This is to certify that the

thesis entitled

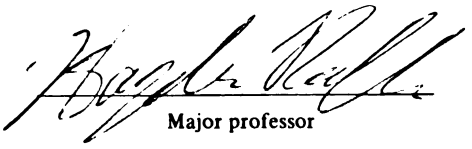
Analysis and Modeling of Errors and Losses over 802.11b Networks

presented by

Syed Ali Khayam

has been accepted towards fulfillment
of the requirements for

Masters degree in Electrical Engineering



Major professor
(Dr. Hayder Radha)

Date 04/28/03

ANALYSIS AND MODELING OF ERRORS AND LOSSES OVER 802.11B NETWORKS

By

Syed Ali Khayam

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Department of Electrical and Computer Engineering

2003

ABSTRACT

ANALYSIS AND MODELING OF ERRORS AND LOSSES OVER 802.11B NETWORKS

By

Syed Ali Khayam

Inherent vulnerability of wireless networks renders them more susceptible to errors and losses than classical (wired) media. Some of these errors are not corrected by the physical layer and result in link layer packet drops. Design of wireless applications can benefit substantially from a thorough understanding of this error/loss phenomenon at the link layer. This work, within the context of high-bitrate communication, presents statistical modeling of errors and losses over an 802.11b (wireless) network. We introduce and employ an *Entropy Normalized Kullback-Leibler (ENK)* measure to evaluate the performance of the models. First, we propose a two-state Markov chain to capture the link layer packet loss behavior. Based on ENK, it is demonstrated that the two-state Markov chain yields a very suitable approximation of the packet loss patterns. We then focus on the bit-level errors introduced by the channel. We employ the traditional modeling approach and model the process using k-order *Full State Markov (FSM)* chains (i.e., 2^k states, k = 1, 2, ..., 14). The 2^{14}-state model provides a very good approximate of the error process. However, the complexity of this model renders it impractical. This leads to one of the key contributions of this work, and that is, a *Zero-Crossing Markov (ZCM)* chain which reduces the number of states from 2^{14} (in the FSM case) to 14 while providing comparable performance.

To my parents, and in the loving memory of my grandmother

"I can no other answer make, but, thanks, and thanks."
William Shakespeare, *Twelfth Night*

ACKNOWLEDGMENTS

My feeble expression fails to describe my thankfulness to the most praiseworthy. I, therefore, use His words: "*And say: `All the praises and thanks be to Allah.*'" **The Quran** (17:111).

In addition, I would like to extend the deepest gratitude to my advisor, Professor Hayder Radha, for supporting and mentoring me from my first day at Michigan State to date. My research, in general, and this thesis, in particular, would not have been possible without his relentless contribution and innovation. I continue to be humbled and inspired by his immeasurable intellect both as a researcher and an instructor.

I am indebted to my family, especially my parents, for their continuous support and motivation throughout this effort. Words cannot express my appreciation for the selfless outlook and affectionate prayers of my parents. I thank them for always putting my interest ahead of their personal aspirations. Moreover, I thank Aapi, Wasim Bhai, Khurram, Nazish and Owais for their unreserved love.

I would like to thank Shirish, Aparna and Mujahid for being wonderful colleagues and even better friends. Their contribution to my overall research and academic experience cannot be summarized in such short space. Nevertheless, I think they deserve a personal mention here: Shirish – for his ever critical, and mostly correct, research perspective; Aparna – for always showing absolute faith in me; Mujahid – for being a considerate and steadfast friend.

I am also grateful to Professor Jack Deller for helping me identify and pursue future research directions. His considerate encouragement of my novice research ideas never failed to boost my confidence. Likewise, I thank Dr. Percy Pierre for his accommodating attitude as a member of my advisory committee. I am honored to have three of the most distinguished scholars on my committee, namely Dr. Radha, Dr. Deller and Dr. Pierre.

Lastly, I acknowledge Dmitri for being an excellent research and racquetball partner.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS AND ACRONYMS

ACK	Acknowledgment
AP	Wireless Access Point
BSS	Basic Service Set
CDF	Cumulative Density Function
CRC	Cyclic Redundancy Check
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
DCF	Distributed Coordination Function
DS	Distribution System
ENK	Entropy Normalized Kullback-Leibler measure
FCS	Frame Check Sequence
FEC	Forward Error Correction
FSM	Full State Markov chain
IP	Internet Protocol
LAN	Local Area Network
LoS	Line of Sight
MAC	Medium Access Control
PC	Point Coordinator
PCF	Point Coordination Function
PDF	Probability Density Function
PHY	Physical Layer
pmf	Probability mass function

QoS	Quality of Service
RVLC	Reversible Variable-Length Coding
STA	Wireless Station
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VLC	Variable-Length Coding
ZCM	Zero-Crossing Markov Chain

CHAPTER 1

INTRODUCTION

With the advent of computer networks, real-time multimedia communication entered a new realm. The rapidly growing ubiquity of these networks, especially the Internet, rendered a promising alternative for distribution of real-time multimedia. Such considerations stipulated migration of contemporary media technologies to the electronic domain. In order to strike a balance between the scarce bandwidth and delay-sensitive nature of the real-time content, a lot of research effort was targeted at optimization and calibration of multimedia schemes. These efforts led to improved multimedia coding schemes that were tailored to meet specific (network-oriented) constraints, for example, bandwidth, multimedia quality, network conditions etc. Some examples of these coding schemes are MPEG-4, MPEG-4 FGS, H.26L/JVT, G.723, G.729a/e etc. These coding schemes were designed to adapt to a certain set of network conditions and, in turn, render a minimum quality of service (QoS). The last decade has witnessed the unprecedented success of multimedia delivery over local/wide area networks, in particular the Internet.

Despite the tremendous promise of computer networks, they suffer from errors and losses in the presence of network congestion and transmission medium degradation. This can have an adverse affect on the perceived quality of a real-time multimedia transmission. Classical data applications provide reliability by attempting to recover the corrupted/lost packets through retransmissions. Real-time requirements of emerging delay-

sensitive multimedia applications (e.g., internet telephony, video conferencing, multicast audio/video etc.) necessitate a retransmission-less infrastructure (to avoid low-latency and/or potential implosion of feedback messages). Meanwhile, one positive aspect of such applications (especially those delivering streaming media) is their inherent tolerance to a certain level of errors and losses in the multimedia content. Design and implementation of such applications require a thorough understanding of the error and loss patterns encountered over the network.

The reasons stated above motivated studies, such as the ones reported in [1] and [2], which analyzed and modeled packet losses over the Internet. Most of the losses over wired media are a result of network congestion under high-load conditions. Packet corruptions are very infrequent and, therefore, packets with errors are dropped without regard to the number and location of such errors. Over the last decade, the reliability of wired networks has allowed a steady (cost-effective) increase in the available bandwidth. Migration from kilobit copper wire to gigabit fibre optics is a clear testament of this trend.

While the analysis of network impairments for wired networks received some attention by researchers, thorough analysis and modeling of errors and losses encountered over wireless Local Area Networks (LANs), in general, and in the context of the Medium Access Control (MAC) layer[1], in particular, have not been conducted. The primary objec-

[1] The MAC layer is typically divided into two sub-layers; the *data link layer* and the *medium access sub-layer*. The functionality of these sub-layers, though quite different, is irrelevant in this work. Hence, the terms *MAC layer* and *link layer* will be used interchangeably throughout this document.

tive of this thesis is to fill this gap. In other words, this thesis focuses on the analysis and modeling of errors and losses observed at the MAC layer of the most widely used wireless LANs: 802.11b.

1.1. Motivation

Due to the increased error-rate, design considerations for the wireless networks differ substantially from the wired media. In order to cater for these errors, designers of the wireless networks introduced enhanced robustness at the physical (PHY) and MAC layers of the protocol stack. The promise of real-time multimedia over the wired Ethernet is often attributed to the simplicity and pragmatism of TCP/IP protocol suite. This protocol suite has shown enough promise to drive the market toward real-time communication over the omnipresent computer networks. Thus the improved wireless PHY-MAC combination was in turn deployed with the classical TCP/IP stack at the network and transport layers. Another rationale for employing TCP/IP was seamless integration of wireless and wired frameworks. However, the TCP/IP protocol suite has been designed for the wired medium and its (unmodified) deployment in wireless networks caused overall performance degradation. In particular, and keeping in view the increased error-rate, the error protection schemes (e.g., Cyclic Redundancy Check) at the MAC, network and transport layers came under question. Our previous work evaluated the efficacy of cross-layer protocol strategies to mitigate the above drawback of the 802.11 networks [3]. Summary of this work is presented in Appendix A.

Real-time content is inherently resilient to a certain level of errors and losses. In addition, applications targeted for wireless networks are equipped with better error resilience

features, for example, reversible Variable-Length Coding (VLC) for MPEG-4 [4]. Therefore, it was suggested that wireless multimedia applications can tolerate, and therefore should, receive corrupted packets. This results in improvement of end-to-end bandwidth utilization. Schemes tailored for delivery of multimedia content over wireless networks consider the increased error-rate and attempt to improve bandwidth utilization by processing corrupted packets. This allows multimedia encoders to adjust adaptive parameters based on the number and distribution of corruptions/losses. For example, an audio encoder can adjust its rate [5], and a video encoder can invoke error-resilience features, such as, data partitioning, reversible VLCs etc. [3]. Modifications at the link and transport layer have also been proposed to facilitate such real-time error-resilience features (e.g., partial-protection of sensitive headers in UDP Lite [6], [7], [8]). For such technologies, error patterns inside a packet payload are important, since, decision to drop or retain a packet is taken at the application layer. Hence, it is important to investigate how many and what type of errors are introduced in packets transmitted over wireless networks.

Typically, statistical channel models are employed to characterize the error/loss patterns observed over wireless networks. The advantages of developing such models are manifold. First of all, a statistical model can be used to study the impact of a system on a set of inputs. For example, an input applied to the system is distorted by noise. The signal observed at the output can be used to characterize and remove the noise. The second reason for employing the models is to study the source without having the real-world process available. The complexity of obtaining a signal from the actual source can be quite high under some circumstances. For example, and as elaborated later, we have to modify a certain device driver in a particular operating system in order to observe the real-life

process. This is quite impractical, and potentially impossible, in most scenarios due to the underlying dependence on programming languages, I/O control and operating systems. Such a situation can greatly benefit from a statistical model that will allow study of the actual source via simulations. Lastly, statistical models are important because they allow simple and efficient realizations of important practical systems.

1.2. Research Problem

We analyze and model the error and loss patterns introduced by an 802.11b channel under realistic settings. We perform analysis of errors at the 802.11b link layer, that is, errors that are not corrected by the physical layer. Hence, our definition of channel encompasses the wireless medium and the 802.11b PHY. This MAC layer channel definition with respect to the TCP/IP model is shown in Figure 1.

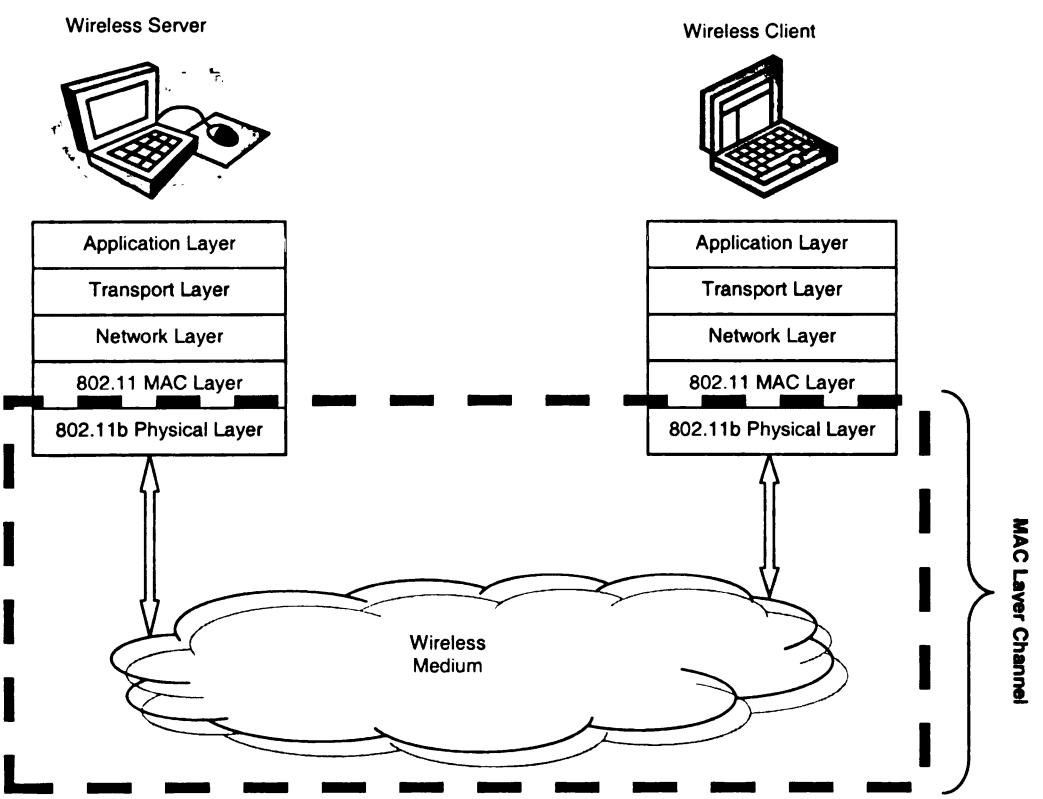


Figure 1. A traditional protocol stack highlighting the "MAC Layer Channel".

We focus on packet- and bit-level analysis and modeling. Our analysis evaluates the feasibility of supporting high-bitrate, especially real-time, applications over 802.11b networks. We conduct our study and analysis at the full-rates provided by the different modes at the physical layer. Consequently, our definition of "high-bitrate" follows the 2, 5.5, and 11 Mbps bitrates supported by the 802.11b standard. Some similar studies have been conducted to model loss patterns observed on the link layer of GSM-based networks [9], [10]. In previous works, we presented a simple model to capture the 802.11b error and loss patterns [11], [12]. This work is an extension of the previous model to provide subtle performance improvements by adding slight complexity to the model. Our decision to focus on high-bitrate multimedia has been mainly influenced by the fact that 802.11b LAN support (relatively) high bitrates. Consequently, it is quite feasible that in the near future these LANs could utilize unicast and/or multicast frameworks to provide real-time television-like services to large numbers of users. Evaluating this feasibility requires an understanding of the error and loss performance at different layers of the protocol stack. Here, we rely on the assumption that, in addition to its adherence to the constraints of the underlying standard, the physical layer is providing the best possible performance at a given desired bitrate and under given channel conditions.

Our analysis at 2 Mbps outlines a very low error/loss rate, which cannot have a significant impact on the quality of the multimedia transmission, thus deeming further study at this bitrate inconsequential. At 11 Mbps the error rate is very high and it is concluded that a practical forward error correction scheme cannot remedy these errors. Therefore, most of the analysis and modeling presented in this thesis focuses on 5.5 Mbps to support high-bitrate multimedia.

Packets corrupted on the wireless medium are dropped at the 802.11 MAC layer irrespective of the number and location of the errors. Naturally, this results in a much higher packet loss ratio as opposed to the wired media. Our analysis shows that the 802.11 packet losses are largely bursty and, therefore, can be modeled as a two-state Markov model. We propose and employ an information theoretic measure, based on Kullback-Leibler distance, to analyze the appropriateness and efficacy of the model. This measure, which we refer to as *Entropy Normalized Kullback-Leibler measure (ENK)*, indicates an entropy based source-coding-like overhead rendered by the model. Based on ENK, our results show that the 2-state Markov chain captures the packet loss process quite adequately.

Our previous work has shown advantages of cross-layer schemes to improve wireless bandwidth utilization by relaying partially damaged packets to the application layer (See Appendix A, [3]). Such schemes can greatly benefit from analysis and modeling at the byte/bit-level. Therefore after developing the packet-level model, we shift our focus to corrupted packets and perform analysis on the bit-level. This analysis and modeling is an extension of our previous work presented in [11], [12].

Our analysis establishes that the 5.5 Mbps trace segments exhibit largely non-stationary behavior. In addition, autocorrelation of these trace segments outlines a certain level of temporal dependencies. However, the temporal dependence decreases with time. For processes exhibiting such decreasing statistical correlation, there exists a length k such that the conditional probability does not change substantially if conditioned on subsequences longer than k [13], [14]. The argument k is generally referred to as *memory length*. The process can, therefore, be modeled by an order k discrete-time Markov chain.

Traditionally, a k order Markov chain has 2^k discrete states. Thus the number of states increases exponentially with the increase in memory length. This renders high order models computationally inconceivable. Previous works, in an attempt to restrain the computational complexity, have used orders smaller than the memory length [1], [9], [10].

We compute the autocorrelation for different 5.5 Mbps traces and establish that the memory length is always less than 14. We evaluate the suitability of 2^k state Markov models ($k = 1,2,\ldots,14$) to capture the channel behavior. We refer to this choice of Markov states as a *Full State Markov (FSM) chain*. It can be observed that for k = 1, we have a simple 2-state Markov Chain (commonly referred to as the *Gilbert Model*). As the order of the Markov chain is increased (i.e., k is incremented), the complexity of the model increases exponentially. As mentioned previously, we employ the ENK measure to gauge the performance of each model. It is observed that models of order greater than 10 render satisfactory performance. The order 14 model provides the best (ENK-based) performance. However, the high complexity of these models makes them impractical for most real-life simulation scenarios.

We, therefore, propose a *Zero-Crossing Markov chain (ZCM)*, in which we define the states based on the bit transitions within a memory window. This choice of states reduces the total number of Markov states from 2^k to k. Thus the ZCM provides orders of magnitude reduction in the model complexity. We again employ the ENK measure to show that the performance of the ZCM is comparable with FSM.

1.3. Thesis Organization

The remainder of this document is structured as follows. Chapter 2 provides essential background information on 802.11 networks. We discuss the basic network entities, access methodologies and some important 802.11 MAC layer functionalities. In addition, this chapter provides background information on Markov chains and the use of autocorrelation analysis in determining the order of a Markov chain. Lastly, this chapter describes the use of ENK to determine the source-coding overhead of a model which in turn quantifies its statistical performance.

Chapter 3, first, describes the experimental setup to generate error traces at 2, 5.5 and 11 Mbps. We perform packet-level throughput analysis on the collected traces. This analysis outlines that very high (packet) throughput at 2 Mbps renders further analysis on this bitrate inconsequential. However, 11 Mbps has extremely low (packet) throughput and is, therefore, in adequate for high-bitrate multimedia. Hence, we focus our attention to 5.5 Mbps which has lower throughput than 2 Mbps but still shows significant promise for high-bitrate multimedia support. We determine that the packet losses are bursty and, therefore, can be modeled as a two-state Markov chain. This model is evaluated using the ENK criterion. Some concluding analysis of this chapter shows that the byte-level throughput is much higher than the packet-level throughput.

Chapter 4 provides some preliminary bit-level analysis and establishes that the error traces exhibit largely non-stationary behavior. Order of the respective Markov chain is determined using autocorrelation analysis. We develop FSM for various window sizes and the performance is evaluated using ENK. The FSM is deemed adequate for orders

greater than 10, however, the complexity of such a model is unreasonable. We, therefore, define the ZCM in order to reduce the overall complexity of the Markov chain. Results outlining the comparative performance of the ZCM are provided at the end of this chapter.

Chapter 5 iterates some key conclusions of this work and identifies future directions.

CHAPTER 2

BACKGROUND

In this chapter we provide necessary background information on 802.11b networks, autocorrelation of random variables for determining the order of a Markov chain, theory of Markov chains and the use of Kullback-Leibler distance to quantify the information theoretic (source-coding-like) overhead.

2.1. 802.11b Wireless Networks

Due to their (relatively) high data rates and use of the time-tested TCP/IP protocol suite, 802.11b networks have experienced widespread deployment in recent years. These LANs are finding their way into homes and businesses ubiquitously. However like other wireless technologies, 802.11b networks also suffer from severe quality degradation in the presence of physical obstructions and inter-symbol-interferences. Performance evaluation of 802.11 networks has emerged as an area of active research [17], [18], [19].

The Basic Service Set (BSS) is the basic building block of an 802.11 LAN [15], [16]. A BSS contains member stations (STAs) that can transmit/receive data within the coverage area of the BSS. There are two modes of operation in an 802.11 network; ad hoc and infrastructure. In the ad hoc mode STAs can communicate with each other directly. To allow expansion of 802.11 networks, a distribution system (DS) is used. A DS provides address to destination mapping and seamless integration of multiple BSSs. A STA that

provides access to the DS by providing DS services is called an Access Point (AP). Figure 2 shows the basic components of an 802.11 network.

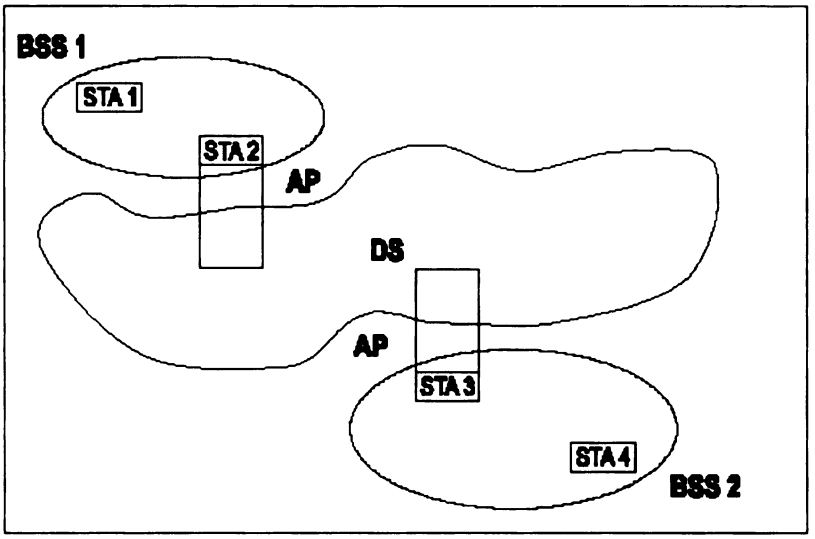


Figure 2. Typical components of an 802.11b network.

Many wireless stations share the same bandwidth. Hence, the 802.11 standard employs two access methodologies that resolve channel contention. The fundamental and mandatory access methodology is called Distributed Coordination Function (DCF). It employs *carrier sense multiple access with collision detection* (CSMA/CD) and uses random backoff in case of deferral [16]. There is an optional access methodology for infrastructure network configurations, known as Point Coordination Function (PCF). It uses a point coordinator (PC) at the AP to determine which STA has the right to transmit. In other words, the PC acts as a polling master to allow STAs to transmit over the wireless medium.

802.11b networks support four basic physical layer data rates that are 1 Mbps, 2 Mbps, 5.5 Mbps and 11 Mbps. Increase in the data rate reduces the robustness of the 802.11b physical layer. More specifically, the bandwidth conserved by dropping to a lower bitrate

is used to render added reliability at the physical layer. The physical layer data rate is decided on the basis of the application requirements and network conditions. For example, low-bitrate applications should employ the more reliable bitrates, such as, 1 and 2 Mbps. However, even for high-bitrate applications if the number of retransmission requests exceeds a certain threshold, the AP drops down to a lower bitrate.

For retransmissions 802.11b relies on a 32-bit Frame Check Sequence (*FCS*) that computes checksum over the entire frame. Immediate positive acknowledgement (ACK frame) is employed to signal successful transmission of a frame. If a frame fails checksum then it is dropped at the receiver MAC layer. The sender after timing out schedules a retransmission (identified by its sequence number with a *retry* flag set in the headers).

2.2. Autocorrelation of Random Processes

Let $x(n_1)$ and $x(n_2)$ be two random variables derived from a random process X. The "sample" correlation of these random variables is defined as,

$$\gamma(n_1, n_2) = \mathrm{E}\{x(n_1)x(n_2)\} \tag{2.1}$$

where, $\mathrm{E}\{x\}$ represents the "sample" mean of the random variable x. Since both $x(n_1)$ and $x(n_2)$ are derived from the same random process, the correlation is often referred to as *sample autocorrelation* [23], [24]. Let n_1 and n_2 be separated in time by a lag η such that

$n_1 = 0$ and $n_2 = n_1 + \eta = \eta$. In this case, the autocorrelation becomes a function of the lag η, and an *sample autocorrelation coefficient*[2] can be defined as,

$$\rho(\eta) = \frac{E\{x(0)x(\eta)\} - E\{x(0)\}E\{x(\eta)\}}{\sigma_{x(0)}\sigma_{x(\eta)}} \tag{2.2}$$

where, σ_x represents the "sample" standard deviation of the random variable x. The sample autocorrelation function, when computed for different values of the lag, is a direct metric for the level of temporal dependence in the random process. Since there is one-to-one correlation between symbols (random variables) at lag zero, the autocorrelation has its maximum value at this point. This can be easily verified by a closer examination of (2.2),

$$\rho(0) = \frac{E\left\{x^2(0)\right\} - \left(E\{x(0)\}\right)^2}{\sigma^2_{x(0)}} = 1$$

For a large range of statistical data, autocorrelation between two symbols (random variables) decreases rapidly with the increase in the lag between them. Lag beyond which the autocorrelation coefficient drops to an insignificant value is known as the *memory length* of the process. In slightly relaxed jargon, memory length represents the lag beyond which the symbols (random variables) comprising the random process are (virtually) uncorrelated [13].

---

[2] This representation is also referred to as the *normalized sample autocorrelation coefficient* since it is normalized by the standard deviation thereby ensuring $|\rho(\eta)| \leq 1$.

2.3. Markov Chains

Markov chains are employed to model a wide range of statistical data. In particular, data with temporal dependencies can be modeled using Markov chains. Let a stochastic process, X_n, take on values denoted by non-negative integers {0, 1, 2, ...}. If $X_n = i$ then the process is said to be in state i at time n. Whenever the process is in state i there is a fixed probability that the next state of the process will be state j. If that probability can be expressed as,

$$P\{X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, \ldots, X_1 = i_1, X_0 = i_0\} = P\{X_{n+1} = j | X_n = i\} \quad (2.3)$$

for all states i_0, i_1, ..., i_{n-1}, i, j and all $n \geq 0$, such a stochastic process is known as a *Markov Chain* [25]. The property given in (2.3) is commonly referred to as *Markov Property*. Thus, for a Markov chain the conditional distribution of any future state X_{n+1} given the past states X_0, X_1, ..., X_{n-1} and the present state X_n is independent of the past states and depends only on the present state. Equation (2.3) is also referred to as *homogeneity property* since it ensures that the transition probabilities do not vary with time.

Let us define $P_{ij} = P\{X_{n+1} = j | X_n = i\}$, that is, the probability of transiting to state j from i. Since P_{ij} represents a probability measure, it exhibits the following properties,

$$P_{ij} \geq 0, \qquad i, j \geq 0, \qquad \sum_{j=0}^{\infty} P_{ij} = 1, \qquad i = 0, 1, \ldots$$

The probability of transiting to the next state can be represented in a matrix form (P),

$$P = \begin{bmatrix} P_{00} & P_{01} & P_{02} & \cdots \\ P_{10} & P_{11} & P_{12} & \cdots \\ \vdots & & & \\ P_{i0} & P_{i1} & P_{i2} & \cdots \\ \vdots & \vdots & \vdots & \end{bmatrix}$$

P is referred to as the one-step transition probability matrix.

The memory length of a Markov chain is commonly referred to as its *order*. Discussion in the preceding section outlined that autocorrelation analysis can be performed on the realizations of a random process to determine the appropriate order of the respective Markov chain. This observation will be used throughout this work to identify the order of Markov chains.

2.4. Information Theoretic Evaluation of Markov-Based Models

A statistical source emits a random sequence of *symbols* that belong to a particular set of permissible values. This set is commonly referred to as the *alphabet set*. Each such symbol is an outcome of a *random experiment*. For example, a coin-toss random experiment represents a source that emits symbols from an alphabet set $\{Head, Tail\}$. Information theoretic measures quantify the minimum amount of information that needs to be communicated by the source encoder in order to uniquely specify each random outcome. For all information measures, a fundamental assumption is made about the availability of a probabilistic measure to represent the random experiment.

2.4.1. Entropy of a Random Experiment

From a source coding viewpoint, entropy provides a measure for the average number of bits required to represent the source completely [27]. Note that the entropy is largely dependent on the random variable (or probability mass function) used to represent the random experiment. We assume the availability of an appropriate random variable representing the random experiment. Entropy of this random variable provides a weighted average of the minimum information[3] of the source. Entropy is expressed as,

$$H(p(X)) = -\sum_{i=1}^{N} p(X = x_i)\log\left(p(X = x_i)\right) \tag{2.4}$$

where, $p(x_i)$ is the value of the probability mass function (pmf) at $X = x_i$, and, N represents the total number of outcomes.

2.4.2. The Kullback-Leibler Distance

Let $p(X)$ and $q(X)$ be two probabilities distributions based on the random variable X and an alphabet set Ψ. *Kullback-Leibler distance*[4] provides a measure as to how (statistically) different p and q are from each other. Mathematically it can be expressed as,

---

[3] Here, the term *information* corresponds to the number of bits required to uniquely represent all possible outcomes of the source.

[4] In the literature, Kullback-Leibler distance is also referred to as *relative entropy* and *informational divergence*.

$$D\big(p(X)\|q(X)\big)=\sum_{i=1}^{N}p(X=x_i)\log\left(\frac{p(X=x_i)}{q(X=x_i)}\right) \tag{2.5}$$

Thus, Kullback-Leibler distance provides a non-negative statistical divergence measure which is zero if and only if $p = q$. It can again be observed that Kullback-Leibler distance is completely dependent on the choice of the random variable or, in other words, the choice of the probability distribution used to represent the random experiment. Therefore, utmost care should be exercised in choosing an appropriate random variable to represent a random observation. In addition, a closer examination of (2.5) reveals non-symmetry and violation of the triangle inequality. The Kullback-Leibler distance, therefore, is not a true metric [26].

From a source coding point of view, let us assume that $p(X)$ represents the probability distribution of the actual source, whereas, $q(X)$ represents the distribution of a model approximating the source. Let, $m = D\big(p(X)\|q(X)\big)$ quantify the statistically divergence between the actual source and its model approximation. Thus "m" provides a direct measure of the number of overhead bits incurred because an approximation is used instead of the actual source. For example, assume that the entropy of the actual source be H_s. More specifically, H_s is the minimum number of bits required to represent all possible outcomes of the actual source. The Kullback-Leibler distance (m) between the actual source distribution and the model distribution corresponds to the number of extra bits required to represent all possible outcomes of the model. In other words, the total number of bits required to represent the model are $H_s + m$. Hence, an overhead of m-bits is incurred due to the use of a model instead of the actual source.

Nevertheless, in order to accurately judge the performance of the model, this measure needs to be weighed in accordance with the entropy. For example, let us assume that the entropy of the source is 20 bits whereas the overhead incurred by the model is 0.75 bits. The overhead is relatively insignificant since the source requires a large number of bits to be represented. However, for the same overhead, if the entropy of the source is lower, say 1 bit, this overhead is extremely high. Hence, to accurately represent the performance of a model both entropy and Kullback-Leibler distance should be taken into consideration.

2.5. Entropy Normalized Kullback-Leibler Measure

The entropy function provides a measure of average number of bits required to represent a source. Also, Kullback-Leibler distance specifies how many extra bits are incurred due to the use of a model instead of an actual source. Both these measures can be used collectively to indicate the level of overhead incurred by a particular (source) model. We define a new measure *Entropy Normalized Kullback-Leibler measure (ENK)* to evaluate the performance of a model. More specifically,

$$ENK\left(p(X)\|q(X)\right)=\frac{D\left(p(X)\|q(X)\right)}{H\left(p(X)\right)} \tag{2.6}$$

where, $D(p\|q)$and $H(p)$are defined in (2.5) and (2.4) respectively.

Thus ENK provides a measure for the level of source-coding-like overhead incurred by employing a model instead of the actual (random) source.

CHAPTER 3

PACKET-LEVEL ANALYSIS AND MODELING

In this chapter we first introduce the simulation setup employed to collect error traces over the 802.11b network. This explanation is followed by some preliminary analysis and modeling of observed loss and error patterns in the collected traces. As emphasized previously, our focus is to investigate the feasibility of supporting high-bitrate real-time multimedia (i.e., 2Mbps and above).

The analysis and modeling provided in this chapter mainly focuses on packet-level traces. Note that the next chapter contemplates bit-level traces. It can be argued that the bit-level modeling can be extended to bytes and, in turn, packets. However, packet-level analysis in this chapter is necessary to motivate the need for analysis and modeling at finer levels of granularity. Furthermore, packet-loss events represent a key performance measure for any communication network. Recall that the 802.11 MAC-layer discards the corrupted packets without regard to the number and location of the errors (see Section 2.1). Our analysis outlines that the bit-level throughput is generally quite high at all the bitrates. However, these errors are spread across multiple packets thereby reducing the packet-level throughput substantially. Thus the primary objective of this chapter is to present analysis that leads to a packet loss model.

Traditionally, retransmissions are employed to recover corrupted and lost packets. The delay-sensitive nature of many real-time applications necessitates a retransmission-

less framework. The 802.11b standard requires the MAC layer to verify a 32-bit checksum on each received frame. MAC frames corrupted on the wireless medium fail the FCS and are dropped regardless of the number and temporal location of these errors. Due to the inherent lossy nature of the wireless medium, real-time applications might prefer corrupted packets as opposed to dropped packets. Real-time applications can tolerate corruptions to some degree and error resilience features of some modern real-time applications further boost performance in such an error-prone scenario [3]. Furthermore, retransmission-based recovery of corrupted packets is not a viable option in some key emerging multimedia applications (e.g., multicast audio/video). Therefore, we performed all analysis while disabling the MAC layer retransmission functionality at the server.

3.1. Experimental Setup

Our simulation setup employed an 802.11b AP operating in DCF mode and three wireless stations communicating in the infrastructure network configuration. One of the stations was operating as the server and the remaining two as multicast clients. All wireless stations were Linux boxes using DLink DWL-650 PCMCIA wireless cards with Prism2 chipset (linux-wlan-ng-0.1.14-pre3) device drivers [20]. Prism2 device drivers support a "monitor mode" that allows delivery of MAC frames with failed FCS. Source code of the device driver at the clients was modified to capture screenshots of all (corrupted/uncorrupted) MAC data frames

Initially, the server was placed in clear line of sight (LoS) of the AP. The AP was forced to transmit at 2, 5.5 and 11 Mbps for each observation. The server was stationary and transmitted a continuous stream of predetermined patterns to the multicast clients. Traces were generated for each bitrate at different stationary client positions with and without LoS. It was observed that, with clear LoS, the error rate at all bitrates was extremely low. Such excellent performance deemed further LoS study inconsequential. Hence, both clients were positioned in a separate room across two walls in order to simulate a more realistic business/classroom/home-network wireless setup, as shown in Figure 3.

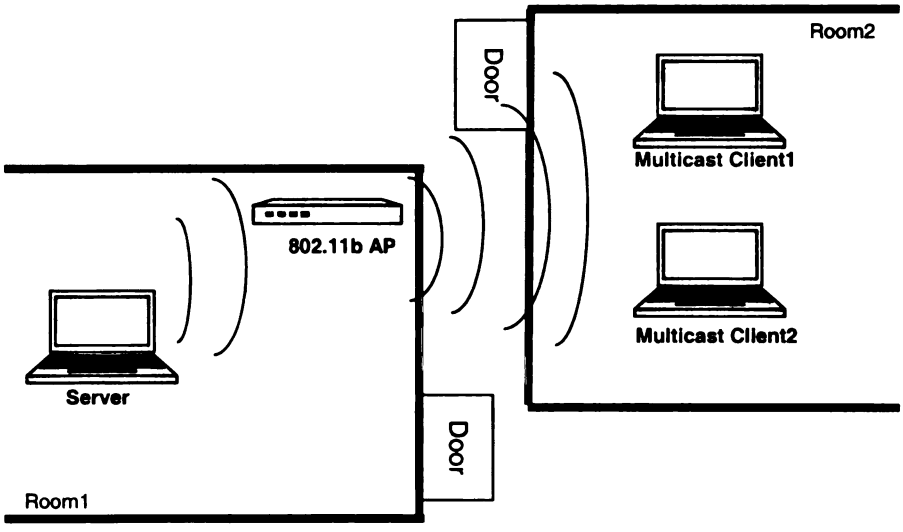


Figure 3. Simulation setup for error trace collection.

A total of three experiments were conducted for each bitrate. Each experiment was conducted at a different (stationary) client position and involved the transmission of approximately 2048 packets of 512 bytes each (512x2048 = 1 MByte of data). These experiments were performed at different times of the day to nullify the effects of unrelated

traffic and interference. Special care was taken to ensure that the distance from the AP, in all measurements, does not exceed the operational range of a typical 802.11b network and our particular hardware.

3.2. Packet-Level Analysis and Modeling

This section first provides MAC layer "throughput" statistics of the collected error traces. MAC layer "throughput" corresponds to the number of packets that are getting relayed to the network layer, that is, ratio of the number of packets received by the MAC layer to the number of packets received by the network layer (See Figure 4). Recall that the 802.11b MAC drops all corrupted packets regardless of the number and location of the errors. Thus, each packet with (one or more) errors results in a packet loss and adversely impacts the throughput. This reduction in throughput propagates to the application layer and, therefore, in a retransmission-less scenario could drastically degrade the quality of perceived media.

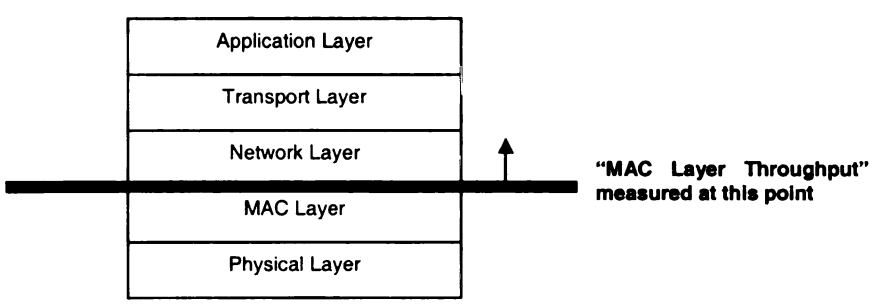


Figure 4. Throughput measurement at the MAC Layer of the protocol stack.

Here it is noteworthy that bursty errors/losses are much more detrimental to the multimedia content than isolated losses. The rationale for such an attribute is the context-oriented nature of multimedia data. For example, assume that each packet represents a video frame of a real-time transmission. In such a scenario, losing frames in a burst will cause significant degradation in video quality. On the other hand, if the same number of losses is spread over time then the adverse impact will be considerably less noticeable.

Mathematically throughput can be expressed as,

$$\tau_{packet} = \frac{total\ \ number\ \ of\ \ uncorrupted\ \ packets\ received\ at\ the\ MAC\ Layer}{total\ \ number\ \ of\ \ transmitted\ \ packets} = 1 - P_{packet} \quad (3.1)$$

where, P_{packet} represents the probability a packet being dropped, i.e., the probability that a packet has (one or more) errors.

In the last section of this chapter, we extend the throughput analysis to the byte-level in order to develop a more thorough understanding of the overall error phenomenon.

3.2.1. Packet-Level Throughput Analysis

The average packet-level burst length and throughput statistics for all bitrates are tabulated in Table 1. Interestingly, the average packet throughput at 2 Mbps is greater than 95%. Furthermore, and keeping in view the maximum burst of 7 consecutive bad packets, the mean packet burst length at 2 Mbps is quite small. This behavior is somewhat bursty since it implies that the loss bursts generally comprise of 2 consecutive packet drops. Recall that multimedia content is inherently resilient to a certain level of errors and losses. This is particularly true for scalable multimedia solutions such as the

MPEG-4 FGS video coding method [21], [22]. Consequently, the high-quality performance at 2 Mbps construed further investigation at this bitrate insignificant. Nevertheless, analysis at higher bitrates is essential to evaluate the feasibility of high-bitrate applications.

	Maximum Packet Burst	**Mean Burst Length**	τ_{packet} (%)
2 Mbps	7	1.458	94.56
5.5 Mbps	21	1.978	76.61
11 Mbps	22	4.16	37.87

Table 1. Packet-Level Burst-Length and Throughput Statistics at 2, 5.5 and 11 Mbps

The packet-level throughput reduces monotonically with the increase in bitrate. This observation is unsurprising since the robustness at the physical layer is inversely proportional to the bitrate. In particular, notice that the throughput at 11 Mbps is very low in comparison with 5.5 Mbps. The maximum packet bursts for both 5.5 and 11 Mbps are almost identical. However, the mean burst length at 5.5 Mbps is significantly smaller than 11 Mbps. Multimedia content is highly sensitive to bursty losses. Consequently, the quality degradation due to the 11 Mbps losses will be much more profound than the 5.5 Mbps case.

The above observations conclude that Forward Error Correction (FEC)-based recovery[5] is required to deliver high-quality multimedia at bitrates higher than 2 Mbps. The

[5] The FEC can be provided at the application layer with lower layers relaying corrupted packets to the higher layers. The lower layers should employ protocol modifications such as [6], [3].

FEC scheme in question can be an erasure[6]-recovery scheme since all packets with errors have already been dropped at the MAC layer. The loss patterns observed at 11 Mbps hamper the design of an efficient FEC scheme. Henceforth, all following analysis and modeling presented in this work focuses on the 5.5 Mbps case. Nevertheless, all analysis and modeling techniques employed are generic and are readily applicable to other bi-trates.

3.2.2. Packet-Level Modeling

Analysis in the last section reveals that the packet losses are bursty. Therefore, we employ a simple two-state Markov model to capture the bursty packet loss behavior at 5.5 Mbps. The two states are: 1) a *GOOD* state representing a successful packet transmission (i.e., received packet had no errors); and 2) a *BAD* state representing a packet loss (i.e., received packet had one or more errors and was dropped). The transition probabilities of this model are given as,

$$P_{gg} = P\left[X_{n+1} = GOOD \middle| X_n = GOOD\right]$$

$$P_{gb} = P\left[X_{n+1} = BAD \middle| X_n = GOOD\right] = 1 - P_{gg}$$

$$P_{bb} = P\left[X_{n+1} = BAD \middle| X_n = BAD\right]$$

$$P_{bg} = P\left[X_{n+1} = GOOD \middle| X_n = BAD\right] = 1 - P_{bb}$$

---

[6] An erasure is an error such that the position of the error is known.

where, $X_n = GOOD$ and $X_n = BAD$ respectively refer to an uncorrupted (error-free) and a corrupted (erroneous) packet received at time $t = n$. In other words, *GOOD* and *BAD* represent a successful packet transmission and a packet loss respectively. The long-run (stationary) probability of a packet loss can be expressed as,

$$P_{packet} = \frac{P_{gb}}{P_{gb} + P_{bg}} \tag{3.2}$$

Transition probabilities of the packet-loss model were generated from three different packet-level traces using the above equations. A two-state transition probability matrix was generated for each of these traces. These three transition probability matrices were then averaged to obtain the (overall) transition probabilities. The model and its respective transition probabilities are given in Figure 5.

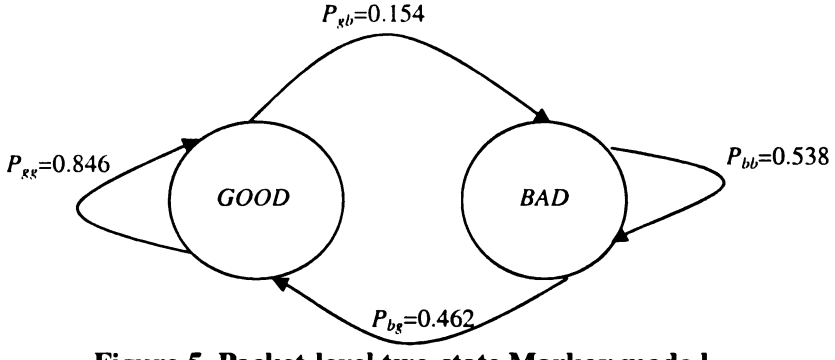


Figure 5. Packet-level two-state Markov mode.l

Clearly, the probability of staying in the good state is quite high. A good (i.e., packets reach the receiver without any errors) transmission is followed by another good transmission 84.6% of the times. However when the model transits to the bad state (i.e., a packet with one or more errors is received), there is a 53.8% probability that the system will stay in the bad state (i.e., next packet will also be corrupted). Hence, the process mostly pro-

duces successful packet transmissions. However, the packet losses are bursty in nature, which implies that whenever there is a packet drop the probability that the next packet will be corrupted (dropped) is quite high. Moreover, the overall packet-loss ratio, computed using (3.2), is 25%.

3.2.2.1. Performance Evaluation of Packet-Level Model

Throughout this work, Entropy Normalized Kullback-Leibler measure (ENK) will be used for statistical performance evaluation of the proposed loss and error models (See (2.6)). Recall that ENK is dependent on the choice of an appropriate random variable to represent the stochastic process. Hence in order to employ the proposed ENK measure, we need to identify appropriate random variables that can efficiently represent the packet loss process.

We employ two random variables to represent the packet loss process: 1) inter-arrival-rate of packet loss' bursts (I), where I takes on non-zero positive integers; 2) burst length[7] of packet losses (B) , where B also takes on non-zero positive integers. Here, it is important to justify the rationale for selecting these two random variables. The first random variable (I) characterizes (inversely) the frequency of occurrences of the packet losses. Small values of I imply large numbers of packet-loss events, and vice versa. As emphasized previously, a packet loss corresponds to a packet with (one or more) corrupted symbols. Thus, the inter-arrival-rate random variable (I) is tantamount to the num-

[7] The term "burst length" represents the number of *consecutive* occurrences of an event. For example, burst length of packet losses represents the number of *consecutive* packet drops.

ber of good packets (i.e., packets with no errors) received between occurrences of bad packets (i.e., packets with errors). In other words, the inter-arrival-rate random variable represents the burst length of good (zero-error) packets. The second random variable (B) represents the length of consecutive packet drops. Bursty packet losses are much more detrimental to a multimedia transmission than isolated losses. Thus we characterize the burstiness of the packet loss process using these two random variables.

At this point, it is important to describe the notation that will be used consistently to evaluate the performance of a model. As specified before, the inter-arrival-rate and the burst-length random variables are represented by I and B respectively. Probability distributions corresponding to actual (source-based[8]) traces are referred to as p_{s_i} or q_{s_j}, where $i \neq j$. The subscript "s" denotes that the random variable was derived from the "source-based" traces. The s_i's represent traces that were collected at different times under similar conditions. In our evaluation, we employ a particular source-based trace as the reference. Thus, a source-based trace (s_i) provides the "reference" probability distribution (p_{s_i}) of the ENK measure. Another source-based trace (s_j) provides the "secondary" probability distribution (q_{s_j}) of the ENK measure. This ENK between source-based traces serves as a performance criterion while evaluating the ENK between a source- and a model-based trace. Let us extend the ENK definition, given in (2.6), to incorporate the new notation,

---

[8] Throughout this document, we refer to the actual (collected) traces as source-based traces since they are generated by the actual random source.

$$ENK\left(p_{s_i}(X)\middle\|q_{s_j}(X)\right)=\frac{D\left(p_{s_i}(X)\middle\|q_{s_j}(X)\right)}{H\left(p_{s_i}(X)\right)} \tag{3.3}$$

where, $D\left(p_{s_i}(X)\middle\|q_{s_j}(X)\right)$ and $H\left(p_{s_i}(X)\right)$ represent the Kullback-Leibler distance (2.5) and the entropy (2.4) of the (source-based) probability distributions derived with respect to the random variable X. The random variable (X) can be either the inter-arrival-rate or the burst-length, i.e., $X = I$ or $X = B$.

We denote the probability distribution of the traces generated by the model as q_m, where the subscript "m" denotes that the random variable was derived from a "model-based" trace. It should be mentioned again that the source-based traces (s_i) were employed to determine the transition probabilities of the two-state model. More specifically, the transition probability matrices of all the source-based traces were generated and the average of these matrices was employed for the model. In accordance with preceding discussion, a source-based trace (s_i) serves as the "reference" probability distribution (p_{s_i}) of the ENK measure. However, the "secondary" probability distribution of the ENK measure is rendered by the model-based trace (q_m). Thus (3.3) becomes,

$$ENK\left(p_{s_i}(X)\middle\|q_m(X)\right)=\frac{D\left(p_{s_i}(X)\middle\|q_m(X)\right)}{H\left(p_{s_i}(X)\right)} \tag{3.4}$$

We present comparison between $ENK\left(p_{s_1}\middle\|q_{s_3}\right)$, $ENK\left(p_{s_2}\middle\|q_{s_3}\right)$, $ENK\left(p_{s_1}\middle\|q_m\right)$ and $ENK\left(p_{s_2}\middle\|q_m\right)$. Note again that $ENK\left(p_{s_1}\middle\|q_{s_3}\right)$ and $ENK\left(p_{s_2}\middle\|q_{s_3}\right)$ provide reference val-

ues against which $ENK(p_{s_1} \| q_m)$ and $ENK(p_{s_2} \| q_m)$ can be evaluated. Since ENK is a non-symmetric measure, special care was exercised in retaining a specific order while computing ENK. Note that when comparing a trace (source- or model-based) to s_1, the probability distribution of s_1 (i.e., p_{s_1}) appears first. Similarly, when comparing a trace (source- or model-based) to s_2, the probability distribution of s_2 (i.e., p_{s_2}) appears first. Table 2 tabulates the ENK of the source- and model-based observations for the two random variables.

X	$ENK(p_{s_1} \| q_{s_3})$	$ENK(p_{s_2} \| q_{s_3})$	$ENK(p_{s_1} \| q_m)$	$ENK(p_{s_2} \| q_m)$
I	0.02535	0.03621	0.006113	0.01883
B	0.03849	0.056467	0.019183	0.04829

Table 2. ENK-Based Performance of the Packet-Loss Model for the Inter-Arrival-Rate and the Burst-Length Random Variables

For both of the random variables, the overhead incurred by the model-based observations is nominal. The overhead is sometimes more than the actual traces. However, the extra overhead incurred by the model is (relatively) meager, e.g., $ENK\left(p_{s_2} \| q_{s_3}\right) - ENK\left(p_{s_2} \| q_m\right) = 0.056467\text{-}0.04829 = 0.008177$. These results clearly depict the appropriateness of the two-state model in approximating the packet loss patterns. Hence, we conclude that a simple two-state (packet-level) model is adequate to capture the loss patterns of an 802.11b network. This model can be used as a generator of packet loss sequences and/or to study the behavior of the channel.

3.3. Discussion

At this point in our work we have established that in a traditional 802.11 environment packet losses are bursty and can be modeled efficiently using a simple two-state Markov model. The 802.11b MAC layer discards packets with errors without regard to the number and location of these errors. However in accordance with the inherent resilience of real-time applications to a certain level of errors, partially damaged packets might be preferable to lost packets. Recently, cross-layer strategies, relaying partially corrupted packets to the real-time application, have shown some promise of improving the bandwidth utilization while maintaining multimedia quality [3], [6]. See Appendix A for detailed discussion on one such cross-layer framework.

Naturally, this raises a fundamental question: Does further analysis and modeling at lower levels of granularity yield any dividend? The answer to this question stipulates a thorough investigation of the tradeoff between lost packets and packets with corrupted bits (i.e., errors). Rationale for further analysis and modeling can be justified by performing byte-level throughput measurement using the partially corrupted packets. Here, byte-level throughput corresponds to the ratio between the total number of transmitted bytes (i.e., corrupted and uncorrupted) and the number of bytes that reach the receiver without any errors (i.e., uncorrupted only). Alternatively, we can extend the throughput definition given in (3.1) to the byte-level,

$$\tau_{byte} = \left(\frac{total\ number\ of\ uncorrupted\ bytes}{total\ number\ of\ transmitted\ bytes} \right) = 1 - P_{byte} \tag{3.5}$$

where, P_{byte} represents the (overall) probability that a byte is corrupted. Byte-level (percentage) throughput for all bitrates is tabulated in Table 3. Comparison of throughputs at both the packet and byte-level is illustrated in Figure 6.

	τ_{byte} (%)
2 Mbps	99.905
5.5 Mbps	99.14
11 Mbps	88.51

Table 3. Byte-Level Percentage Throughput at 2, 5.5 and 11 Mbps

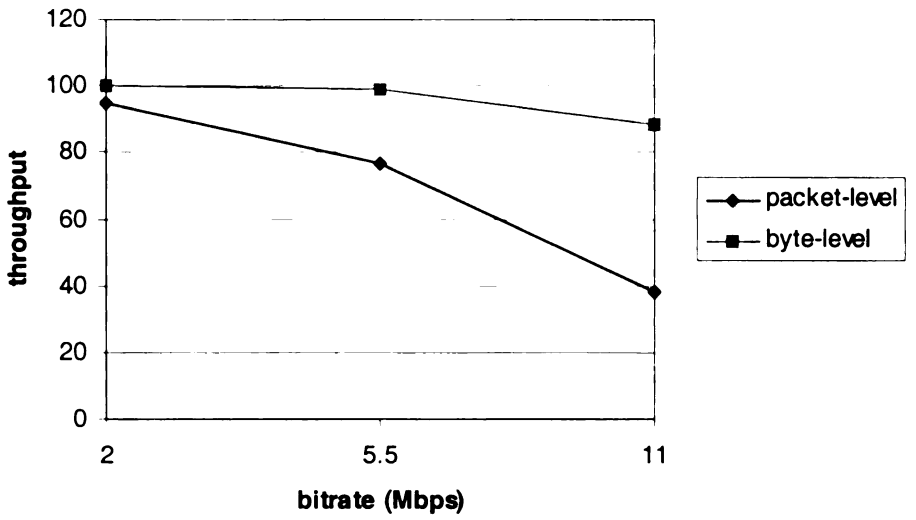


Figure 6. Throughput at the packet- and bit-levels.

Clearly, the byte-level throughput is significantly higher than packet-level throughput for all bitrates. Hence, real-time applications can benefit from partially corrupted packets since such packets contain a reasonable number of good bytes. We, therefore, perform analysis and modeling at the bit-level in the following chapter. This bit-level model can be extended to bytes and in turn packet. Here, it is important to note that higher byte-/bit-

level throughput does not undermine the usefulness of the packet-level model. We will elaborate on this issue further in the following chapter. At this point it suffices to say that the bit-level model is orders of magnitude more complex than the packet-level model. In addition, and as mentioned before, in order to relay partially corrupted packets to the application layer, appropriate modifications have to be introduced at the MAC, network and transport layers. Furthermore, the bit-/byte-level application layer FEC needs to cater for both errors (i.e., corrupted bits/bytes) and erasures (i.e., dropped packets). Due to these pragmatic considerations, some applications might prefer the simpler approach of using a packet-level model.

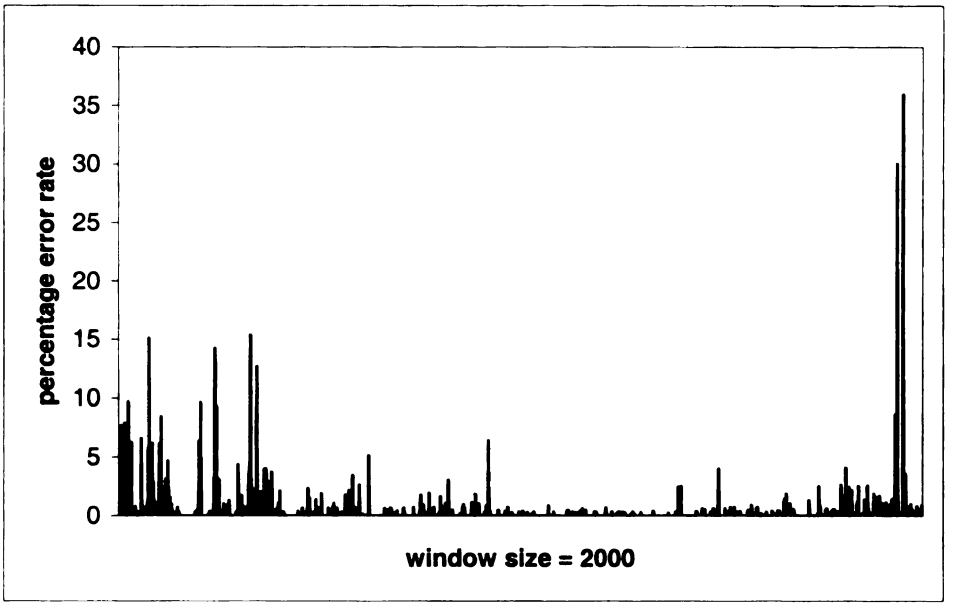
CHAPTER 4

BIT-LEVEL ANALYSIS AND MODELING

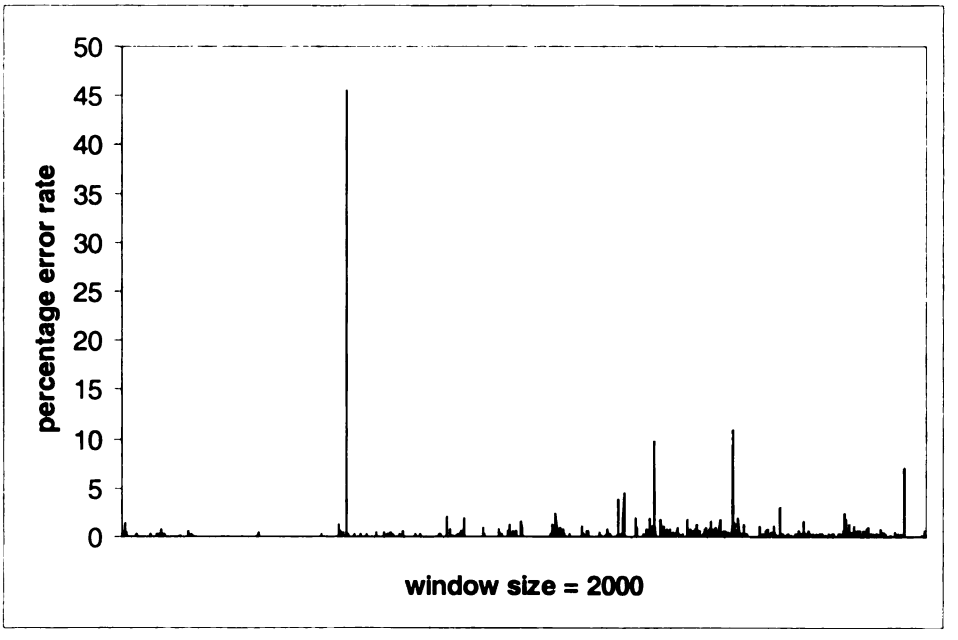
Some preliminary analysis in the previous chapter has motivated the need for analysis at finer levels of granularity. All following work will focus on bit-level. It is important to note that a bit-level model can be used to generate traces with bit-level granularity, and consequently, can be used to generate coarser-level traces such as byte-level or packet-level traces. As mentioned before, we focus on the 5.5 Mbps case. Nevertheless, the overall analysis and modeling approach presented in this chapter is also applicable to other bitrates.

4.1. Stationarity

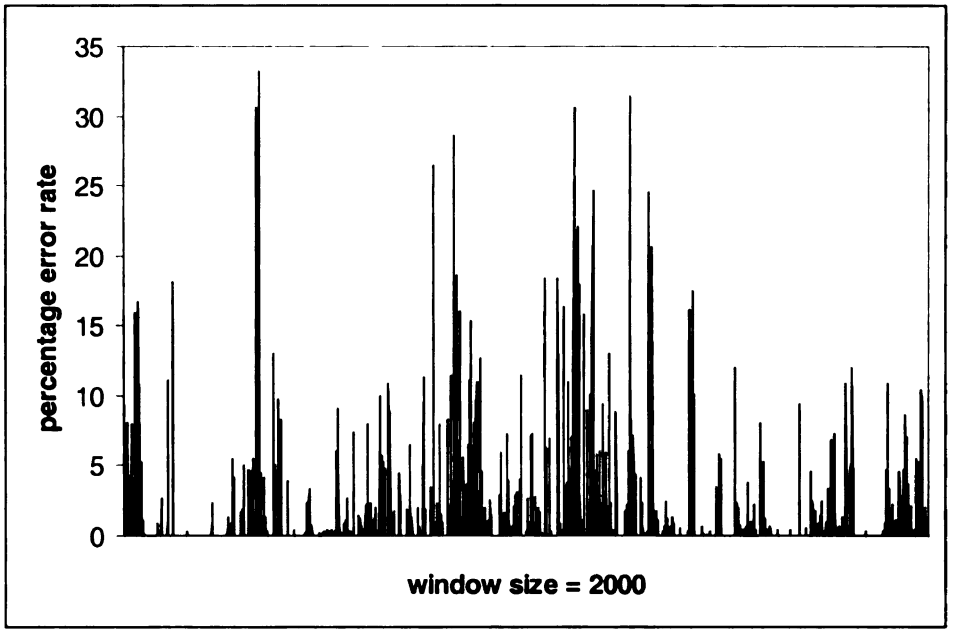
A random process X_n is *strictly stationary* if the distribution of $(X_{p+1}, X_{p+2}, ..., X_{p+k})$ is the same as that of $(X_1, X_2, ..., X_k)$ for all p and k. Testing a time series for stationarity is theoretically impossible. In order to determine the stationarity characteristics of our data, we generated the first moment (i.e., the average number of bad bits) over varying (non-overlapping) window sizes. We declare the process as *loosely stationary* if the mean bit-error-rate of the process remains relatively constant over time. An example of the error-rate for three traces using a window size of 2000 bits is outlined in Figure 7. Results for other window sizes exhibited similar properties.


40
35
30
25
20
15
10
5
0
percentage error rate
window size = 2000

(a)


50
45
40
35
30
25
20
15
10
5
0
percentage error rate
window size = 2000

(b)



(c)

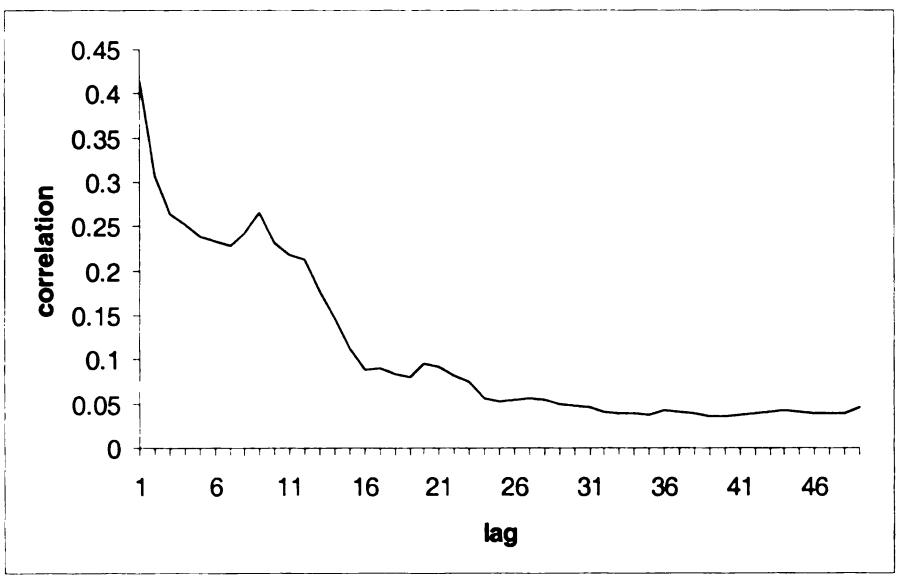
Figure 7. Mean bit-error-rate for a window of 2000 bits.

The sharp variations in the average bit-error-rate outline the non-stationary behavior of the traces. This experiment was repeated for all (5.5 Mbps) traces and it was observed that the maximum error-rate varied between 30% and 47% for different traces. Such erratic behavior of the error traces rendered stationary data analysis techniques ineffectual. However, standard analysis methods can be employed to determine temporal dependencies in the traces. This analysis is presented in the following section.

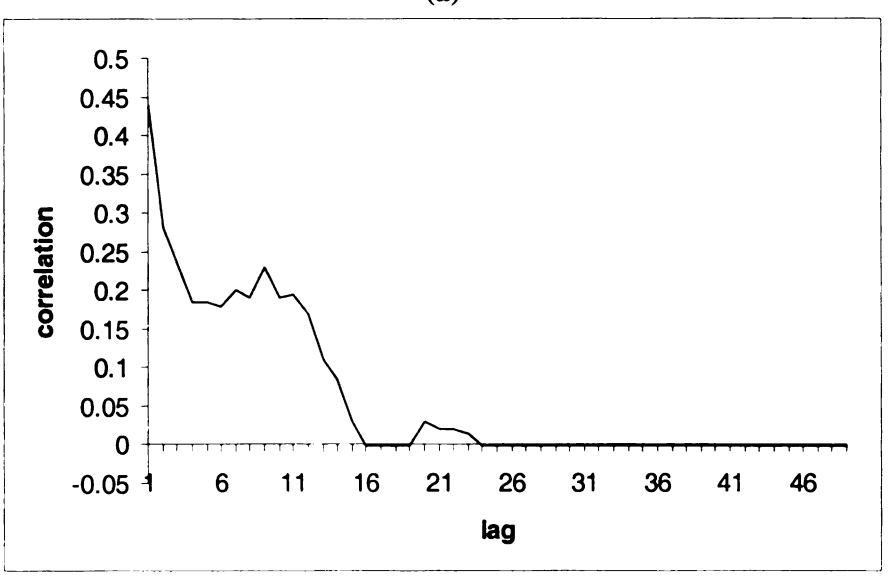
4.2. Autocorrelation of Bit-Level Traces

Autocorrelation can be employed to determine the memory length of a Markov chain (see Section 2.2). In this section we perform autocorrelation analysis to formulate the basis of our first modeling attempt in the following sections of this chapter. The bit-level

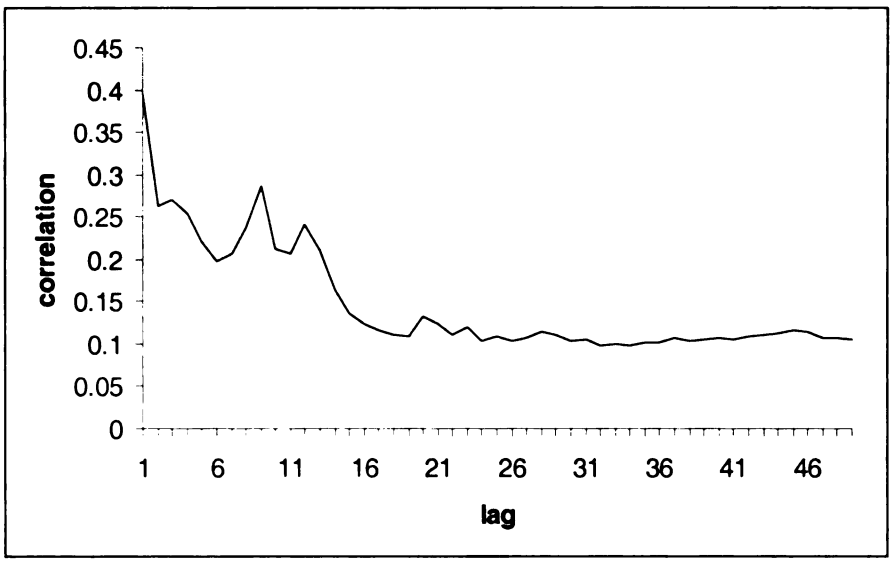
traces are represented as a binary time series $\{x_i\}_{i=1}^{n}$, where $x_i \in \{-1,1\}$ and n is the length of the time-series.



(a)



(b)



(c)

Figure 8. Autocorrelation of the binary error traces.

It is clear from Figure 8 that, overall, the correlation is a decaying function and drops down to a (relatively) insignificant value after a certain lag. From the examples provided in Figure 8, we assume that the memory length is determined by the lag beyond which the correlation drops below 0.15 and stays within that bound [23]. Thus the memory lengths for the traces of Figure 8(a) (b) and (c) are 13, 12 and 14 respectively. We use memory length 14 as the maximum order of our Markov chain.

4.3. Full State Markov Chain

For a memory length k, the states of the Full State Markov (FSM) Chain correspond to the 2^k different possible combinations of k consecutive bits. Transition probabilities between states are computed by the number of times a bit-pattern $\bar{x} = [x_1 x_2 \ldots x_k]$ is followed by another bit-pattern $\bar{y} = [y_1 y_2 \ldots y_k]$ in the time series. In order to present an ex-

ample of the FSM we represent the possible binary values of the time series as $x_t \in \{0,1\}$. However, all preceding and subsequent sections strictly follow the previously defined representation (i.e., $x_t \in \{-1,1\}$). A sliding window was used to compute the transition probability matrix. Due to the sliding window, the transition possibilities between the states were restricted. An example given in Figure 9 clearly demonstrates this observation. A memory length of four is used in this example. The current state is $(0110)_2 = (6)_{10}$ and, as the window slides, the 0 in the most significant bit position will be dropped and a bit will be added to the least significant bit position. Since the data are binary, the chain can transit to either $(1100)_2 = (12)_{10}$ or $(1101)_2 = (13)_{10}$. This observation can be extended to claim that any current state can jump to only two possible next states (current state inclusive) at each transition. Hence, the transition probability matrix computed by this procedure will be sparse. At this point we expect the FSM to perform adequately since it completely captures the frequency of occurrence and location of each possible bit pattern. This performance evaluation is provided in the next section.

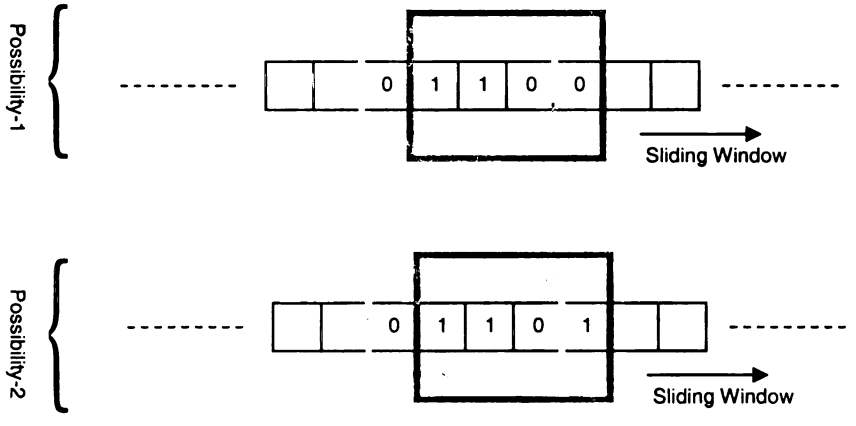


Figure 9. Transition possibilities for a sliding window scenario (memory length=4).

4.3.1. Performance Evaluation of FSM

We generated Markov chains with varying memory lengths (k) such that each chain had 2^k states, where k = 1, 2, ... , 14. The number of states in the FSM increases exponentially with the increase in memory length. The total number of states directly corresponds to the overall complexity of the model. For example, the transition probability matrix for memory length of 14 has $2^{14} \times 2^{14}$ (=268435456) entries (32-bit precision each). Storage, random access and computation complexity of such data is a nontrivial task. Due to these pragmatic considerations, previous (related) studies employed a manageable (relatively small) order of the Markov chain [1], [9], [10].

In order to efficiently and accurately represent the transition probability data, we examined the transition probability matrices for bit-patterns that never occur in the collected traces. These bit-patterns result in an "unused state". In other words, such states result in zero columns in the transition probability matrix. An all-zero column implies that the probability of jumping to that state from any state is zero. The total number of unused states for each order is enumerated in Table 4. We observed that the number of unused states grew as the order of the Markov chain increased. For example, in case of a 2^{14} state model, approximately 70% of the states are never used. We lay special emphasis on this observation since the total number of states directly corresponds to the complexity of the model. In addition, the number of states plays an instrumental role in evaluating the performance of this model (as illustrated in the following sections). Therefore, all following observations will strictly employ the "used states" only. It is still not possible to enumerate the transition probability matrices for each memory length. We, however, provide ENK-based performance of each model.

Total (Initial) States	Unused States	Used States
2^1 (=2), ..., 2^8 (=256)	0	2, 4, ..., 256
2^9 (=512)	1	511
2^{10} (=1024)	33	991
2^{11} (=2048)	309	1739
2^{12} (=4096)	1388	2708
2^{13} (=8192)	4395	3797
2^{14} (=16384)	11536	4848

Table 4. Unused States in 2^i State Chains, i = 1, 2, ... , 14

We again employ the same random variables, i.e., inter-arrival-rate (I) and burst-length (B), to evaluate the performance of the bit-level models. However in this (bit-level) scenario, I and B represent the inter-arrival-rate and burst-length of the bad bits respectively. More specifically, I characterizes the frequency of occurrence of the bad bits, that is, the number of good bits received between occurrences of bad bits. In other words, the (bit-level) inter-arrival-rate random variable represents the burst length of good bits. B represents the length of consecutive bad bits. We emphasize again that the effect of bursty errors on the multimedia quality is much profound than isolated errors. Thus we characterize the burstiness of the bit-error patterns using these two random variables.

In accordance with the previous performance evaluation example (Section 3.2.2.1), probability distributions corresponding to actual (source-based) traces are referred to as p_{s_i} or q_{s_j} ($i \neq j$). The subscript "s" denotes that the random variable was derived from the "source-based" traces. We, however, extend our notation for the probability distribution of the traces generated by the model as q_{m_k}, where the subscript "m" denotes that the random variable was derived from a "model-based" trace and "k" represents the number of states in the Markov chain. With the exception of this added notation, the rest of the ENK definition remains the same as described in (3.3) and (3.4). Due to the unrea-

sonable storage, memory and complexity requirements of high-order (bit-level) models, only one source-based (actual) trace was employed to generate the model transition probability matrices. This (source-based) trace is, henceforth, referred to as s_1. Table 6 outlines the ENK-based performance evaluation of the FSM.

	$X = I$	$X = B$
$ENK(p_{s_1} \| q_{s_2})$	0.011397	0.002276
$ENK(p_{s_1} \| q_{s_3})$	0.008645	0.00291
$ENK(p_{s_1} \| q_{m_2})$	0.5238	0.0105
$ENK(p_{s_1} \| q_{m_4})$	0.3974	0.009104
$ENK(p_{s_1} \| q_{m_8})$	0.3037	0.010113
$ENK(p_{s_1} \| q_{m_{16}})$	0.2346	0.006378
$ENK(p_{s_1} \| q_{m_{32}})$	0.1851	0.004868
$ENK(p_{s_1} \| q_{m_{64}})$	0.1521	0.005918
$ENK(p_{s_1} \| q_{m_{128}})$	0.1373	0.00378
$ENK(p_{s_1} \| q_{m_{256}})$	0.1048	0.003452
$ENK(p_{s_1} \| q_{m_{511}})$	0.06993	0.00372
$ENK(p_{s_1} \| q_{m_{991}})$	0.0605	0.0031
$ENK(p_{s_1} \| q_{m_{1739}})$	0.05831	0.002671
$ENK(p_{s_1} \| q_{m_{2708}})$	0.0584	0.002328
$ENK(p_{s_1} \| q_{m_{3797}})$	0.0532	0.001994
$ENK(p_{s_1} \| q_{m_{4848}})$	0.05163	0.001531

Table 5. ENK-based Performance Evaluation of the 2^i-state FSM, $i = 1, 2, \ldots, 14$, for the Inter-Arrival-Rate and Burst-Length Random Variables

Clearly, the FSM performs remarkably for the burst-length random variable. Note that even smaller order chains perform adequately with the source coding overhead less

than 1% for all cases. However, the inter-arrival rate random variable incurs profound overhead for smaller order chains. For example, the 2-state chain renders an overhead of approximately 50% and is, therefore, not a viable option. Nevertheless, as we move to higher order chains, the overhead decreases and drops to a reasonable level[9] at and after the 511-state model. The best performance is provided by the 4848-state chain which incurs an overhead of 5.163% and 0.1531% for the inter-arrival rate and burst-length random variables. Thus we conclude that the (high order) FSM renders a very good bit-level approximate of the 802.11b channel.

4.3.2. Discussion

The results presented in this section depict the efficacy of the FSM to capture the bit-level channel behavior. It was illustrated, with the help of the ENK performance measure, that the FSM adequately approximates the occurrence of good and bad bits, in other words, the inter-arrival-rate and the burst-length. However, the complexity of this model is unreasonably high thereby rendering it impractical for most real-time applications.

4.4. Zero-Crossing Markov Chain

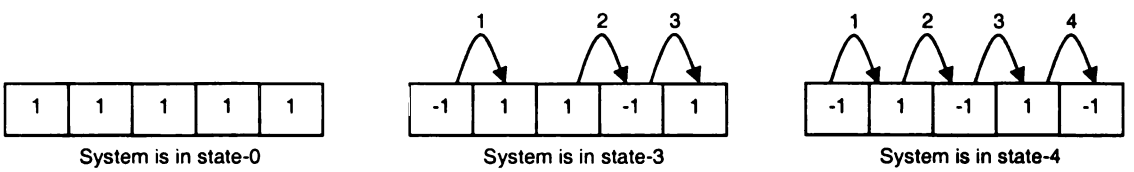
The analysis provided in the previous section clearly illustrates a significant improvement potential. Some previous studies have suggested the use of variable memory length Markov processes to efficiently utilize the context-dependent nature exhibited by

---

[9] Since a very low overhead might yield "data overfitting", we assume that any overhead less than 10% is acceptable.

most natural sequences [14], [28], [29]. We employ the inherent characteristics exploited by the FSM to render a simpler solution. Here, it should be noted that the two basic characteristics that the FSM captured within the memory window are: 1) number of bad bits; 2) position of bad bits. Based on this observation, we propose a new approach for state definition, which captures both of these characteristics while providing a low-complexity alternative to the FSM 2^k state mapping.

We define the states of this new Markov chain as the number of zero crossings (i.e., transitions from -1 to 1 and 1 to -1) within a memory window of length k. We refer to this model as the *Zero-Crossing Markov Chain*. Examples to elaborate the ZCM state allocation are provided in Figure 10. In essence, each ZCM state represents the number of bit toggles occurring within the memory window (k). Thus, there are a total of k possible states, i.e., 0, 1, 2, ..., k-1.



Example-1 **Example-2** **Example-3**

Figure 10. Example states in a ZCM of memory length 4-bits.

This state definition captures both the characteristics mentioned previously. More specifically, this model captures the total number of bad bits and the (relative) position of bad bits within a memory window. Furthermore, this choice of states reduces the exponentially increasing number of states to a linear increase. Hence, the total states required to represent a 14-order model with a ZCM is 14 instead of 2^{14} for the FSM (see Figure 11). This is an order of magnitude decrease in the number of states which, in turn, renders

outstanding reduction in the complexity of the model. This represents one of the key advantages of the proposed model: the number of states is the same as the size of the memory window. Hence, the proposed model captures the desired temporal dependency while maintaining very low complexity. As demonstrated below, the performance of this model is also excellent (based on the ENK measure).

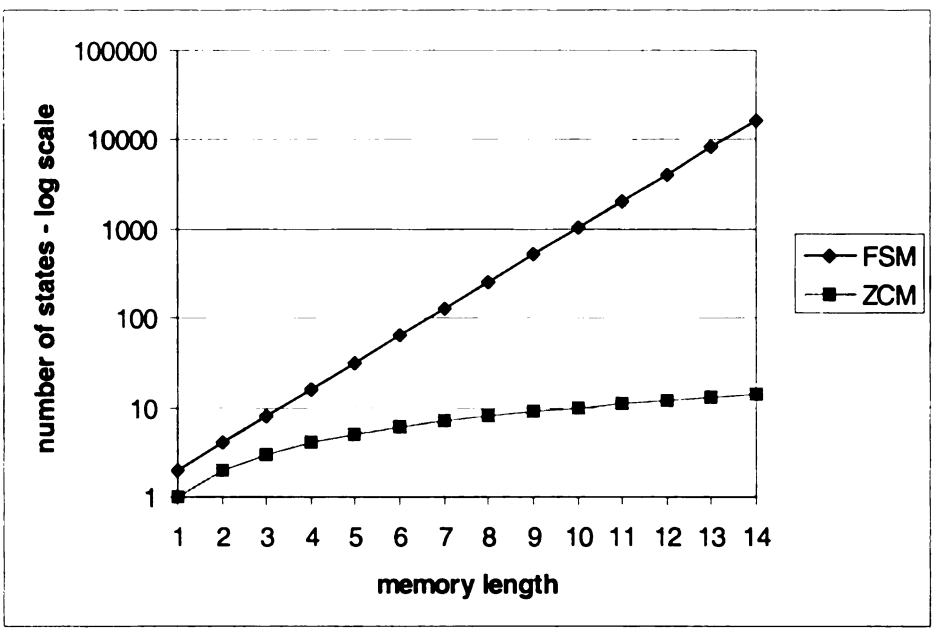


Figure 11. The total number of states as a function of the memory length for FSM and ZCM.

Although the particular choice of ZCM states has reduced the model complexity tremendously, the performance of this model with respect to FSM needs to be investigated. The following section compares the performance of ZCM with FSM in order to determine the relative efficacy of ZCM in approximating the 802.11b (bit-level) channel.

4.5. Performance Comparison of FSM with ZCM

Figure 12, Figure 13 and Figure 14 compare the performance of FSM and ZCM for the inter-arrival-rate and the burst-length random variables. Clearly, the performance of ZCM is comparable to the FSM for a certain number of states. However, the corresponding number of states for FSM is much higher, for example, the 14-state ZCM has an ENK of 0.058623 versus 4848-state FSM's ENK of 0.05163 for the inter-arrival rate random variable. For the burst-length random variable, the performance of FSM is better for models with number of states greater than 16. However, the performance improvement is meager, for example, 14-state ZCM has an ENK of 0.007839 versus 4848-state FSM's ENK of 0.001531. Hence by incurring a performance overhead of 0.0063, the ZCM reduces the number of states by 4834. Table 6 outlines the complete performance of ENK for the inter-arrival rate and the burst-length random variables. The ZCM reduces the (classical) exponential increase in the order-complexity to a linear increase. The ZCM renders FSM-like performance while rendering orders of magnitude reduction in model complexity. This exceptionally good performance of the ZCM substantiates that fact that it offers a viable (low-complexity) alternative to the FSM.

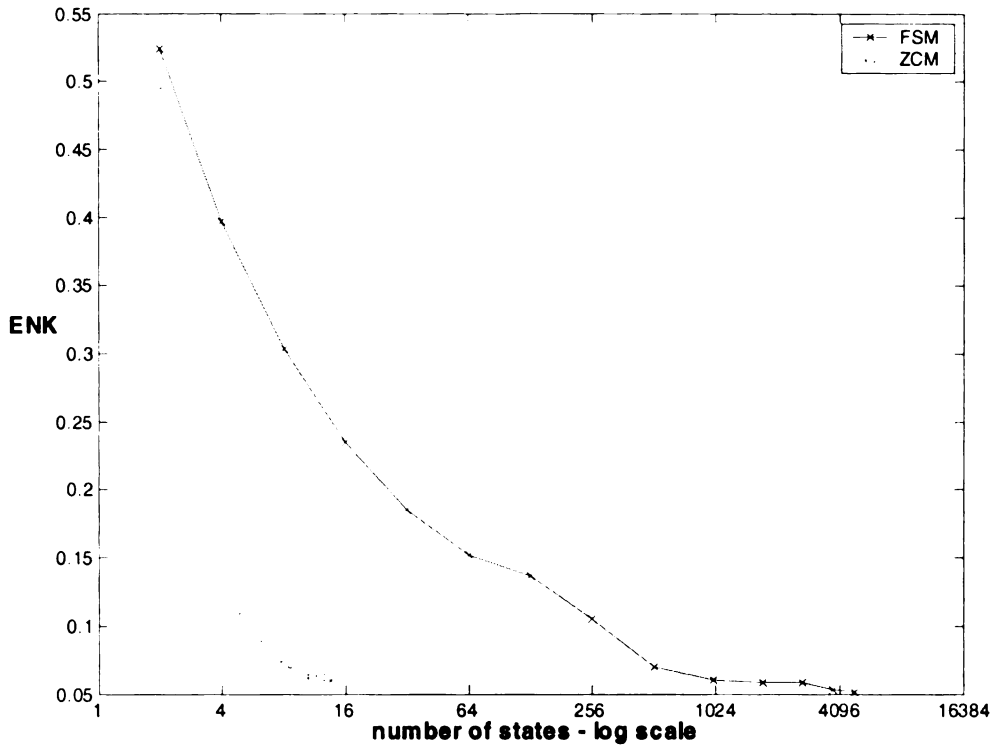


Figure 12. ENK-based performance comparison of FSM with ZCM for inter-arrival rate random variable.

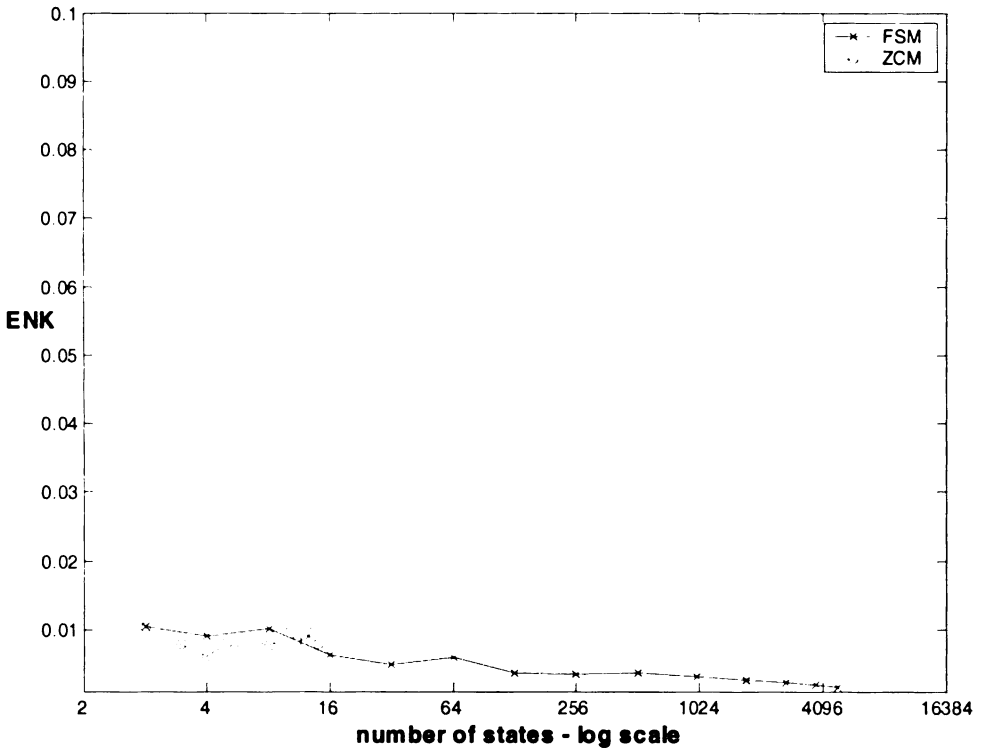


Figure 13. ENK-based performance comparison of FSM with ZCM for burst-length random variable.

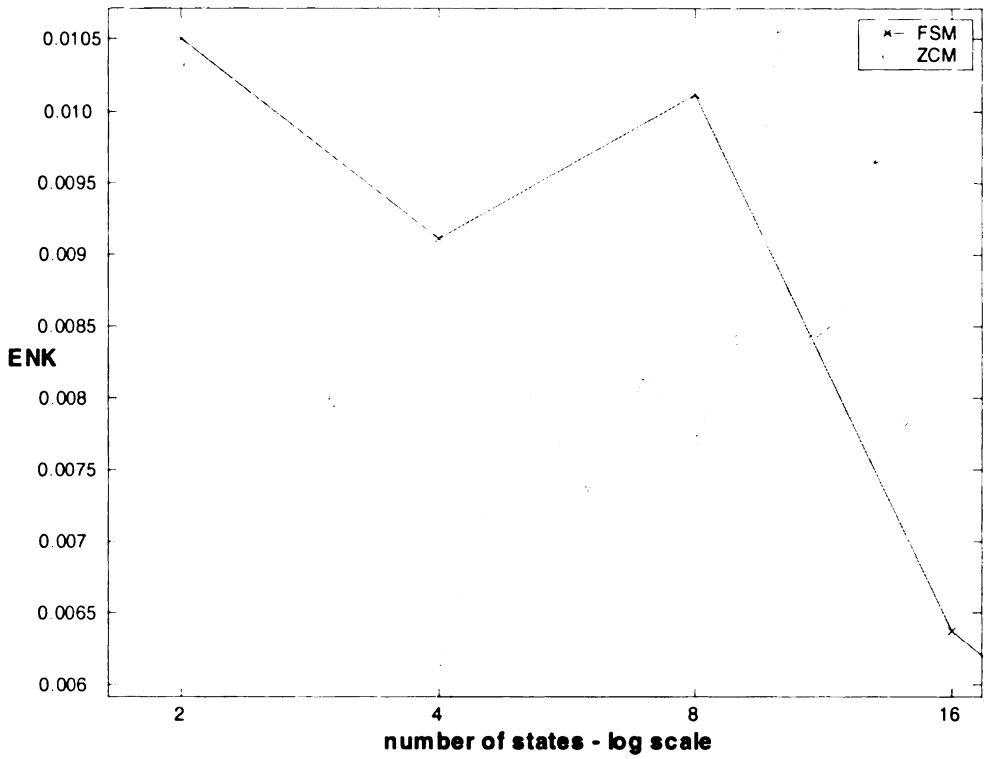


Figure 14. Magnified ENK-based performance comparison of FSM with ZCM for burst-length random variable

	$X = I$	$X = B$
$ENK\left(p_{s_1} \middle\| q_{s_2}\right)$	0.011397	0.002276
$ENK\left(p_{s_1} \middle\| q_{s_3}\right)$	0.008645	0.00291
$ENK\left(p_{s_1} \middle\| q_{m_2}\right)$	0.49867	0.010354
$ENK\left(p_{s_1} \middle\| q_{m_3}\right)$	0.18388	0.007964
$ENK\left(p_{s_1} \middle\| q_{m_4}\right)$	0.13433	0.006107
$ENK\left(p_{s_1} \middle\| q_{m_5}\right)$	0.10586	0.008118
$ENK\left(p_{s_1} \middle\| q_{m_6}\right)$	0.091524	0.007347
$ENK\left(p_{s_1} \middle\| q_{m_7}\right)$	0.078734	0.00815
$ENK\left(p_{s_1} \middle\| q_{m_8}\right)$	0.071516	0.007714
$ENK\left(p_{s_1} \middle\| q_{m_9}\right)$	0.066809	0.008408
$ENK\left(p_{s_1} \middle\| q_{m_{10}}\right)$	0.062289	0.010578
$ENK\left(p_{s_1} \middle\| q_{m_{11}}\right)$	0.061995	0.008403
$ENK\left(p_{s_1} \middle\| q_{m_{12}}\right)$	0.062012	0.00858
$ENK\left(p_{s_1} \middle\| q_{m_{13}}\right)$	0.061978	0.009674
$ENK\left(p_{s_1} \middle\| q_{m_{14}}\right)$	0.058623	0.007839

Table 6. ENK-based Performance Evaluation of *i*-state ZCM, $i = 2, \ldots, 14$, for the Inter-Arrival-Rate and Burst-Length Random Variables

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

In this work, we provided analysis and modeling of errors and losses over 802.11b wireless LANs. Our focus throughout the thesis has been the feasibility of supporting high-bitrate (real-time) applications. First, we presented measurements of the MAC layer loss patterns over an 802.11b network at 2, 5.5 and 11 Mbps rates. The throughput at 2 Mbps is very high which deems further analysis at this bitrate inconsequential. Throughput at 11 Mbps was very low thereby rendering it unsuitable for high-bitrate applications. This observation is easily explained since the robustness of the 802.11b physical layer reduces with the increase in bitrate. Hence, we focused our attention on the 5.5 Mbps case which, in addition to its (relatively) high throughput, exhibits potential for supporting high-bitrate applications.

We perform preliminary packet-level throughput analysis on the 5.5 Mbps traces. It was observed that the packet losses are bursty in nature. Therefore, we employed a simple two-state Markov model to approximate the packet drop bursts. We showed that this model provided very promising results. The employed performance criterion quantifies that the (overall) ENK-based overhead incurred by this model is negligible.

It was observed that the throughput at the byte-level is much higher than the packet-level throughput. This is easily justified since the 802.11b MAC layer drops all packets with errors irrespective of the number or location of such errors. Most of these dropped

packets have substantial number of good (uncorrupted) bytes. We, therefore, extend our analysis and modeling to lower levels of granularity. However, it should be mentioned that the byte-/bit-level framework can only be effective when employed with appropriate cross-layer schemes that allow partially corrupted packets to reach the application layer (See Appendix A).

At the bit-level, we first employed a (classical) 2^k -state model to approximate the channel behavior. We showed that the model performed adequately at orders higher than 10 with the best performance rendered by the order-14 model. However, this model comprises of 2^{14} states (4848 used states) thereby stipulating very high complexity. Such considerations render this model impractical. Hence, we propose a Zero-Crossing Markov chain in which the states are defined as the number of bit transitions within a memory window. This model reduces the exponential increase in complexity to a linear increase. Finally, we show that the performance of this model is comparable to the 2^k -state model.

During the course of this work we identified some future directions. We are currently developing a more comprehensive set of error traces. This set will comprise of be error traces collected at all bitrates for varying client positions and network configurations. The effect of interference will be further distinguishable with this complete data set. We would also like to investigate the error patterns for mobile (ad hoc) clients. This is an active area of research and our preliminary analysis depicts that the position of the client plays an instrumental role in determining the overall throughput of the network.

Moreover, cross-layer strategies that can employ the information provided by the modeling at the link layer need further investigation. In our previous work, we suggested

some simple, but useful, cross-layer strategies that can be employed to render high-bitrate multimedia over wireless networks [3]. These strategies can be supplemented with modeling and subtle modifications at the link and transport layers in order to boost the overall performance of the system (See Appendix A).

We are also investigating other Markov-based modeling techniques that can provide a suitable alternative to the approaches presented in this work. Two most prominent approaches under consideration are based on Hidden and Hierarchical Markov models [11], [30]. For both these modeling approaches, error and (almost) error-free parts of the traces need to be isolated so that analysis can be performed on these trace segments separately. We proposed a state demarcation heuristic in [11] which employed the relative lengths of the good and bad bursts to identify state boundaries. We believe that the algorithm proposed by Rabiner *et al.,* [31] can provide very good results in this context. The performance comparison of these algorithms and the respective models will be provided in future work.

APPENDIX A

FEASIBILITY OF A CROSS-LAYER FRAMEWORK TO SUPPORT HIGH-BITRATE REAL-TIME MULTIMEDIA

The advent of wireless networks has advanced real-time multimedia communication into a novel realm. The wireless medium, due to its inherent vulnerability and physical characteristics, is more error prone than most of the contemporary (wired) media. Nevertheless, migration of technologies from the wired to wireless domain is currently underway. One of the key challenges in this context is the delivery of real-time applications over the wireless medium. The promise of real-time multimedia over the wired Ethernet is often attributed to the simplicity and pragmatism of UDP/IP protocol suite. However, this protocol suite was designed for considerably low error-rate environments as opposed to wireless networks. In this regard, a positive aspect of real-time applications is their inherent tolerance to a certain level of corrupted and/or dropped packets. This characteristic of real-time applications motivated the development of new methods and protocols for wireless networks. For example, a new transport-layer protocol, which is tailored for real-time applications over error-prone networks, has been proposed recently [6], [7], [8]. This protocol, known as UDP Lite, relies on the premise that multimedia applications can tolerate, and therefore should, receive corrupted packets. Naturally, this UDP-Lite-based

framework makes it necessary for the MAC layer to forward corrupted packets to the higher layers. Consequently, developers of wireless multimedia applications are faced with two (high level) options: (1) employing current wireless MAC functions, which include dropping of corrupted packets and attempting to recover these packets through retransmissions, in conjunction with the traditional UDP protocol; or (2) allowing the MAC layer to pass corrupted packets to a UDP Lite-type transport layer, which, in turn relies on the application layer to handle the corrupted data. A key advantage of the second option is the potential increase in the, potentially corrupted, throughput of the end-to-end transport-layer packets. However, these "high-throughput" packets contain corrupted data.

Therefore, and based on the above two options, three key questions can be raised: (1) How much improvement in the "throughput" is actually being gained by using the new UDP Lite-based framework? (2) How badly corrupted these "high-throughput" packets are? In other words, how much and what type of error patterns are observed in these packets? (3) As a consequence of the first two questions, what level of application-layer loss-protection, error-concealment and/or error-correction would be required to achieve acceptable quality under the two protocol-stack variants? Here, we address the above questions in the context of high-bitrate[10] multimedia applications[11] over 802.11b wireless LANs.

---

[10] Since we conducted our study and analysis at the full-rates provided by the different modes at the physical layer, our definition of "high-bitrate" follows the 2, 5.5, and 11 Mbps bitrates supported by 802.11b.

At this point, it is important to outline how these three questions can be answered. The first question, which has been addressed partially by previous studies (see for example [7], [10]), can be answered through a set of experiments and related analysis, which will be provided in the following sections. Answering the second question requires a thorough analysis and some modeling of the error patterns at the MAC layer (i.e., errors that are not corrected by the physical layer). This question has, for the most part, been answered by the MAC layer analysis and modeling provided in this thesis. However in the context of this discussion, a little insight into byte-level error patterns is required since most error-recovery schemes operate on bytes. We, therefore, provide byte-level analysis by fitting probability distributions for the byte-level burst length random variable.

Addressing the third question is naturally more challenging than the first two questions, as it depends on a wide spectrum of objective and subjective evaluations of a variety of multimedia applications and corresponding error protection and concealment algorithms. For example, some applications targeted for wireless networks may take advantage of better error resilience features, such as Reversible Variable-Length Coding (RVLC) that is supported in the MPEG-4 video standard [3]. Other examples include a range of (video) error concealment algorithms and/or FEC methods [40], [41]. Due to this wide spectrum of methods and algorithms, which can influence the answer to the above third question, we had no option but to focus on one important (sub-) question of this

---

[11] An example of real-time multimedia is MPEG-4 video [3], which we use occasionally to illustrate some visual simulation results in support of our findings.

spectrum: What is the amount of overhead that is needed at the application layer to achieve different levels of lost- and corrupted-packet recovery for the two protocol-stack scenarios? The answer to this question provides a worst-case measure of the level of overhead needed to correct/recover a desired level of corrupted/dropped packets for both UDP and UDP Lite. As explained later, this leads to an important conclusion about the viability of multimedia applications at rates higher than 2 Mbps under realistic settings.

Our decision to focus on high-bitrate multimedia has been mainly influenced by the fact that emerging wireless LAN standards are designed to support unprecedented high bitrates. In particular, 802.11 LANs are finding their way into homes and businesses. Consequently, it is quite feasible that in the near future these LANs could utilize unicast and/or multicast frameworks to provide real-time television-like services to large numbers of users. As explained above, evaluating this feasibility requires a methodical understanding of the error and loss performance at different layers of the protocol stack that is above the physical layer.

A.1 Related Work

Many studies have attempted to characterize errors in wireless networks. For example, Ludwig *et al.* [9] analyze block erasure traces in cellular networks. Improvement of TCP performance over wireless links has also been under study by several researchers (e.g, [32], [33]). Recent years have seen a tremendous increase in the demand for streaming real-time applications. In order to provide television-like services, efficient multimedia coding on reliable multicast networks has been an area of active research [34], [35], [36]. With wireless networks fast becoming ubiquitous, an imperative direction is to tai-

lor future applications and protocols to adapt accordingly. Some recent works have explored the area of real-time communication over 802.11 networks (see for example [37], [38]). However, we believe that the impact of such networks on high-bitrate multimedia needs further investigation.

Despite the robustness of the 802.11 physical layer, some of the errors are propagated to the link layer. These errors are detected using the FCS and the link layer discards such frames with no regard to the number and location of the errors [15], [16]. A scheme, suggested in [6] tried to address this issue by making adjuZCMents to the protocol stack at the transport and link layers. This variant, which is known as UDP Lite, is a lightweight version of traditional UDP for real-time applications over wireless networks. The protocol allows for delivery of partially damaged packets to the application layer by ignoring errors in the application layer payload. UDP Lite relies on the premise that real-time applications often prefer partially damaged packets over lost packets. It, therefore, allows for checksum on the transport and application layer headers while ignoring checksum on the actual payload. Each packet is divided into a "sensitive" and "insensitive" part. The sensitive part, starting from the transport layer header, is covered by the partial checksum. The "Length Field" in UDP header is utilized to specify the partial checksum length (referred to as "Coverage" in the UDP Lite header)[12]. Figure 15 outlines the UDP Lite headers.

---

[12] Traditionally, the "Length Field" specifies the length of UDP packet (UDP header+payload) [39]. A potential problem arises when UDP Lite uses this field to specify "Coverage", since the receiver uses UDP

One fundamental problem with this approach is that the link layer drops frames with errors before they reach the transport layer. To cater for this problem, Larzon *et al.* [6] suggested that the device driver at the receiver be modified to ignore checksum errors for packets carrying UDP Lite (real-time) traffic. This modified wireless MAC layer is referred to as *MAC Lite*[13]. This modified protocol framework raises a key question: What is the efficacy of MAC Lite/UDP Lite protocol stack when compared with a UDP-based stack for high-bitrate applications? The following discussion addresses this question thoroughly.

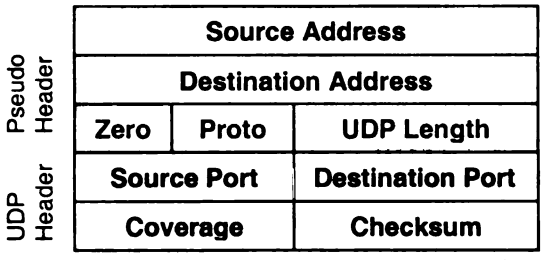


Figure 15. The UDP Lite headers [6].

A.2 Packet-Level Throughput Analysis

The inherent error/loss tolerance of real-time multimedia applications provides a foundation for UDP Lite-like protocols. This section answers the first question raised at

---

packet length in transport layer checksum verification. To address this issue, UDP Lite calculates the UDP packet length from the IP header.

[13] In [6] this link layer strategy was referred to as PPP Lite since Point-to-Point Protocol was being employed for their observations.

the start of this appendix, i.e., how much improvement in the "throughput" is actually being gained by using the new UDP Lite-based framework? Throughout this discussion, the term "throughput" refers to the number of MAC/UDP packets (error-free or corrupted) that are passed by the MAC-/transport-layer to the higher layers of the protocol stack. Before addressing the question at hand, we briefly discuss related 802.11b MAC layer functionality and its scope and application in this study.

The 802.11b standard requires the MAC layer to verify a 32-bit CRC (referred to as FCS) on each received frame. Corrupted MAC frames fail the FCS and are dropped regardless of the number and location of these errors. Due to the inherent lossy nature of the wireless medium, real-time applications might prefer corrupted packets as opposed to dropped packets. As mentioned previously, real-time applications can tolerate corruptions to some degree and error resilience features of some modern-day real-time applications further boost performance in such an error-prone scenario. Furthermore, retransmission-based recovery of corrupted packets is not a viable option in some key emerging multimedia applications (e.g., multicast video). Therefore, we performed all analysis while disabling the MAC-layer retransmission functionality at the server.

We evaluated different combinations of MAC and transport layer variants i.e., traditional 802.11b protocol stack (using 802.11b MAC with UDP), MAC Lite with UDP, and MAC Lite with UDP Lite. The fourth combination (i.e., 802.11b MAC with UDP Lite) provides the same results as 802.11b MAC with UDP, since all frames with errors are dropped at the MAC layer, and therefore, corrupted packets never reach the UDP Lite-based transport layer. For all UDP Lite examples, we performed checksum only on the UDP headers without any assumption about the real-time application headers. Thus, the

application is solely responsible for handling (i.e., decoding or discarding) the corrupted data. Here, it is noteworthy that the traces employed for this feasibility study are different from the traces used throughout the thesis. The traces used for modeling and analysis throughout the thesis are "on-average" traces, that is, on-average such error rates are observed at different stationary client position across two walls. On the other hand, traces employed in this appendix are "worst-case" traces, that is, these traces have (relatively) higher error rate. Rationale for this fact is justified by the fact that we want to evaluate the performance of this cross-layer framework for the worst-case (unicast/multicast) transmission.

2.5.1. MAC Layer Throughput Analysis

As mentioned before, MAC layer "throughput" corresponds to the number of packets that are getting relayed to the network layer i.e., ratio of the number of packets received by the MAC layer to the number of packets received by the network layer. Since the 802.11b MAC drops all corrupted frames, each such packet adversely impacts the throughput. This reduction in throughput propagates to the application layer and, therefore, in a retransmission-less scenario could drastically degrade the quality of perceived media. In this section we evaluate the 802.1b MAC layer throughput at 2, 5.5 and 11 Mbps, and suggest remedies to mitigate the throughput reduction experienced due to channel-induced errors.

The maximum packet loss burst at 2 Mbps is two packets long and the throughput is approximately 100%. Such high-quality performance construed further investigation at this bitrate insignificant. Nevertheless, analysis at higher bitrates is essential to evaluate

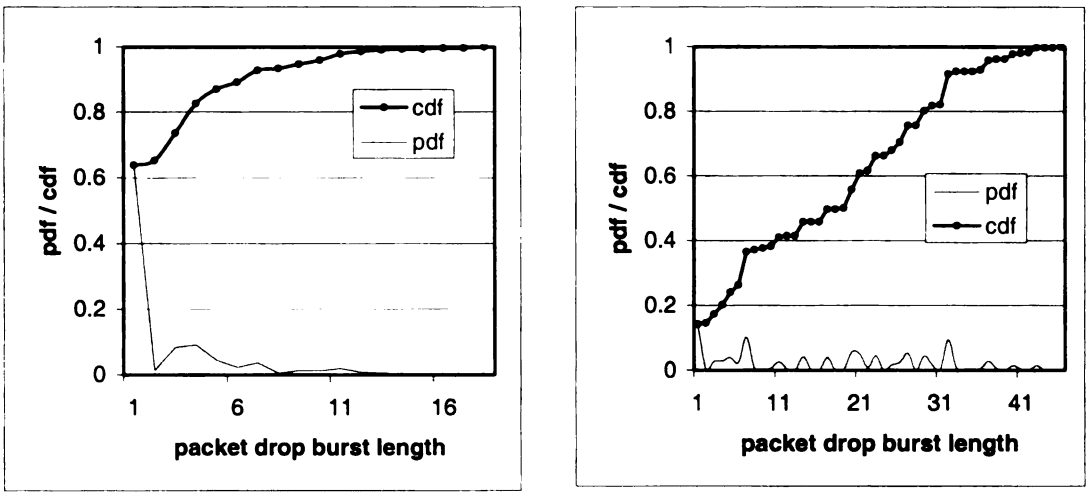
the feasibility of high-bitrate applications. Our previous analysis has revealed that mostly the errors occurred as bursts of corrupted bits. Often, these bit error bursts extended across byte (and sometimes packet) boundaries, thus, resulting in bursts of corrupted bytes and packets. Cumulative and probability density functions of packet loss bursts for 5.5 and 11 Mbps are outlined in Figure 16. At 5.5 Mbps (see Figure 16 (a)), more than 60% of packet bursts had a length of one dropped packet. Also, more than 90% of the times, the length of the packet drop burst is smaller than 6 packets. This depicts that at 5.5 Mbps the error-rate of the 802.11b LAN is relatively low when compared with 11 Mbps. In other words, at 5.5 Mbps the byte-level bursts are small and, therefore, they are not frequently spread across packet boundaries. Nevertheless, the longest burst in this case is of 17 consecutive packets. Moreover, it is clear from Figure 16 (b) that at 11 Mbps more than 90% of the times the packet drop bursts are 31 packets long, incurring bursts as long as 45 packets. In addition, the probability distribution has a very high standard deviation, which complicates the design of an application-layer error resilience scheme. Table 7 outlines the packet drop mean, standard deviation and throughput at 2, 5.5 and 11 Mbps, respectively.

Bitrate	Mean	Standard Deviation	Throughput (%)
2	0.00029	0.02329	99.9825
5.5	1.67930	2.94070	63.8613
11	16.3493	12.3343	14.2361

Table 7. Packet Drop Statistics of Traditional MAC-Layer for an 802.11b Network

The average length of packet bursts is 0.0003, 1.68 and 16.35 for 2, 5.5 and 11 Mbps respectively. Clearly, at 2 and 5.5 Mbps the channel bursts are relatively small, whereas at 11 Mbps bursts are on average 16 packets long. Furthermore, at 11 Mbps the standard

deviation of packet burst length is very high, i.e., approximately 12 consecutive packets drops. The resulting throughput drops to 14.23%, which can have a profound adverse affect on the quality of perceived media.



(a) (b)

Figure 16. Cumulative density and probability distribution of 802.11b MAC packet drop bursts at (a) 5.5 Mbps, and (b) 11 Mbps.

2.5.2. Transport Layer Throughput Analysis

In the light of our previous discussion, MAC Lite can be used to improve throughput at the MAC layer. At this point an imperative direction is to study strategies at layers above MAC, which can be employed to maximize the throughput of packets with zero-errors or with errors that are tolerable by the multimedia decoders. Network layer amendments are impractical since any modification at this layer must be reflected in the routers. Also, IP layer computes the checksum over the IP header only and we believe that packets with errors in the IP header can be potentially damaging and, thus, should be dropped. The transport layer, however, performs checksum over the entire packet (trans-

port layer header and payload) and is, therefore, the major source of packet drops in this scenario. Hence, further analysis of the transport layer is necessary to improve the eventual "throughput" reflected at the application layer.

As mentioned previously, UDP Lite addresses this problem by providing the flexibility of partial checksum. The performance of UDP Lite for video over cellular networks was evaluated in [42]. It shows significant improvement over UDP in terms of throughput and PSNR. The simulations were performed on a custom wireless simulator (*WSim*) utilizing error traces provided in [9]. Traces used for this study listed a sequence of corrupt and correct frames without providing error distributions within a corrupt frame. Traces with error rate of 1.58% were employed, which generated a packet drop of approximately 2.1% (for UDP without retransmissions). Non-bursty (random) errors were induced at a fixed rate (20%) for each corrupted video frame[14].

A custom simulation testbed was developed for our simulations. This simulator generated the appropriate UDP, IP and MAC header fields and appended them to the application-layer payload. At the client side, the error traces were used to corrupt 802.11b frames before subjecting them to MAC Lite, IP and UDP layer parsing.

2.5.3. Throughput of UDP with MAC Lite

Figure 17 shows the CDF and PDF for packet drops of UDP in conjunction with MAC Lite at 5.5 and 11 Mbps. These plots support the comment that most of the cor-

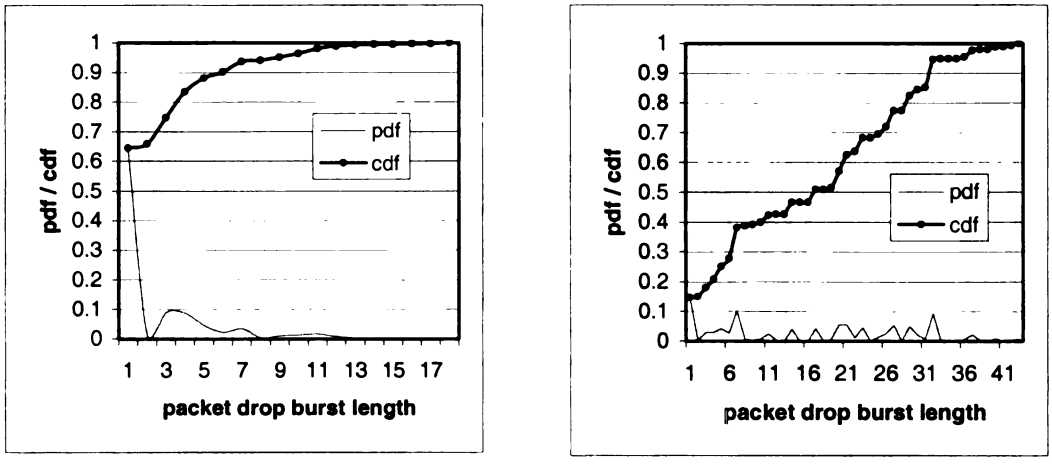
[14] Our analysis indicates a much higher packet drop in 802.11b networks (@ 5.5 and 11 Mbps).

rupted frames being passed by MAC Lite are being dropped at IP and UDP layers. The distributions largely resemble traditional 802.11b MAC statistics (see Figure 16). The longest packet drop burst for 5.5 and 11 Mbps is 17 and 42 packets, respectively.

Table 8 outlines the packet drop burst statistics of UDP (with MAC Lite) at 2, 5.5 and 11 Mbps. Table 8 suggests that MAC Lite improves the overall throughput by 0.47%. This improvement is quite insignificant and will be further reduced for longer traces.

Bitrate	Mean	Standard Deviation	Throughput (%)
2	0.00029	0.02329	99.9825
5.5	1.59579	2.82139	64.4225
11	15.6719	11.8852	14.702

Table 8. Packet Drop Statistics for UDP (in Conjunction with MAC Lite)



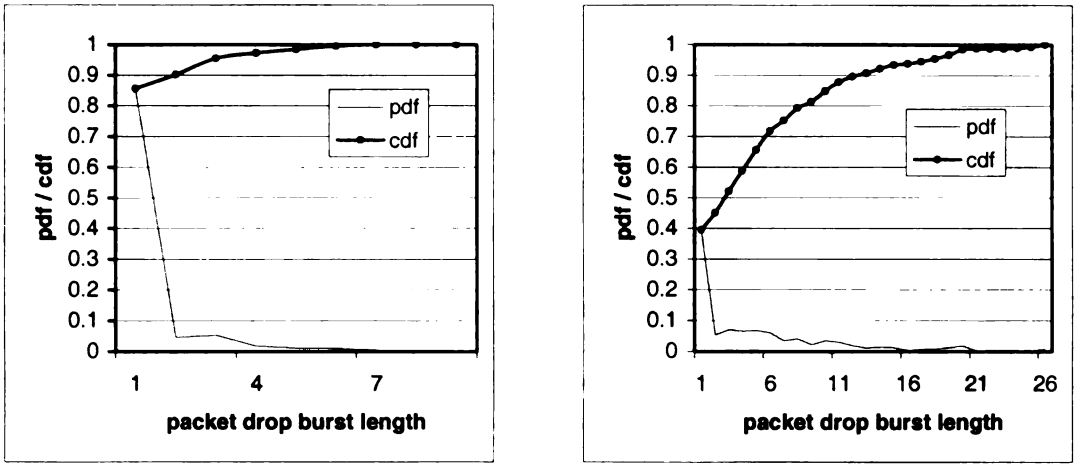
(a) (b)

Figure 17. Cumulative density and probability distribution of UDP (in conjunction with MAC Lite) packet drop bursts at (a) 5.5 Mbps and (b) 11 Mbps.

2.5.4. Throughput of UDP Lite with MAC Lite

The probability distributions in Figure 18 show a positive improvement in the behavior and standard deviation of the packet bursts. Approximately 95% of the times the

packet drop burst is smaller than 2 packets. This is an encouraging result, since it depicts a clear improvement in the throughput by a UDP Lite-based framework. The longest packet drop burst length is 8 and 25 for 5.5 and 11 Mbps, respectively.



(a) (b)

Figure 18. Cumulative density and probability distribution of UDP Lite packet loss bursts at (a) 5.5 Mbps and (b) 11 Mbps.

Bitrate	Mean	Standard Deviation	Throughput (%)
2	0.00006	0.00783	99.9994
5.5	0.33651	0.97775	85.61
11	4.20488	5.44947	39.57

Table 9. Packet Drop Statistics for UDP Lite

Table 9 shows that the average burst length is much smaller in case of UDP Lite (4.2 as compared to 15.67 for 11 Mbps). Moreover, standard deviation of the UDP Lite distribution is approximately 50% less than UDP. These observations could facilitate the design of an application-layer FEC that is capable of combating these impairments. Another promising aspect is the throughput increase of 21.2% and 24.86% for 5.5 and 11 Mbps.

2.5.5. Discussion

The analysis provided in this section clearly depicts that UDP Lite enhances the end-to-end throughput significantly. In addition, it reduces the variance of the packet drop distribution which in turn mitigates the complexity of designing an application-layer error control scheme. Nevertheless, increase in throughput is not a direct metric of improvement in the quality of real-time media. The number and distribution of errors, in the partially corrupted packets provided by UDP Lite, can severely degrade the quality of perceived media. Consequently, a focal question needs to be addressed in this context; Can the additional "high-throughput" (corrupted) packets provided by UDP Lite enhance the quality of transmitted real-time media? The answer to this question depends on the number and distribution of errors in a corrupted packet. The following section provides insight into the byte-level error patterns observed in the high-throughput (corrupted) packets.

A.3 Byte-Level Error Pattern Analysis

In this section, we investigate the byte-level error patterns induced by the channel. We present some of the key statistics of the byte-level burst lengths. Figure 19 illustrates the PDF for byte-level burst-lengths at 2, 5.5 and 11 Mbps. Mean and standard deviation of these bursts are tabulated in Table 10.

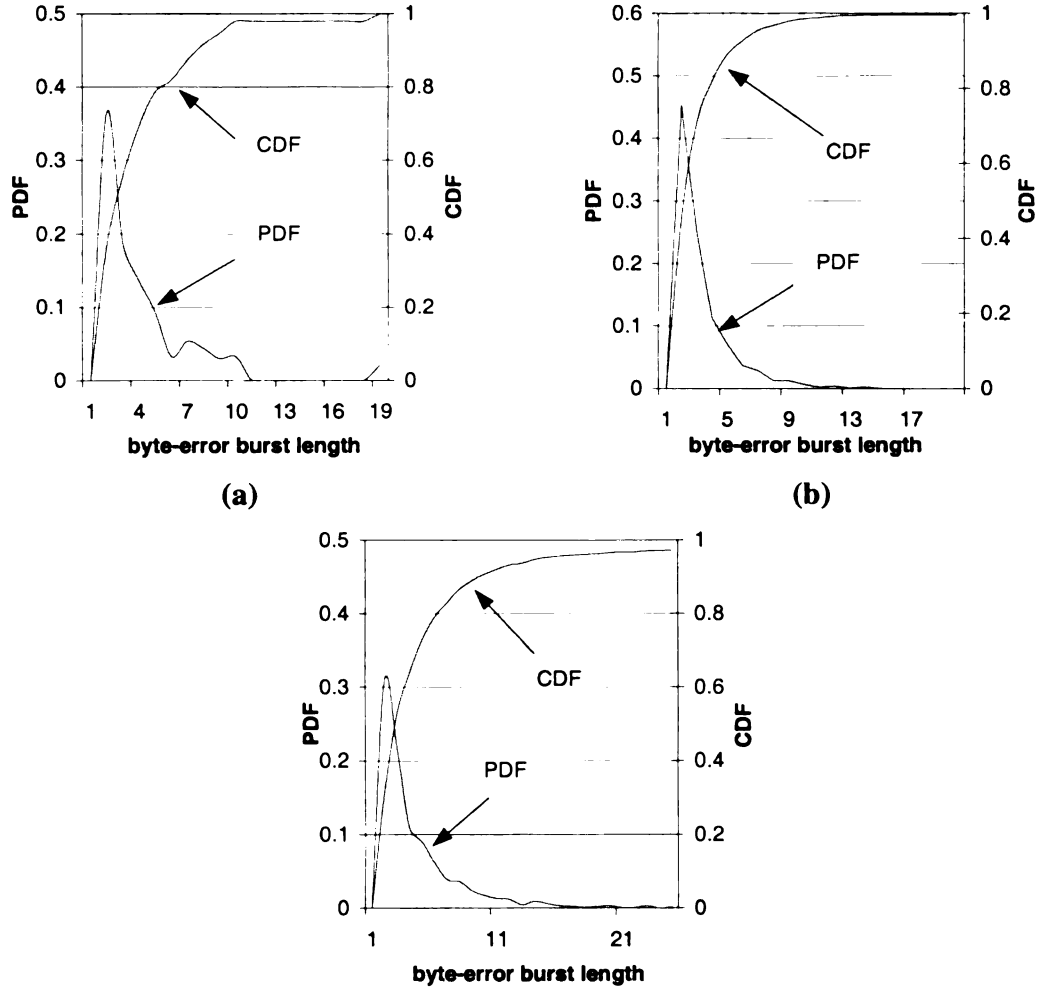


Figure 19. Distribution of byte-level bursts at (a) 2 Mbps, (b) 5.5 Mbps, and (c) 11 Mbps

Bitrate	Mean	Standard Deviation
2	4.232	3.108
5.5	3.3948	2.384
11	5.458	6.722

Table 10. Statistics For Byte-Level Analysis at 2, 5.5, and 11 Mbps

A.3.1 Modeling of Byte-Level Probability Distribution

Due to the diversity of the Gamma distribution in capturing the shape and scale of a wide range of statistical data, we employed non-linear regression analysis to generate a best-fit Gamma PDF for our byte-level burst length:

$$f(x) = \frac{\lambda e^{-\lambda x} (\lambda x)^{\alpha-1}}{\Gamma(\alpha)}$$

for $x \geq 0$, $\lambda > 0$, $\alpha > 0$. The resulting Gamma PDFs are shown in Figure 20. The shape and scale parameters for the fits are given in Table 11. Overall, the Gamma PDF fits the general shape of the collected-data histograms, but is unable to fit the tails. Many error control techniques are not designed for the very worst-case scenario. Hence, distribution of the tail is important for determining an efficient error control scheme.

Exponential and Pareto distributions were applied to provide a suitable fit for the tail. Although the exponential distribution is a special case of the Gamma distribution, one can model the tail with Gamma parameters that are different from the Gamma parameters used for the main distribution. In this case, the tail will have the parameter $g = 1$, which represents an exponential random variable, while the main PDF will have $g \neq 1$. These fits are shown in Figure 21. The Pareto distribution provided a good fit as the correlation was greater than 90% for all data rates. The fit provided by the exponential distribution had a lower correlation than the Pareto distribution for all three data rates, but for 5.5 Mbps the fit given by exponential was comparable to the Pareto fit. Thus, it can be said that even though the overall probability distribution looks like Gamma, the tail indeed

exhibits a heavy-tail behavior. This characteristic can be attributed to the occurrences of long byte-level errors.

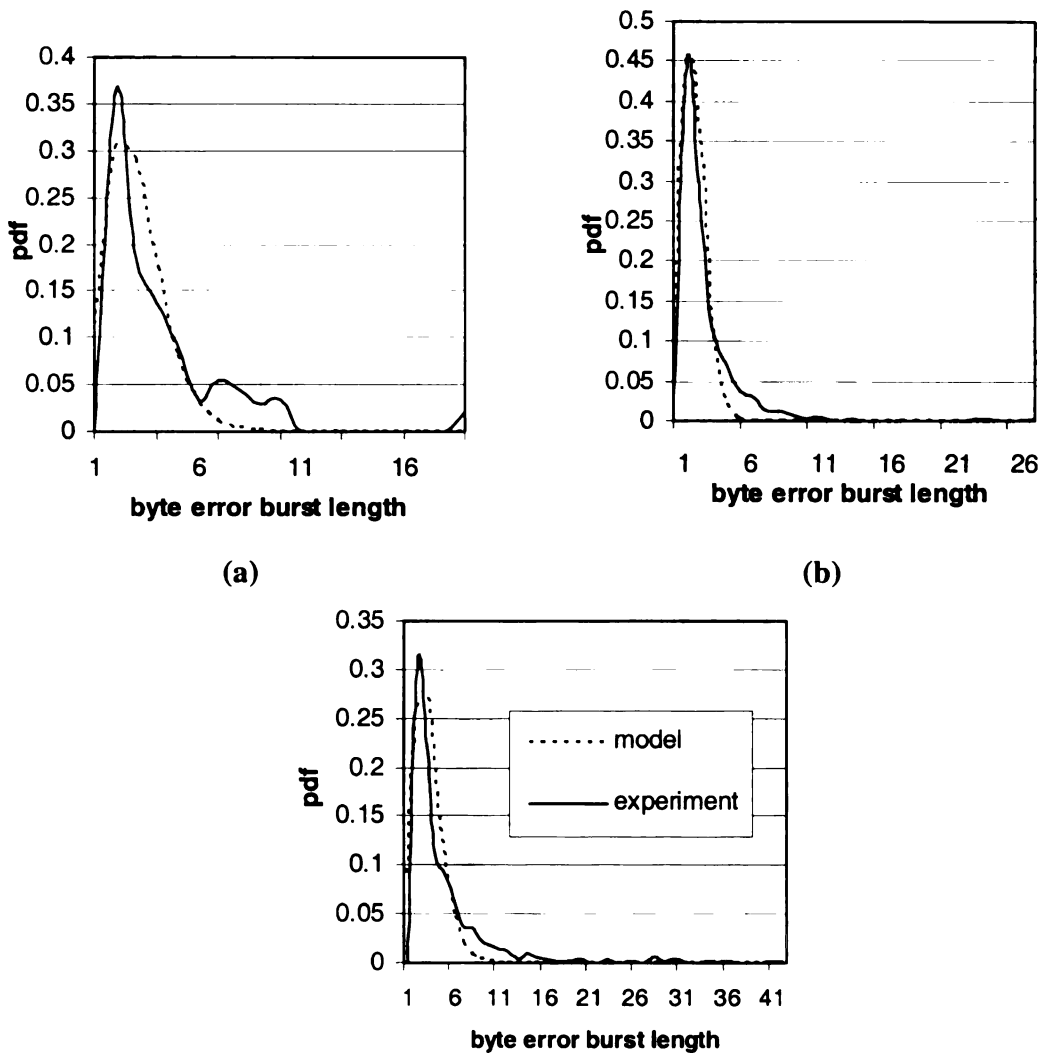
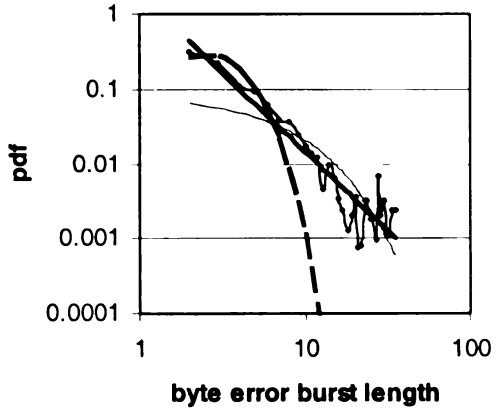


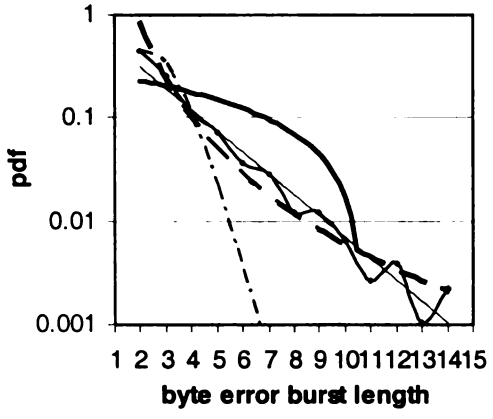
Figure 20. Gamma approximation for byte-level burst distribution at (a) 2 Mbps, (b) 5.5 Mbps, and (c) 11 Mbps.

Bitrate	g (shape)	b (scale)
2	4.96682	0.60563
5.5	8.07703	0.31632
11	4.48498	0.726767

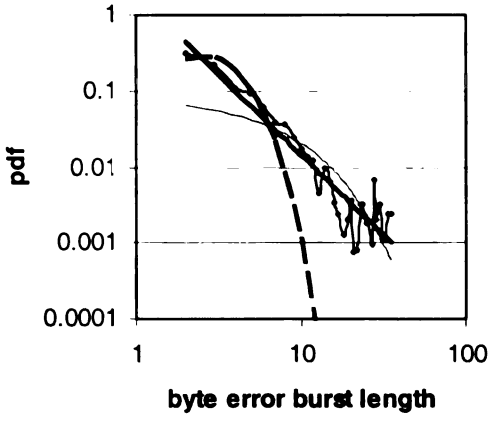
Table 11. Parameters of Gamma Distribution for 2, 5.5 and 11 Mbps



(a)



(b)



(c)

Figure 21. Tail modeling of byte-level distribution for (a) 2, (b) 5.5, and (c) 11 Mbps.

A.4 Application Layer Analysis

Discussions in the previous sections have primarily focused on throughput improvements and analysis at layers below the application layer. At this point it is essential to study the impact of the "high-throughput" (corrupted) packets on the application. To evaluate an example of a high-bitrate multimedia application, several MPEG-4 video sequences were compressed and a corresponding set of video bitstreams were generated to simulate transmissions at 2, 5.5, and 11 Mbps. Visual evaluation of these simulated transmissions were conducted and are provided below. It is important to note that MPEG-4 provides error resilience features such as reversible variable length coding, data partitioning, and other techniques. These error resilience techniques were enabled in our simulations. Packetization utilized the "video packet feature", thus, preventing the propagation affect of a single packet loss across multiple packet frames. Figure 22 presents examples of our video experimental results. Figure 22 (b) is consistent with our previous observations (i.e. the error-rate at 2 Mbps is negligible) so some modest artifacts can be observed in the corrupted frames but the overall quality of the sequence remains largely unaltered. Therefore, the following visual results are provided for the 5.5 and 11 Mbps cases only. Figure 22 (c) and (d) clearly depict that, even with some video-specific error-resilience, and in the absence of any error- or loss-recovery, the quality of perceived image degrades drastically at higher bitrates. In addition, comparison of Figure 22 (e)(f) and Figure 22 (g)(h) outlines that the high throughput (corrupted) packets provided by UDP Lite fail to improve the perceived video quality. Our conclusions are in agreement with [43] which concludes that current video-specific error resilience techniques cannot deliver high-

quality video under bursty (high error-rate) channel conditions. Thus it can be concluded that irrespective of the transport layer protocol, an application-layer FEC is required to deliver high-quality multimedia at bitrates higher than the 2 Mbps rate. Hence, the remainder of this section addresses the last question raised in the beginning of this appendix: What level of FEC would be required to support transmission of high-quality and high-bitrate multimedia?

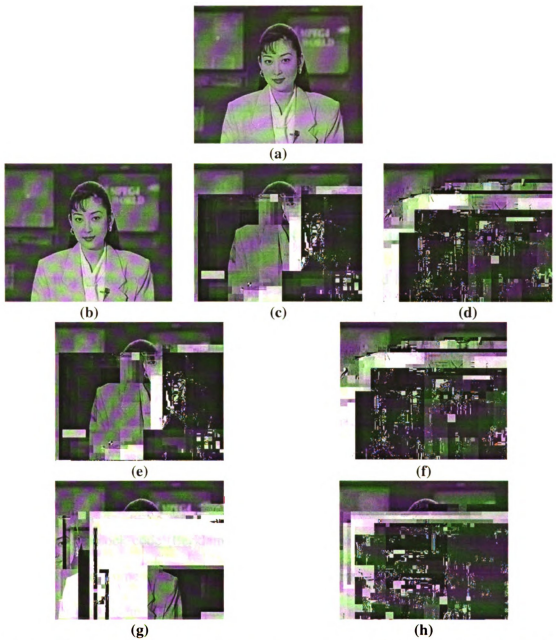


Figure 22. (a) MPEG-4 encoded original sequence; Decoded video after simulated transmission using traditional 802.11b protocol stack at (b) 2 Mbps, (c) 5.5 Mbps, and (d) 11 Mbps; using UDP (with MAC Lite) at (e) 5.5 Mbps, and (f) 11 Mbps; using UDP Lite (with MAC Lite) at (g) 5.5 Mbps, (h) 11 Mbps.

Analysis in preceding sections concludes that no FEC is required for the 2 Mbps case and the error probability at the 11 Mbps is too high for an efficient FEC scheme to provide significant improvement in media quality. Henceforth, further analysis in this section focuses on the 5.5 Mbps case. Before proceeding, it is worth mentioning that we need to address both packet losses and errors, where a packet loss is considered as an *erasure*[15]. For error correction, if a codeword has r number of redundant symbols then a maximum of $\lfloor r/2 \rfloor$ transmission errors in that block can be corrected. No error can be corrected if the number of transmission errors in a block is greater than $\lfloor r/2 \rfloor$. Meanwhile, many contemporary FEC schemes have been designed for and applied to erasure recovery (e.g., [40]). Erasures naturally imply knowledge of error locations. This knowledge of the error locations can help the FEC algorithm in error recovery. Hence, if a codeword has a redundancy of r then a maximum of r erasures can be recovered. Here, we focus on block-based FEC schemes due to their popularity and simplicity for recovering and correcting losses and errors.

For UDP-based transport layer schemes, we employ a Generalized Reed-Solomon-based erasure block code (Berklamp algorithm [41]) to recover dropped packets. In the UDP-Lite case, some of the packets reaching the application layer have errors, while some other packets are still completely dropped. Thus, an FEC scheme for the UDP Lite should be able to decode errors and erasures simultaneously. Therefore, we use a variant

---

[15] A channel "erasure" is an error, for which the position of the error is known but the magnitude is not. Therefore, a dropped packet can be classified as an erasure.

of the abovementioned FEC algorithm that is capable of error and erasure recovery [41]. When using the FEC algorithm for either UDP or UDP Lite, we record a "*decoding error*" if the redundancy in a block is not sufficient to recover and correct all the erasures and/or errors. Therefore, the *decoding error probability* (P_e) quantifies the level of failure (to recover corrupted or lost bytes) that is encountered by the FEC scheme. Other important measures that we considered include "*message packet throughput*" (T_M) and "*bandwidth utilization*" (BW_U), where:

$$T_M = 100 \times \left(\frac{\text{Number of message packets received}}{\text{Number of message packets transmitted}} \right)$$

$$BW_U = 100 \times \left(\frac{\text{Number of message packets received}}{\text{Total number of packets (message+redundancy) transmitted}} \right)$$

BW_U quantifies the efficiency of the FEC scheme and is inversely proportional to the redundancy in a codeword. Similar to a previous section, we evaluate the performance of FEC for the three combinations of MAC/UDP protocol scenarios: traditional 802.11b MAC with UDP, MAC Lite with UDP, and MAC Lite with UDP Lite. We use an FEC codeword length of n=100 bytes. And each FEC codeword is composed of one byte from a different packet, where each packet consists of 512 bytes. Therefore, each packet contributes to 512 separate FEC codewords, and each codeword spans over 100 packets. The results for the two UDP scenarios are shown in Table 12. The first row of the table shows the performance numbers when no FEC is employed. Note that although the throughput is high when no (or small) redundancy is used, the *decoding error probability* is very high (100% for the zero-redundancy case). Therefore, in essence Table 12 outlines the tradeoff between efficiency and reliability. Also from the tables, one can observe that the

message packet throughput without any redundancy is slightly greater than the 10% redundancy case. In such a scenario, the redundant packets will not cause a noteworthy reduction in the decoding error probability; hence they can be replaced by message packets which in turn improve the overall bandwidth utilization.

UDP with 802.11b MAC				UDP with MAC Lite			
r (%)[16]	P_e	T_M	BW_U	r (%)	P_e	T_M	BW_U
0	1	64.42	64.42	0	1	63.86	63.86
10	0.96	64.39	57.95	10	0.97	63.81	57.93
20	0.75	68.05	54.44	20	0.76	67.41	53.93
30	0.57	72.31	50.62	30	0.58	71.85	50.29
40	0.41	78.05	46.83	40	0.42	77.34	46.40
50	0.28	83.62	41.8	50	0.29	82.87	41.43
60	0.02	98.55	39.42	60	0.03	98.10	39.24

Table 12. FEC Analysis for UDP with Traditional 802.11b MAC and MAC Lite at 5.5 Mbps

For the UDP Lite case, we need to consider other parameters since both errors and erasures are involved. In addition, it is important to identify the condition that causes a decoding error to occur in this case. Let e_1 be the number of erasures and e_2 be the number of errors in an FEC codeword, then, complete recovery is possible iff $e_1+2\times e_2 \leq r$. Therefore, if $e_1+2\times e_2 > r$, then we have a *decoding error* and no (error-free) recovery is possible. Consequently, insufficient redundancy may cause some packets to have corruptions even after the FEC decoding. Hence, we measure two additional parameters: "cor-

[16] r (%): percentage redundancy

rupted part of message packet throughput (T_{MC})" and "corrupted part of message bandwidth (BW_{UC})", where:

$$T_{MC} = 100 \times \left(\frac{\text{Number of corrupted message packets received after FEC}}{\text{Number of message packets received after FEC}} \right)$$

$$BW_{UC} = 100 \times \left(\frac{\text{Number of corrupted message packets received after FEC}}{\text{Number of message packets transmitted}} \right)$$

We also measure the packet *corruption level* (c_l), which is a direct measure of the level of errors in the corrupted packets, where:

$$c_l = 100 \times \left(\frac{\text{Number of corrupted bytes in packets with corruptions}}{\text{Total number of bytes in a packet}} \right)$$

Table 13 summarizes the FEC performance numbers based on the above parameters for the UDP Lite scenario. It is clear from the table that higher levels of effective (error-free) throughput are achievable in this case when compared with the traditional UDP scenario. For example, based on our traces at 5.5 Mbps, we were able to achieve 65% effective bandwidth utilization while having zero decoding errors (which implies $T_{MC} = 0$ and $BW_{UC} = 0$). Also, at relatively low decoding error probabilities, we achieved around 70% bandwidth utilization.

r (%)	P_e	T_M	BW_U	T_{MC}	BW_{UC}	c_l
0	1	85.25	85.2	23.9	20.4	40.7
10	0.88	86.39	77.8	20.3	15.8	37.6
20	0.38	89.53	71.6	12.6	9	50.4
30	0.12	95.89	67.1	3.17	2.13	4.36
35	0	100	65	0	0	0

Table 13. FEC Analysis for UDP Lite at 5.5 Mbps

A key reason for these improvements in effective throughput when compared with the UDP scenarios can be explained by taking the corruption level into consideration. Under the MAC Lite/UDP Lite case, even when a decoding error occurs, the corruption level in already received corrupted packets could be reduced, and consequently, at least some part of a dropped packet could be recovered. Indeed, not all 512 codewords in a block fail to recover any bytes in the case of a decoding error. Such a scenario was not possible in the UDP case since all the 512 codewords had equal number of erasures to correct. Furthermore, when the number of lost packets (erasures) were higher than the number of redundant packets, then no recovery was possible in the UDP case. However, in the case of errors and erasures, codewords can have different number of errors, and hence, the number of recoveries will change respectively. In effect, if the number of errors in corrupted packets is relatively low (i.e., substantially lower than 50%), then the above property enables complete recovery with comparatively lesser redundancy. It can be observed that there is more than 60% improvement in effective throughput (while achieving complete recovery) in the UDP Lite case when compared with UDP.

Hence, for our 5.5 Mbps experiments, completely reliable high-bitrate multimedia could be delivered while utilizing more than 3.5 Mbps of the total 5.5 Mbps bitrate (i.e., around 65% utilization). This gives some measure for a lower bound on maximum bandwidth utilization with 100% reliability on an 802.11b LAN for multimedia applications. Naturally, higher bitrates can be achieved if some errors that can be concealed by the application are acceptable. Hence, we conclude that the MAC-Lite/UDP Lite-based framework in conjunction with application-layer FEC, exhibits clear throughput improvements over traditional UDP. It is important to note that this conclusion is true regardless of the

MAC layer strategy used in conjunction with UDP (i.e., in conjunction with either traditional MAC-based retransmissions or MAC Lite-based).

REFERENCES

[1] M. Yajnik, S. Moon, J. Kurose, and Don Towsley, "Measurement and Modelling of the Temporal Dependence in Packet Loss," *Proceedings of IEEE INFOCOM*, March 1999.

[2] D. Loguinov and H. Radha, "End-to-End Internet Video Traffic Dynamics: Statistical Study and Analysis," *Proceedings of IEEE INFOCOM*, June 2002.

[3] S. Khayam, S. Karande, M. Krappel, and H. Radha, "Cross-Layer Protocol Design for Real-Time Multimedia Applications over 802.11b Networks," to appear in *Proceedings of IEEE Conference of Multimedia and Expo (ICME)*, July 2003.

[4] ISO/IEC JTC 1/SC 29/WG 11, "Text of ISO/IEC 14496-2:2001 (Unifying N2502, N3301, N3056, and N3664," Doc. N4350, July 2001.

[5] J. Bolot, and A. V. Garcia, "Control Mechanisms for Packet Audio in the Internet," *Proceedings of IEEE INFOCOM*, April 1996.

[6] L. Larzon, M. Degermark, and S. Pink, " UDP Lite for Real Time Multimedia Applications," *Proceedings of IEEE International Conference of Communications (ICC)*, June 1999.

[7] L. Larzon, M. Degermark, and S. Pink, " Efficient Use of Wireless Bandwidth for Multimedia Applications," *Proceedings of IEEE International Workshop on Mobile Multimedia Communications (MoMUC)*, November 1999.

[8] L. Larzon, M. Degermark, and S. Pink, "The UDP Lite Protocol," *IETF Internet Draft*, February 2001.

[9] R. Ludwig, A. Konrad, and A. Joseph, "Optimizing the End-to-End Performance of Reliable Flows over Wireless Links", *Proceedings of ACM Mobicom*, August 1999.

[10] A. Konrad, B. Y. Zhao, A. D. Joseph, and R. Ludwig, "A Markov-Based Channel Model Algorithm for Wireless Networks," *Proceedings of Fourth ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, 2001.

[11] S. Khayam, S. Karande, H. Radha, and D. Loguinov, "Performance Analysis and Modeling of Errors and Losses over 802.11b LANs for High-Bitrate Real-Time Multimedia," to appear in *Signal Processing: Image Communication Journal*, 2003.

[12] S. Karande, S. Khayam, M. Krappel, and H. Radha, "Analysis and Modeling of Errors at the 802.11b Link Layer," to appear in *Proceedings of IEEE Conference of Multimedia and Expo (ICME)*, July 2003.

[13] N. Merhav, M. Gutman, and J. Ziv, "On the Estimation of the Order of a Markov Chain and Universal Data Compression," *IEEE Transactions on Information Theory*, vol. 35, pp 1014-1019, September 1989.

[14] N. Slonim, G. Bejerano, S. Fine, and N. Tishby, "Discriminative Feature Selection via Multiclass Variable Memory Markov Model," *Proceedings of International Conference on Machine Learning (ICML)*, July 2002.

[15] ISO/IEC 8802-11:1999(E), "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," August 1999.

[16] IEEE Std 802.11b-1999, "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Higher-Speed Physical Layer Extension in the 2.4 GHz band," September 1999.

[17] F. Cali, M. Conti, and E. Gregori, "IEEE 802.11 Protocol: Design and Performance Evaluation of an Adaptive Backoff Mechanism," *IEEE Journal on Selected Areas in Communications (JSAC)*, Vol. 18, No. 9, pp. 1774-1786, September 2000.

[18] S. Khurana, A. Kahol, S. Gupta, and P. Srimani, "Performance Evaluation of Distributed Co-Ordination Function for IEEE 802.11 Wireless LAN Protocol in Presence of Mobile and Hidden Terminals," *Proceedings of 7th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, March 1999.

[19] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda, "Performance anomaly of 802.11b," *Proceedings of IEEE INFOCOM*, April 2003.

[20] Linux-wlan Home Page. *http://www.linux-wlan.org/*.

[21] H. Radha, M. van der Schaar, Y. Chen, "The MPEG-4 Fine-Grained Scalable Video Coding Method for Multimedia Streaming over IP," *IEEE Transactions on Multimedia, March*, 2001.

[22] M. van der Schaar, Y. Chen, H. Radha, "Embedded DCT and Wavelet Methods for Scalable Video: Analysis and Comparison," *Visual Communications and Image Processing*, January 2000.

[23] P. Brockwell and R. Davis, "Introduction to Time Series and Forecasting," Springer:Verlag, 1996.

[24] Alberto Leon-Garcia, "Probability and Random Processes for Electrical Engineering," 2nd Ed., Addison-Wesley: Reading - Massachusetts, 1994.

[25] Sheldon M. Ross, "Introduction to Probability Models," 7th ed., Harcourt Academic Press: San Diego - California, 2000.

[26] Raymond W. Yeung, "A First Course in Information Theory," Kluwer Academic/Plenum Publishers: New York – New York, 2002.

[27] Hayder Radha, "Lecture Notes: ECE 802 - Information Theory and Coding," January 2003.

[28] F. M. J. Willems, Y. M. Shtarkov, T. J. Tjalkens, "The Context Tree Weighting Method: Basic Properties," *IEEE Transactions on Information Theory*, vol. 41, pp. 653--664, May 1995.

[29] Dana Ron, Yoram Singer, and Naftali Tishby, "Learning Probabilistic Automata with Variable Memory Length," *Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory*, 1994.

[30] Lawrence R. Rabiner, "A tutorial on Hidden Markov Models and selected applications in speech recognition," *Proceedings of the IEEE*, 77(2):257--286, 1989.

[31] L. R. Rabiner and M. R. Sambur, "An Algorithm for Determining the Endpoints for Isolated Utterances," *Bell System Technical Journal*, vol. 54, no. 2, pp 297--15, February 1975.

[32] C. E. Koksal, H. I. Kassab, and H. Balakrishnan, "An Analysis of Short-Term Fairness in Wireless Media-Access Protocols," *Proceedings of ACM SIGMETRICS*, June 2000.

[33] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A Comparison of Mechanisms for Improving TCP Performance over Wireless Links," *IEEE/ACM Transactions on Networking*, December 1997.

[34] S. D. Servetto, R. Puri, J. Wagner, P. Scholtes, and M. Vetterli, "Video Multicast in (Large) Local Area Networks," *Proceedings of IEEE INFOCOM*, June 2002.

[35] S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-Driven Layered Multicast," *Proceedings of ACM SIGCOMM*, 1996.

[36] H. Ma and M. El Zarki, "Broadcast/Multicast MPEG-2 Video over Broadband Fixed Wireless Access Networks," *Proceedings of IEEE Network*, December 1998.

[37] M. Veeraraghavan, N. Cocker, T. Moors, "Support of voice services in IEEE 802.11 wireless LANs," *Proceedings of IEEE INFOCOM*, June 2002.

[38] J. L. Sobrinho and A. S. Krishnakumar, "Real-time traffic over the IEEE 802.11 Medium Access Control Layer," *Bell Labs Technical Journal*, vol. 1, no. 2, pp. 172--187, Autumn 1996.

[39] J. Postel, "User Datagram Protocol," RFC 768, August 1980.

[40] L. Rizzo, "Effective Erasure Codes for Reliable Computer Communication Protocols," *ACM Computer Communication Review*, vol. 27, no. 2, pp.24-36, April 1997.

[41] R. E. Blahut, "Theory and Practice of Error Control Codes," Reading, MA: Addison-Wesley, May 1984.

[42] A. Singh, A. Konrad, and A. D. Joseph, "Performance Evaluation of UDP Lite for Cellular Video," *Proceedings of International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, June 2001.

[43] S. Gringeri, R. Egorov, K. Shuaib, A. Lewis, and Bert Basch, "Robust Compression and Transmission of MPEG-4 Video," *Proceedings of ACM Multimedia (MM)*, 1999.