput (bps)	469533	435000	427066	380133	354466
Mean	469533	462613	416920	404186	340746
Std Dev	0	15436	10936	27479	10833
95% CI	469533	443450	403343	370072	327298
	469533	481776	430496	438301	354195

Table A.10: Statistical Multiplexing Mean Throughput Ten On-Off Traffic Sources

Peak Data Rate One User (bps)	**47**	**50**	**60**	**65**	**70**
Seed1 Throughput (bps)	238835	249637	299246	320521	355730
Seed2 Throughput (bps)	229994	248345	298210	328690	348031
Seed3 Throughput (bps)	232316	250760	300924	327879	347334
Seed4 Throughput (bps)	236293	245387	294334	320422	341963
Seed5 Throughput (bps)	234686	249559	299800	326038	339787
Mean	234425	248738	298503	324710	346569
Std Dev	3431	2059	2528	3987	6202
95% CI	230165	246182	295364	319760	338869
	238684	251294	301641	329660	354269

Table A.11: Dynamic Code Sharing High QoS Throughput Exponential Traffic

High QoS Mean Data Rate (bps)	100000	150000	200000	250000	300000	400000
Seed1 Throughput (bps)	99503	148717	198304	247630	298860	399191
Seed2 Throughput (bps)	99291	148731	198635	248112	298056	398593
Seed3 Throughput (bps)	99578	149134	198189	248251	298907	398593
Seed4 Throughput (bps)	99400	149043	198755	248053	298876	398772
Seed5 Throughput (bps)	99068	148642	198349	247841	298636	398928
Mean	99368	148853	198446	247977	298667	398815
Std Dev	200	220	238	244	358	252
95% CI	99120	148581	198151	247675	298222	398502
	99616	149126	198742	248280	299112	399129

Table A.12: Dynamic Code Sharing Medium QoS Throughput Exponential Traffic

High QoS Mean Data Rate (bps)	100000	150000	200000	250000	30000	400000
Seed1 Throughput (bps)	199244	175714	129276	82539	45200	8315
Seed2 Throughput (bps)	199062	175598	128991	81664	45787	8686
Seed3 Throughput (bps)	199131	176309	129386	81475	44955	8415
Seed4 Throughput (bps)	199132	176112	128542	81936	45410	8426
Seed5 Throughput (bps)	198426	175785	129080	81851	45282	8206
Mean	198999	175904	129055	81893	45327	8410
Std Dev	298	278	303	192	298	171
95% CI	198629	175559	128679	81655	44957	8198
	199369	176248	129431	82131	45697	8621

Table A.13: Dynamic Code Sharing Low QoS Throughput Exponential Traffic

High QoS Mean Data Rate (bps)	100000	150000	200000	250000	300000	400000
Seed1 Throughput (bps)	80907	27984	5667	783	103	3.33
Seed2 Throughput (bps)	81460	27989	5521	733	83.6	7.5
Seed3 Throughput (bps)	80885	27647	5851	737	84.7	1.67
Seed4 Throughput (bps)	81185	28037	5660	684	66.94	10.27
Seed5 Throughput (bps)	81637	28029	5627	740	124.72	7.22
Mean	81215	27937	5665	735	93	6
Std Dev	287	162	119	24	21	3
95% CI	80859	27736	5517	706	66	2
	81571	28138	5813	765	119	10

Table A.14: Dynamic Code Sharing Total Throughput Exponential Traffic

High QoS Mean Data Rate (bps)	100000	150000	200000	250000	300000	400000
Seed1 Throughput (bps)	379655	352416	333249	330953	344164	407509
Seed2 Throughput (bps)	379814	352318	333148	330510	343927	407288
Seed3 Throughput (bps)	379595	353091	333426	330465	343948	407010
Seed4 Throughput (bps)	379718	353193	332958	330673	344353	407209
Seed5 Throughput (bps)	379132	352456	333057	330434	344045	407141
Mean	379583	352695	333168	330607	344087	407231
Std Dev	262	384	175	100	171	107
95% CI	379258	352219	332951	330483	343876	407099
	379908	353171	333384	330731	344299	407364

Table A.15: Dynamic Code Sharing High QoS Delay Exponential Traffic

High QoS Mean Data Rate (bps)	100000	150000	200000	250000	300000	400000
Seed1 Delay (sec)	0.0148	0.0136	0.0109	0.0092	0.0084	0.0103
Seed2 Delay (sec)	0.0147	0.0137	0.0110	0.0090	0.0085	0.0103
Seed3 Delay (sec)	0.0148	0.0136	0.0110	0.0091	0.0085	0.0101
Seed4 Delay (sec)	0.1047	0.0137	0.0109	0.0089	0.0084	0.0573
Seed5 Delay (sec)	0.0148	0.0136	0.0110	0.0091	0.0085	0.0138
Mean	0.0328	0.0136	0.0110	0.0091	0.0085	0.0204
Std Dev	0.0390	0.0001	0.0000	0.0001	0.0000	0.0200
95% CI	-0.0156	0.0136	0.0109	0.0090	0.0084	-0.0044
	0.0812	0.0137	0.0110	0.0092	0.0085	0.0451

Table A.16: Dynamic Code Sharing Medium QoS Delay Exponential Traffic

High QoS Mean Data Rate (bps)	100000	150000	200000	250000	300000	400000
Seed1 Delay (sec)	0.06	74.44	30.15	6.87	1.58	0.44
Seed2 Delay (sec)	0.08	74.03	29.30	5.89	1.10	0.15
Seed3 Delay (sec)	0.03	72.34	32.33	5.96	1.03	0.01
Seed4 Delay (sec)	0.05	74.34	30.56	6.25	1.05	1.01
Seed5 Delay (sec)	0.10	75.34	31.47	6.41	1.54	0.06
Mean	0.07	74.10	30.76	6.28	1.26	0.33
Std Dev	0.03	1.08	1.13	0.22	0.21	0.41
95% CI	0.03	72.76	29.37	6.00	1.00	-0.18
	0.10	75.44	32.16	6.55	1.52	0.84

Table A.17: Dynamic Code Sharing Low QoS Delay Exponential Traffic

High QoS Mean Data Rate (bps)	100000	150000	200000	250000	300000	400000
Seed1 Delay (sec)	203	450	545	646	523	795
Seed2 Delay (sec)	207	456	555	559	533	84
Seed3 Delay (sec)	210	450	555	592	563	0
Seed4 Delay (sec)	206	451	541	562	559	399
Seed5 Delay (sec)	196	452	543	603	418	43
Mean	204	452	548	592	519	264
Std Dev	5	2	7	20	59	168
95% CI	198	449	540	568	446	56
	211	455	556	617	592	473

Table A.18: Dynamic Code Sharing Total Delay Exponential Traffic

High QoS Mean Data Rate (bps)	100000	150000	200000	250000	300000	400000
Seed1 Delay (sec)	43.29	72.46	20.57	3.12	0.38	0.03
Seed2 Delay (sec)	44.65	72.48	20.34	2.64	0.29	0.02
Seed3 Delay (sec)	44.86	70.94	22.00	2.73	0.29	0.01
Seed4 Delay (sec)	43.96	72.41	20.63	2.64	0.27	0.06
Seed5 Delay (sec)	42.05	73.11	21.39	2.87	0.37	0.01
Mean	43.76	72.28	20.99	2.80	0.32	0.03
Std Dev	1.11	0.80	0.65	0.10	0.04	0.02
95% CI	42.39	71.29	20.18	2.68	0.27	0.00
	45.14	73.27	21.80	2.92	0.37	0.05

Table A.19: Dynamic Code Sharing High QoS Throughput On-Off Traffic

High QoS Mean Data Rate (bps)	100000	150000	175000	200000	250000
Seed1 Throughput (bps)	49830	75947	87524	95132	132301
Seed2 Throughput (bps)	49410	77145	90778	99465	117489
Seed3 Throughput (bps)	48533	71302	87791	101168	127919
Seed4 Throughput (bps)	49367	71555	93718	100325	122759
Seed5 Throughput (bps)	48206	76558	84253	96257	136779
Mean	49069	74501	88813	98469	127449
Std Dev	530	2726	3515	1896	7137
95% CI	48412	71118	84449	96115	118589
	49727	77885	93177	100824	136310

Table A.20: Dynamic Code Sharing High QoS Throughput On-Off Traffic (continued)

High QoS Mean Data Rate (bps)	300000	350000	400000	450000	470000
Seed1 Throughput (bps)	143720	138237	181315	213941	228845
Seed2 Throughput (bps)	151045	177231	195531	211587	241819
Seed3 Throughput (bps)	147672	173797	196416	232844	241536
Seed4 Throughput (bps)	144966	171021	207651	234603	226795
Seed5 Throughput (bps)	148644	183412		203648	229319
Mean	147209	168740	195228	219325	233663
Std Dev	2213	5746	5984	13379	6890
95% CI	144463	161606	187799	202715	225109
	149956	175874	202657	235935	242217

Table A.21: Dynamic Code Sharing Medium QoS Throughput On-Off Traffic

High QoS Mean Data Rate (bps)	100000	150000	175000	200000	250000
Seed1 Throughput (bps)	118511	115382	114638	89630	35147
Seed2 Throughput (bps)	124511	114635	111357	87866	44843
Seed3 Throughput (bps)	110270	111468	112725	70168	34649
Seed4 Throughput (bps)	117191	113113	116632	81361	33341
Seed5 Throughput (bps)	122481	117537	119023	95524	30420
Mean	118593	114427	114875	84910	35680
Std Dev	5509	2237	3053	9312	5434
95% CI	111754	111650	111085	73349	28934
	125431	117204	118665	96470	42426

Table A.22: Dynamic Code Sharing Medium QoS Throughput On-Off Traffic (continued)

High QoS Mean Data Rate (bps)	300000	350000	400000	450000	470000
Seed1 Throughput (bps)	41608	69167	52247	38432	37524
Seed2 Throughput (bps)	30896	29984	38948	50414	30730
Seed3 Throughput (bps)	40866	35987	36043	28508	32995
Seed4 Throughput (bps)	38586	35506	30775	33803	43322
Seed5 Throughput (bps)	32866	30380		48747	32878
Mean	36964	40205	39503	39981	35490
Std Dev	4098	4275	3994	9421	4905
95% CI	31877	34898	34544	28285	29401
	42051	45512	44462	51676	41579

Table A.23: Dynamic Code Sharing Low QoS Throughput On-Off Traffic

High QoS Mean Data Rate (bps)	100000	150000	175000	200000	250000
Seed1 Throughput (bps)	57013	18063	15191	19947	16583
Seed2 Throughput (bps)	54882	15965	15515	16778	20910
Seed3 Throughput (bps)	54850	17776	16239	15246	14655
Seed4 Throughput (bps)	59147	17111	10476	20119	17619
Seed5 Throughput (bps)	52996	13466	20132	23019	13715
Mean	55778	16476	15511	19022	16696
Std Dev	2261	1652	3436	3013	2813
95% CI	52971	14426	11245	15282	13204
	58584	18527	19776	22762	20189

Table A.24: Dynamic Code Sharing Low QoS Throughput On-Off Traffic (continued)

High QoS Mean Data Rate (bps)	300000	350000	400000	450000	470000
Seed1 Throughput (bps)	20120	35802	25082	16873	15858
Seed2 Throughput (bps)	16088	15142	18271	25179	14677
Seed3 Throughput (bps)	19976	17924	17857	13146	16043
Seed4 Throughput (bps)	21043	19652	21043	16780	20861
Seed5 Throughput (bps)	16406	15217	14586	22999	15363
Mean	18727	20747	19368	18995	16560
Std Dev	2173	2542	2378	4809	2431
95% CI	16029	17591	16415	13026	13542
	21424	23903	22320	24965	19579

Table A.25: Dynamic Code Sharing Total Throughput On-Off Traffic

High QoS Mean Data Rate (bps)	100000	150000	175000	200000	250000
Seed 1	206733	209393	217354	204260	184032
Seed 2	228843	207746	217651	204129	183244
Seed 3	213654	200536	216756	186582	177224
Seed 4	225705	201781	220827	201806	173720
Seed 5	223686	207543	223409	214811	180915
Mean	219724	205400	219199	202318	179827
Std Dev	5868	3304	2648	10081	3654
95% CI	212440	201298	215911	189803	175291
	227009	209501	222487	214832	184363

Table A.26: Dynamic Code Sharing Total Throughput On-Off Traffic (continued)

High QoS Mean Data Rate (bps)	300000	350000	400000	450000	470000
Seed 1	205449	243207	258645	269247	282228
Seed 2	198029	222358	257752	287180	287227
Seed 3	208515	227708	250316	274499	290576
Seed 4	204596	226179	253014	285166	290978
Seed 5	197917	229007		275395	277561
Mean	202901	229692	254932	278297	285714
Std Dev	4519	2915	3135	5756	5424
95% CI	197292	226073	251039	271151	278980
	208511	233311	258824	285444	292448

Table A.27: Dynamic Code Sharing High QoS Delay On-Off Traffic

High QoS Mean Data Rate (bps)	100000	150000	175000	200000	250000
Seed 1 Delay (sec)	0.0083	0.0049	0.0049	0.0055	0.0046
Seed 2 Delay (sec)	0.0083	0.0049	0.0049	0.0054	0.0049
Seed 3 Delay (sec)	0.0083	0.0050	0.0049	0.0056	0.0048
Seed 4 Delay (sec)	0.0083	0.0050	0.0049	0.0055	0.0048
Seed 5 Delay (sec)	0.0083	0.0049	0.0049	0.0055	0.0045
Mean	0.0083	0.0049	0.0049	0.0055	0.0047
Std Dev	0.0000	0.0000	0.0000	0.0000	0.0002
95% CI	0.0083	0.0049	0.0049	0.0054	0.0045
	0.0083	0.0050	0.0049	0.0056	0.0049

Table A.28: Dynamic Code Sharing High QoS Delay On-Off Traffic (continued)

High QoS Mean Data Rate (bps)	300000	350000	400000	450000	470000
Seed 1 Delay (sec)	0.0092	0.0182	0.0405	0.1430	0.1970
Seed 2 Delay (sec)	0.0087	0.0167	0.0415	0.1420	0.1920
Seed 3 Delay (sec)	0.0088	0.0185	0.0432	0.1300	0.1820
Seed 4 Delay (sec)	0.0085	0.0179	0.0376	0.1310	0.1870
Seed 5 Delay (sec)	0.0084	0.0165		0.1400	0.1880
Mean	0.0087	0.0176	0.0407	0.1372	0.1892
Std Dev	0.0002	0.0008	0.0023	0.0053	0.0037
95% CI	0.0085	0.0165	0.0378	0.1306	0.1846
	0.0089	0.0186	0.0436	0.1438	0.1938

Table A.29: Dynamic Code Sharing Medium QoS Delay On-Off Traffic

High QoS Mean Data Rate (bps)	100000	150000	175000	200000	250000
Seed 1 Delay (sec)	0.0042	0.0042	0.16	396	662
Seed 2 Delay (sec)	0.0042	0.0042	0.18	512	683
Seed 3 Delay (sec)	0.0042	0.0042	0.24	441	600
Seed 4 Delay (sec)	0.0042	0.0042	0.22	462	599
Seed 5 Delay (sec)	0.0042	0.0042	0.15	443	577
Mean	0.0042	0.0042	0.19	451	624
Std Dev	0.0000	0.0000	0.03	29	41
95% CI	0.0042	0.0042	0.15	414	574
	0.0042	0.0042	0.23	487	675

Table A.30: Dynamic Code Sharing Medium QoS Delay On-Off Traffic (continued)

High QoS Mean Data Rate (bps)	300000	350000	400000	450000	470000
Seed 1 Delay (sec)	585	809	745	572	572
Seed 2 Delay (sec)	490	596	558	760	575
Seed 3 Delay (sec)	804	737	668	620	709
Seed 4 Delay (sec)	777	631	618	640	811
Seed 5 Delay (sec)	646	615		845	589
Mean	660	678	647	687	651
Std Dev	125	57	48	93	97
95% CI	505	607	588	573	531
	815	748	707	802	771

Table A.31: Dynamic Code Sharing Low QoS Delay On-Off Traffic

High QoS Mean Data Rate (bps)	100000	150000	175000	200000	250000
Seed 1 Delay (sec)	0.00834	1198	1107	1097	1157
Seed 2 Delay (sec)	0.00834	1122	1145	1292	1182
Seed 3 Delay (sec)	0.00834	1066	1224	1105	1089
Seed 4 Delay (sec)	0.00834	1028	1309	1182	1023
Seed 5 Delay (sec)	0.00834	1027	1099	1153	1185
Mean	0.00834	1088	1177	1166	1127
Std Dev	0.0000	41	80	69	68
95% CI	0.0083	1038	1077	1080	1043
	0.0083	1139	1276	1252	1212

Table A.32: Dynamic Code Sharing Low QoS Delay On-Off Traffic (continued)

High QoS Mean Data Rate (bps)	300000	350000	400000	450000	470000
Seed 1 Delay (sec)	936	1274	1225	1049	1110
Seed 2 Delay (sec)	1111	1056	965	1149	1046
Seed 3 Delay (sec)	1165	1311	1144	1128	1239
Seed 4 Delay (sec)	1171	1002	1243	1022	1333
Seed 5 Delay (sec)	1083	1053		1146	1146
Mean	1093	1139	1144	1099	1175
Std Dev	41	122	116	52	107
95% CI	1042	988	1000	1034	1042
	1144	1290	1288	1164	1308

Table A.33: Dynamic Code Sharing High QoS Throughput Hybrid Traffic

High QoS One User Mean Data Rate (bps)	60000	70000	80000
Seed1 Throughput (bps)	302514	346885	396965
Seed2 Throughput (bps)	304933	352238	397918
Seed3 Throughput (bps)	301423	347497	406363
Seed4 Throughput (bps)	299138	348544	394033
Seed5 Throughput (bps)	298705	347941	393640
Mean	301343	348621	397784
Std Dev	2472	1885	5117
95% CI	298274	346281	391431
	304411	350961	404136

Table A.34: Dynamic Code Sharing Medium QoS Throughput Hybrid Traffic

High QoS One User Mean Data Rate (bps)	60000	70000	80000
Seed1 Throughput (bps)	201164	199390	185171
Seed2 Throughput (bps)	202256	200496	182575
Seed3 Throughput (bps)	195986	198187	175864
Seed4 Throughput (bps)	198278	200784	185974
Seed5 Throughput (bps)	195434	196612	186346
Mean	198624	199094	183186
Std Dev	2700	1717	4212
95% CI	195272	196963	177957
	201975	201225	188415

Table A.35: Dynamic Code Sharing High QoS Delay Hybrid Traffic

High QoS One User Mean Data Rate (bps)	60000	70000	80000
Seed1 Delay (sec)	0.00406	0.00380	0.00316
Seed2 Delay (sec)	0.00401	0.00379	0.00312
Seed3 Delay (sec)	0.00405	0.00379	0.00308
Seed4 Delay (sec)	0.00407	0.00384	0.00317
Seed5 Delay (sec)	0.00407	0.00381	0.00316
Mean	0.00405	0.00381	0.00314
Std Dev	0.00002	0.00002	0.00004
95% CI	0.00402	0.00378	0.00309
	0.00408	0.00383	0.00318

Table A.36: Dynamic Code Sharing Medium QoS Delay Hybrid Traffic

High QoS One User Mean Data Rate (bps)	60000	70000	80000
Seed1 Delay (sec)	0.176	3.4	161
Seed2 Delay (sec)	0.137	12.2	308
Seed3 Delay (sec)	0.189	2.9	328
Seed4 Delay (sec)	0.189	3.9	193
Seed5 Delay (sec)	0.146	6.6	208
Mean	0.167	5.8	240
Std Dev	0.024	3.6	60
95% CI	0.138	1.3	165
	0.197	10.3	314

Vita

Carl E. Fossa Jr. is currently serving as a Major in the United States Army Signal Corps. After receiving the Ph.D. degree in electrical engineering from Virginia Polytechnic Institute and State University (Virginia Tech), he will be assigned as an Assistant Professor in the Department of Electrical Engineering and Computer Science, United States Military Academy, West Point, NY. Major Fossa received the B.S. degree in electrical engineering from the United States Military Academy in 1987 and the M.S. degree in electrical engineering from the Air Force Institute of Technology in 1998. His current research interests include quality of service, integrating voice and data networks, mobile communications networks, and network simulation. Major Fossa's publications include:

Carl E. Fossa, Jr. and Nathaniel J. Davis IV, "Dynamic Code Assignment Improves Channel Utilization for Bursty Traffic in 3G Wireless," *IEEE International Communications Conference*, 2002 (accepted for publication).

Carl E. Fossa, Jr. and Nathaniel J. Davis IV, "A Dynamic Code Assignment Algorithm for Quality of Service in 3G Wireless Networks," *IEEE Wireless Communications and Networking Conference*, 2002.

Carl E. Fossa, Jr. and Nathaniel J. Davis IV, "3G Wireless Standards Offer Solution to Mobile Services in the Warfighter Information Network," *IEEE Military Communications Conference*, 2001.

Carl E. Fossa, Jr. and Nathaniel J. Davis IV, "Modeling Dynamic Code Assignment Algorithms for 3G Wireless Systems," *OPNETWORK* 2001

S. R. Pratt, R. A. Raines, C. E. Fossa, and M. A. Temple, "An Operational and Performance Overview of the Iridium Low Earth Orbit (LEO) Satellite System," *IEEE Communications Surveys*, http://www.comsoc.org/livepubs/surveys/, 2nd Quarter 1999, pp. 1-15.

Carl E. Fossa, Richard A. Raines, Gregg H. Gunsch, and Michael A. Temple, "A

Performance Analysis of the IRIDIUM Low Earth Orbit Satellite System with a Degraded Satellite Constellation," *Mobile Computing and Communications Review*, Vol. 2 No. 4, October 1998, pp. 54-61.

Carl E. Fossa, Richard A. Raines, Gregg H. Gunsch, and Michael A. Temple, "An Overview of the IRIDIUM Low Earth Orbit Satellite System," *National Aerospace Engineering Conference* (NAECON) 98, pp. 152-159.