This is to certify that the
dissertation entitled

ADAPTIVE WIRELESS VIDEO USING CHANNEL STATE INFORMATION

presented by

Yongju Cho

has been accepted towards fulfillment
of the requirements for the

Ph.D. degree in Electrical Engineering

Hayder Radha
Major Professor's Signature

Dec 05, 2008
Date

ADAPTIVE WIRELESS VIDEO USING CHANNEL STATE INFORMATION

By

Yongju Cho

A DISSERTATION

Submitted to
Michigan State University
In partial fulfillment of the requirements
For the degree of

DOCTOR OF PHILOSOPHY

Electrical Engineering

2008

ABSTRACT

ADAPTATIVE WIRELESS VIDEO USING CHANNEL STATE INFORMATION

By

Yongju Cho

In this thesis, we tackle the following research problems: i) accurate channel capacity estimation and prediction under Cross-Layer Design with Side-information (CLDS) wireless protocols; ii) an optimal source and channel coding rate prediction and tuning; iii) Unequal Error Protection (UEP); and iv) Syndrome Partial Decoding (SPD) for wireless video in multi-hop networks. i), ii), iii) and iv) are essential factors to complete a rate-adaptive video streaming application over wireless LANs. For these problems, we collected a comprehensive set of datasets (or error traces) which are analyzed and used to verify our proposed schemes in realistic environments.

We provide analysis of bit-errors at IEEE 802.11b MAC layer and develop channel Bit Error Rate (BER) prediction scheme. We show that BER can be accurately predicted by employing multi-tier model which utilizes Received Signal Strength Indication (RSSI) and checksum of a packet. On the basis of the channel BER prediction scheme, an optimal rate tuning; which leverages both the probability distribution of channel capacity prediction error process and an RD (quality) function of a video sequence, is developed.

In order to employ the optimal rate tuning scheme in a practical wireless environment, we deduce a Rate Distortion (RD) model for above-capacity video and an operational rate for a specific channel code. It is observed that the optimal rate tuning provides excellent rate prediction performance. In addition, an UEP scheme; which utilizes characteristics of a LDPC code, is developed, and we show that the UEP scheme reduces the degradation of video quality in case of burst errors. The optimal rate prediction architecture under CLDS protocols, $ORPA_{CLDS}$, which combines all the described schemes above, provides excellent performance in terms of accuracy of rate prediction as well as video quality. Finally, we exhibit the applicability of the proposed architecture in multi-hop wireless networks by introducing SPD which mitigates the complexity and memory requirement of Decoding and Forwarding (DF), while providing reasonably high goodput.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Recently, digital media has had a tremendous impact on the development of the Internet, telecommunications, and broadcasting areas. The rapid popularization of the Internet, wireless communication systems, and digital broadcasting networks have led us to an epochal framework for content services with an end-to-end delivery chain of content generation, distribution and consumption. Multimedia devices and digital TVs have accelerated the easy purchase and consumption of a vast number of media content types. These developments, however, have forced us to consider the reliable delivery of multimedia content over wireless networks to diverse types of devices. Especially Quality of Service (QoS) for multimedia content over mobile wireless networks has been an issue.

To that end, our research embodies an advanced multimedia content delivery that satisfies the needs of reliability and QoS over wireless networks in an efficient manner. Our research on reliable video delivery over wireless networks demonstrates five key aspects: 1) Analyzing the behavior of errors in wireless networks and predicting accurate

Bit Error Rate (BER) by utilizing Cross Layer Design with Side-information (CLDS) protocols, 2) Predicting an optimal source and channel coding rate, 3) Modeling of an RD for above-capacity wireless video, 4) Minimizing the video quality distortion in case of packet loss (Unequal Error Protection (UEP)), and 5) Mitigating the complexity and memory usage at each intermediate nodes in an ad-hoc network, while providing reasonably high goodput. The solution will eventually lead the user to a better QoS experience and enable network providers to benefit from more efficient bandwidth utilization over the entire wireless networks.

1.1. Research Problem

In many wireless environments, deteriorated link conditions cause bit-corruptions. These corrupted packets cause checksum failures and packet drops at wireless receivers. To reduce packet losses at the receivers, many recent efforts utilize cross-layer protocols that do not discard corrupted packets [1][19][31][32]. Consequently, two classes of wireless multimedia protocols have emerged: (i) Cross-Layer-Design (CLD) [19] protocols which relay corrupted packets to higher layer for further processing; (ii) conventional (CON) [19] protocols which drop any packet that has one or more residue

errors[1]. Prior studies have shown that a significant improvement in wireless video throughput can be achieved by CLD [1][19][31][32]. Furthermore, it was also exhibited that side information, which are already available from IEEE 802.11 compliant packets, is quite valuable for providing channel state information and modeling of the underlying (effective) video channel. This side information includes Signal to Silence Ratio (SSR) indicators and MAC-layer checksum, both of which can be used as parameters for channel estimation [21]. (SSR is a packet-level SNR parameter supported by IEEE 802.11 compliant devices.) This form of CLD protocols that utilize side information have been referred to as CLDS protocols in prior literature [19][21][33].

Despite of the demonstrated benefits of CLDS, standard features that can exploit these benefits for CLDS-based video streaming applications remain unexplored. One such important feature is video rate-adaptation based under CLDS protocol. Figure 1 illustrates the rate-adaptive video architecture on which our research focuses. In the architecture a server and a client communicate through a heterogeneous network which consists of a wired and a wireless network. In the given architecture, both a server and a client are designed to support source and channel rate adaptation. Note that a great deal of bit corruptions in packets occur in a wireless network whose link quality fluctuates

[1] Here, a residue error is an error that is not corrected by the physical layer, and hence it appears at the MAC layer.

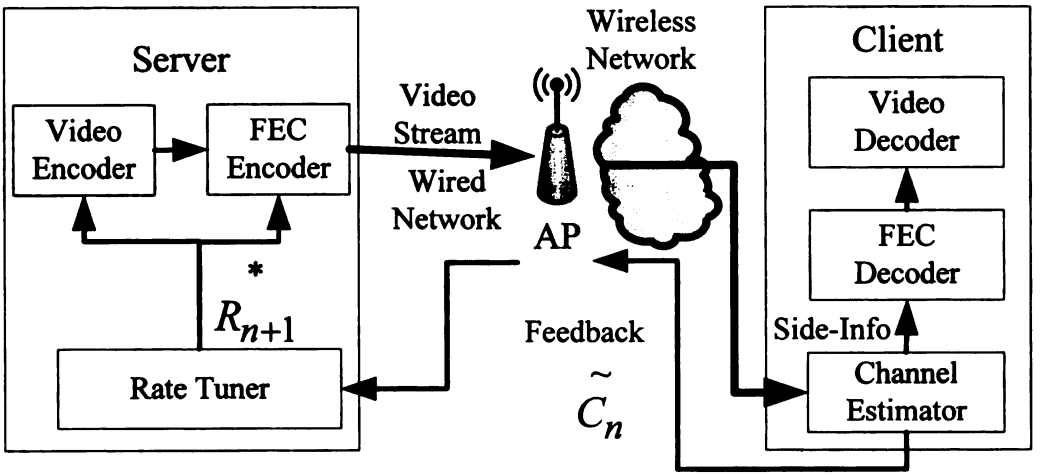


Figure 1 The architecture of the proposed rate adaptation.

significantly due to various interference, fading, multi-path effects, and mobility. Here, we focus on the wireless channel (as seen by above-PHY layers) for rate estimation and prediction in the architecture.

The client supports CLDS protocols that leverage residue-error-process and side information, which can be relayed to an FEC decoder for soft-decoding [37], to estimate the current channel capacity for a block of packets (or time-window). The current channel capacity, which is estimated by the channel estimator in the client with the entropy of the residue error process, is then transmitted to the server as feedback for rate adaptation. Using the feedback, the rate tuner at the server predicts the optimal source and channel rates for the next block of multimedia packets to be transmitted.

It is essential to accurately estimate and predict the channel capacity, which is in turn

can be used to determine the source and channel coding rates, in real-time, for rate-adaptive video [42]. In particular, when a channel-coding rate is under-estimated relative to channel capacity, a client virtually recovers all packets (error-free), but operating below channel capacity naturally and negatively effect the overall throughput of a wireless video streaming session (and hence the video quality). On the other hand, an over-estimated rate provides a larger amount of video information in packets (and less amount of redundancy), but a client receives many corrupted packets with more errors than a channel code can correct. This significantly degrades overall video quality; i.e., the performance of a rate-adaptive video streaming application depends greatly on the accuracy of the channel state estimate.

When CLDS is employed, the channel capacity can be estimated from the entropy of the residue error process [21]. However, accurate estimation/prediction of the residue-error process entropy using only packet-level signal strength information is a challenging task. While these packet level measurements are much coarser than bit-level granularity required for error entropy computation; bit level signal strength measurements are not readily available at the MAC layer of current wireless networks. In addition, we need to adhere to a rate that is strictly below channel capacity to avoid excessive packet drops. Therefore, if the predicted channel capacity is directly employed, there is a good

likelihood that the predicted capacity (as a random variable) may exceed the actual channel capacity, which leads to severe distortion of video quality.

For multimedia streaming applications, it is essential to accurately estimate/predict the channel capacity to provide QoS. However, it is also important to minimize distortion of video quality when video packets are corrupted due to busty errors by severe interference which often occurs in wireless channels. A workaround to this problem is to employ an Unequal Error Protection (UEP) scheme. Under UEP, parts (e.g., Intra-coded frames) of the video bit-stream, which significantly affects video quality, are provided with more protection (i.e. a lower coding rate) and vice versa. Therefore, the video quality depends heavily on how well the video bit-stream is protected for severe interference in wireless channels.

An ad-hoc network consists of many wireless hops from a server to a client, and each wireless link between two intermediate nodes experiences interference. Therefore, it is obvious that there will be channel capacity drops over each hop, and hence, without optimal rate adaptation, the video quality can be expected to degrade significantly at any client located at a far-edge of an ad-hoc network. To overcome this limitation, Decoding and Forwarding (DF), which fully decodes codewords at each intermediate node, can be employed to provide the best video quality. However, complexity and memory usage for

the method are significantly high. Furthermore, intermediate nodes, which may be participating by only forwarding the content toward a receiver further-down a multi-hop chain, do not have much incentive to perform full decoding-encoding of the channel-coded wireless video content. For such nodes, we need to minimize their burden in terms of the operation they need to perform toward the delivery of the video content to the final receiver.

These issues motivate the robust and practical rate adaptation architecture that we develop in this thesis.

1.2. Overview of Contributions

In this thesis, we provide novel schemes to adaptively provide reliable video content delivery over wireless LANs. Our objective is to develop robust and practical rate prediction architecture to support QoS for wireless scalable video. To that end, Chapter 3 focuses on analysis of bit errors at IEEE 802.11b MAC layer and develops a BER estimation/prediction scheme under CLDS protocols. In Chapter 4, on the basis of the BER estimation/prediction scheme (with which the channel capacity is estimated), we take the RD of video into consideration to predict and optimally tune a source and

channel coding rate. In Chapter 5 we develop the operational rate, the empirical RD model for above-capacity wireless scalable Video, and UEP to employ our optimal rate prediction framework to practical rate adaptation architecture. In Chapter 6 we extend our rate adaptation architecture to an ad-hoc network by employing Syndrome Partial Decoding (SPD) to efficiently manage the overall processing requirements of intermediate nodes. Finally, Chapter 7 summarizes key conclusions of this thesis.

In Chapter 3 we describe the method of collecting residual error traces which represents the various and realistic wireless channels and analyze MAC-to-MAC bit-error characteristics. Based on the analysis, we develop the multi-tier model (MTM) [35]; which leverages a received packet's SNR and checksum side-information to predict BER, to accurately predict BER in future packets over a wireless residual channel. It is observed that direct inference of BER from SNR results in optimistic estimates because of the relatively large amounts of error-free data (in comparison with corrupted data) received on viable wireless networks. Consequently, we propose a model that separates packet- and bit-error prediction. For packet-error prediction, a 3^{rd} order Markov chain model is used at the first tier, and the current SSR value is used to predict a SSR value of the next packet at the second tier for bit-error prediction. We demonstrate that the MTM renders higher prediction accuracy than existing Yule-Walker and finite-state Markov

chain predictors at any physical data rate (2, 5.5, and 11 Mbps).

Although our channel capacity prediction scheme based on the BER prediction scheme provides very accurate prediction performance, it can not be free from error. In addition, it is a well know fact that the rate above channel capacity will results in packet drops and degradation of video quality. For these reasons, in Chapter 4, we develop optimal rate prediction architecture under CLDS protocols ($ORPA_{CLDS}$) [55]. In this architecture, we leverage the probability distribution of capacity prediction error process and the RD function of video to optimally predict and tune a rate that maximizes Peak Signal-to-Noise Ratio (PSNR) of video contents. We exhibit the efficacy of the proposed scheme by simulations using actual 802.11b wireless traces, an RD model for the video source and an ideal FEC model. Simulations using source RD models derived from five different popular video codecs (including H.264), show that the proposed framework provides up-to 5 dB improvements in PSNR when compared with conventional rate-adaptive schemes.

In Chapter 4 we assume the following assumptions in $ORPA_{CLDS}$: (i) the channel code achieves the capacity (i.e., an ideal channel coder), and (ii) a block of packets cannot be recovered at the client if it is coded with an overestimated rate, i.e., a channel coding rate that exceeds channel capacity. Thus, in Chapter 5 we incorporate i) an

operational rate for a specific channel code which is not an ideal code, ii) an RD function for above-capacity video, and iii) UEP which utilizes LDPC code to optimally realize $ORPA_{CLDS}$ in practical rate adaptation architecture. We show that $ORPA_{CLDS}$ provides the considerably accurate average PSNR to the maximum PSNR and the higher (at least 6dB in 11Mbps channel) average PSNR to that of $ORPA_{CON}$ for any given channel. This highlights that the proposed schemes bring $ORPA_{CLDS}$ to a certain level of completion in practical wireless environments.

It is obvious that each wireless channel between intermediate nodes faces channel impairment due to interference, fading, and multi-path effects. Hence, when a multi-hop wireless network is introduced, it can be easily observed that channel capacity decreases over each hop, and hence, End-to-End (E2E) channel capacity becomes very low. This leads to significant degradation of goodput and video quality for rate adaptation applications. The workaround to this problem is to employ Decoding and Forwarding (DF). For DF, an intermediate node decodes the transmitted packets (or codewords) to suppress noise on the channel between two intermediate nodes, and re-encodes the packets, potentially with a different codebook, for transmission towards the destination [61]. However, complexity and memory usage for the method are significantly high. Moreover, intermediate nodes, which may be participating by only forwarding the

content toward a receiver further-down a multi-hop chain, do not have much incentive to perform full decoding-encoding of the channel-coded wireless video content. For such nodes, we need to minimize their burden in terms of the operation they need to perform toward the delivery of the video content to the final receiver. To that ends, in Chapter 6 we develop the new partial processing framework for rate-adaptive wireless video, which reduces the overall processing requirements of intermediate nodes. We refer to as Syndrome Partial Decoding (SPD) architecture. We exhibit that SPD, which reduces the overall processing requirements of intermediate nodes, provides reasonably high goodput, when compared to End node decoding and ARQ, and less complexity and memory requirements, when compared to DF.

1.3. Thesis Organization

The rest of this part is organized as follows. Chapter 2 provides background that is required to understand the material presented in this thesis. Chapter 3 focuses on "accurate" BER estimation and prediction of IEEE 802.11b channel. Chapter 4 proposes optimal rate prediction architecture under CLDS protocols for wireless multimedia. In Chapter 5 we complete optimal rate prediction architecture to be employed as a real

world application. In Chapter 6 we extend optimal rate prediction architecture to an ad-hoc network by considering efficient processing requirements of intermediate nodes. In Chapter 7 we summarize key conclusions of this thesis.

CHAPTER 2
BACKGROUND

2.1. Limitation of current multimedia delivery mechanism/protocols

One of the fundamental challenges for wireless multimedia systems is the availability of sufficiently high "effective" bandwidth. In principle, "sufficiently high" "effective" bandwidth implies bandwidth that can support the best possible multimedia quality in presence of packet drops and while resolving access among competing nodes over the shared wireless medium. Video applications are usually extremely bandwidth hungry and can get adversely affected by excessive reduction in the effective bandwidth. Unlike the traditional wired Internet based communication, the number of packet drops due to bit errors can be substantial in wireless networks. In typical multi-room Home/Office settings, there exist many clients that do not have a Line of Sight (LoS) communication path with the wireless Access Point (AP). Such clients often experience significant attenuation in the received signal. In addition, the fading effects and, the variation in

channel quality there of, experienced by such clients is non–trivial. The video quality service available to such clients can be severely limited on account of excessive packet drops due to corruption. The effects of such packet drops can get further exaggerated on account of interference.

The impact of bit corruptions on the effective bandwidth and thus video quality can be substantially reduced by adopting a fresh perspective in the design of Multimedia Communication Protocols for the wireless links. From Shannon's Information Theory (especially the Noisy Channel Coding Theorem), it is well established that it is feasible to communicate information even in presence of bit corruptions (noise). Thus it can be intuitively argued that even partially damaged packets may contain information that should not be completely discarded. This information if retrieved can help alleviate the impact of poor channel quality on video applications. Signal processing algorithms associated with channel/source decoding can enable the efficient retrieval of information from corrupted packets. Thus it makes prudent sense to develop multimedia protocols that do not drop all of the corrupted packets.

The first generation of protocols that relayed corrupted packets to higher layers are developed by simply turning off a checksum at the link-layer and/or by employing partial checksums at the transport layer (and at times other layers). To the best of our knowledge,

the author's were the first to: extended this work to 802.11b WLANS [1] . Subsequently, many researchers have appreciated the utility of not dropping corrupted packets in 802.11b networks (e.g. [11]-[18]) and wireless networks in general. We refer to all these cross-layer protocols that are achieved simplistically by employing partial checksums as Cross-layer Design (CLD).

CLD protocols have shown great promise in improving multimedia delivery, yet they suffer from some fundamental drawbacks. We elaborate upon these drawbacks, by raising the following three important questions.

- *How do we estimate the utility of a corrupted packet?*

The information content and the utility of a corrupted packet are closely tied with the level of corruption in a packet. The CLD schemes do not provide any information about the corruption level in a packet. As a matter of fact, they do not even differentiate between corrupted or uncorrupted packets. Inability to localize the errors can actually be a major drawback for CLD schemes, at times causing it to perform worse than the conventional (CON) protocols that drop/discard corrupted packets. Information retrieval from corrupted packets incurs a processing overhead. The information content in highly corrupted packets may be too little to justify the additional processing, while the processing overhead can be completely avoided if uncorrupted packets could be

identified.

- *How do I retrieve information when packet identities are questionable?*

CLD schemes drop packets that have a corruption in the header, since such corruptions can modify Critical Header Fields (CHF), which are essential to determining the identity of a packet (i.e. flow, destination etc.). Such packet drops due to header corruption could severely diminish the utility of CLD schemes, and thus entirely nullify the primary motivation for their design.

- *How do we efficiently retrieve information from corrupted packets?*

Finally it is important to realize, that Forward Error Correction (FEC) schemes used over CON are not suitable for cross-layer protocols. Channel decoding used with cross-layer protocols need to cater to errors as well as erasures. The ability of a channel decoding to correct errors is closely tied to the ability to localize the errors. In a CLD scheme, the FEC is provided with no information about the correctly received packets. This can severely affect the ability of the decoding algorithm to localize errors. Channel State Information (CSI) is critical in improving the performance of channel decoding algorithms.

2.2. Overview of CLDS

In the discussion provided in the previous sub-section the drawback of CLD schemes were highlighted. In each of the three questions raised above it was noted that the performance of CLD schemes can deteriorate due to lack of receiver-side channel state information or simply side-information that could provide us more information about the corruption status of the packet. Thus a key component of our work is to identify mechanism that can practically facilitate the estimation of the corruption status of packet. We shall use signal strength indications, checksums, traffic information, temporal correlation etc as side-information to estimate the corruption level. We generically refer to all cross-layer protocols, which utilize some sort of side-information as Cross-Layer Design with Side-Information (CLDS).

The various components of a typical CLDS protocol have been shown in the schematic in Figure 2. Unlike CLD schemes which turn off a checksum and thus adhere to a paradigm where the corruption status of a packet is not relayed to higher layers, as shown in Figure 2, in a CLD scheme the checksum status and all other Link-Quality Indications are provided to the higher layers. These link-quality indications can be used by higher layers for improved channel decoding and subsequently for improved source

decoding. The CLDS scheme also employs a Header-estimation block that can detect the identity of a packet in presence of noise. The Header-estimation block typically employs mechanisms/models that facilitate channel prediction/estimation. This channel prediction (possibly in conjunction with other indicators such as SNR, background traffics) is used as side-information for robust detection of headers.

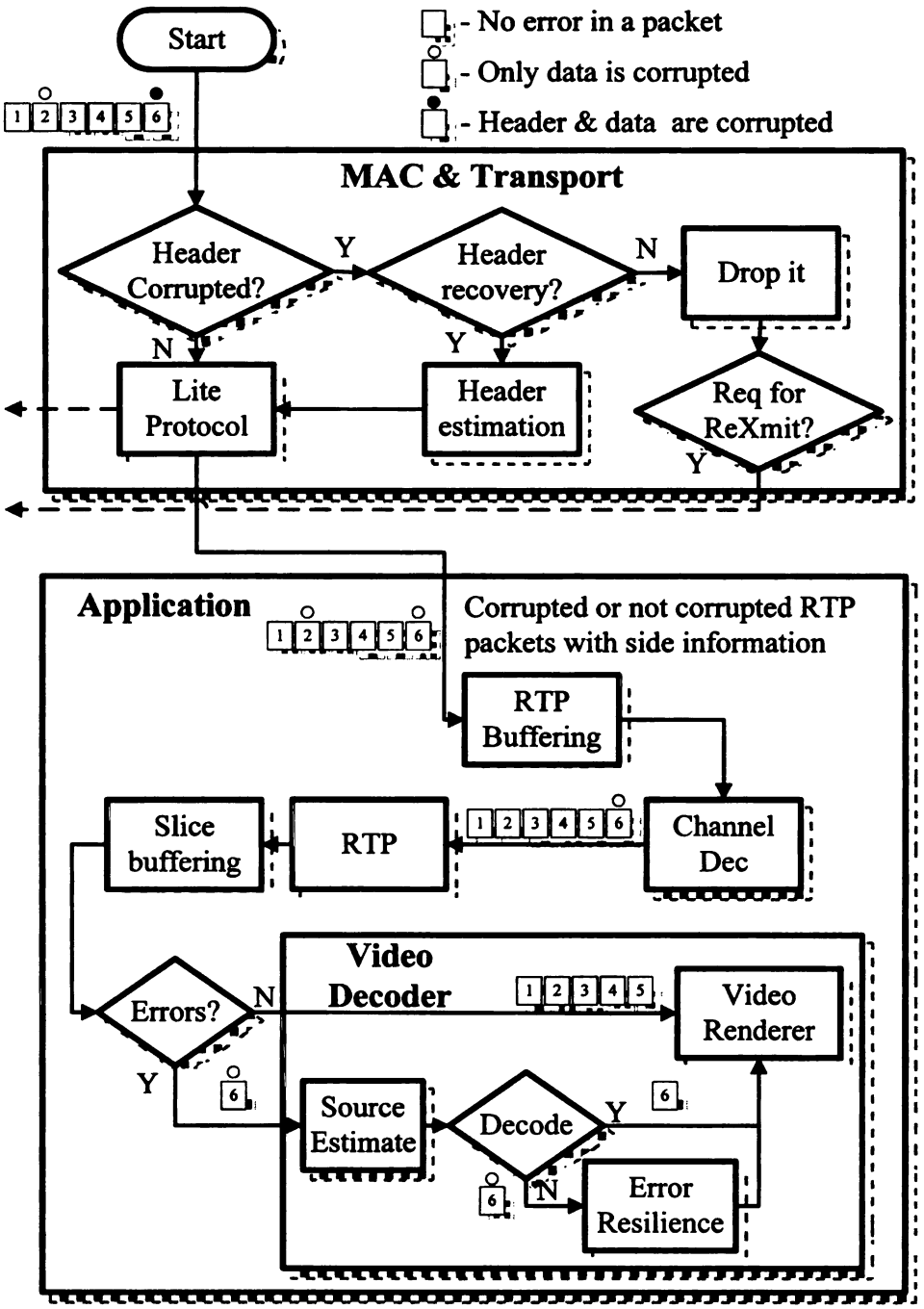


Figure 2 Functional block diagram of CLDS protocol based client

2.2.1. CLDS Channel Evaluation

In order to get a taste of the utility of relaying corrupted packets to higher layers and

the importance of side-information in such protocols, in this section we consider three abstract protocols. We shall characterize the behavior of these protocols with equivalent channel models and subsequently deduce the capacity of these channel models. The comparison of these capacities should allow us to develop clear insight for the problem at hand.

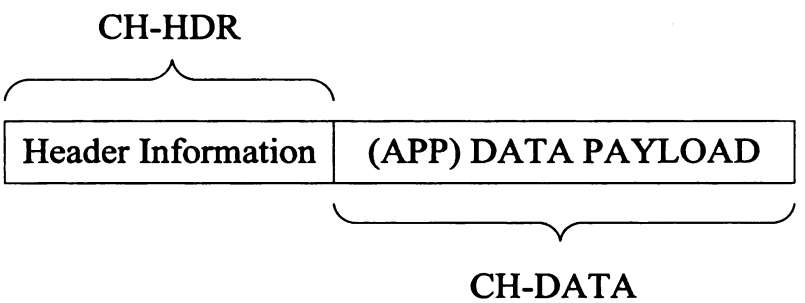


Figure 3 A single Logic Transmission Unit (LTU)

The three communication schemes considered in this paper can be explained by considering a generic Logic Transmission Unit (LTU) as shown in Figure 3. The general packet structure can be segregated into two parts 1) the header information[2] and 2) the data payload. In addition to traditional header information (e.g., node addresses etc), the LTU header contains two sets of checksums CK-HDR and CK-DATA. The CK-HDR checksum is applied to and dependent on the header information only; while the CK-

[2] In many practical implementations the header and CK-HDR might be further partitioned into multiple headers and checksums. In addition a direct correlation of a specific standard/architecture/implementation with the above-considered abstract LTU may not always be possible. Reallocation (or even addition) of some header fields might be required. However, we intentionally do not address such implementation details to maintain the generic nature of arguments presented in this section.

DATA check sum is applied to and dependent on the data payload only. Hence, under this generic LTU model:

- CON, the conventional (non-cross layer) protocol drops a packet if either of the checksums CK-HDR, or CK-DATA is not satisfied.
- CLD, turns the CK-DATA checksum off and drops the packet only if CK-HDR is not satisfied. Therefore, a CLD channel exhibits both erasures (due to CK-HDR violations) and possible errors in some of the delivered packets. It is important to note that (without further information or additional parity bits) the CLD channel receiver does not know which delivered packets are error-free and which packets are corrupted. It only distinguishes between erasures and delivered packets.
- CLDS is an alternative to the above schemes. Similar to CLD, a CLDS channel drops a packet only if CK-HDR is not satisfied. However, in CLDS the CK-DATA is not turned off but neither is the decision to drop a packet dependent on this checksum. Moreover CK-DATA and information about the success or failure of this check-sum is made available to the application layer as side information. Therefore, and unlike a CLD receiver, the CLDS receiver can distinguish corrupted packets from error-free packets.

Each of the above schemes presents a different type of channel to the higher (application) layer. Before we illustrate, what each of these channel look like, we need to

establish certain important parameters that can capture the performance of each of the above protocols. Thus the channels under consideration can be parameterized by:

- δ: The probability that at least a single bit is in error in the header and/or the data payload. Thus δ is the probability of a packet being dropped in a conventional (non-cross layer) protocol because at least one of the checksums, CK-HDR and/or CK-DATA, was not satisfied.
- λ: The probability that the packet header contains at least a single bit in error. Thus λ is the probability of packet being dropped in the cross-layer schemes because the check CK-HDR was not satisfied. (Note that this event could occur regardless if there
- is an error within the packet data or not.)
- Z: A discrete random variable that takes on three possible outcomes: $S_z = \{0,1,?\}$. Where, (i) $Z = ?$ if the header contains at least single bit error and CLDS drops the packet. Thus $p(Z = ?) = \lambda$. (ii) $Z = 0$ if a packet contains no errors in the header but contains at least a single bit error in the data payload. Thus $p(Z = 0) = (\delta - \lambda)$, i.e. $\delta - \lambda$ represents the probability of a corrupted packet being delivered to a CLD/CLDS channel receiver. (iii) $Z = 1$ if neither the header nor the data payload contain even a single erroneous bit. Thus $p(Z = 1) = (1 - \delta)$.

- ε : The conditional probability of a bit in the data payload being in error given that the checksum CK-HDR is satisfied and checksum CK-DATA has failed. Given a corrupted packet at a CLD/CLDS receiver, ε represents the probability of having a random bit selected from that packet to be in error, i.e. ε specifically represents the probability of error in corrupted packets relayed to the application layer.
- p: The conditional probability of a bit in the data payload being in error given that the checksum CK-HDR is satisfied. In other words p is the overall probability of bit error in packets that are received at the application layer (i.e. all packets that don't get dropped due to corruption in the header). Also note that $p = \frac{(\delta - \lambda) \cdot \varepsilon}{(1 - \lambda)}$.

Thus,

- The conventional protocol (CON), which is the simplest among the three (CON, CLD, and CLDS), can be represented by a Binary Erasure Channel (BEC) as shown in Figure 4 (a). It is well known [30] that the channel capacity of a BEC is given by $1 - \delta$; hence, the capacity of a conventional protocol is given by

$$C_{CON} = 1 - \delta \qquad (1)$$

- The cross-layer protocol, CLD, can be represented as a cascade of a BEC channel with probability of erasure equal to λ followed by a Binary Symmetric Channel (BSC) with probability of bit error equal to p, as shown in Figure 4 (b). Such a cascade can hence be termed as Binary Symmetric/Erasure Channel (BSEC). It can be easily shown that the channel capacity of such a cascade is given by the product of the channel capacities of the individual channels:

$$C_{CLD} = (1-\lambda)\cdot(1-h_b(p)) \tag{2}$$

where $h_b(p)$ is the entropy of a binary random variable with parameter p.

- The CLDS protocol can be represented by Figure 4 (c). The channel capacity of the cross-layer channel in presence of side information Z is as follows: when $Z=1$ all the bits are transmitted reliably; and in this case, which occurs with probability $(1-\delta)$, the (conditional) capacity is 1. When $Z=?$ all the bits get erased and the conditional capacity is 0 while when $Z=0$ the channel reduces to a BSC with a cross-over probability ε and the conditional capacity is $(1-h_b(\varepsilon))$. Thus the channel capacity of CLDS is given by:

$$C_{CLDS} = (1-\delta)+(\delta-\lambda)\cdot(1-h_b(\varepsilon)) \tag{3}$$

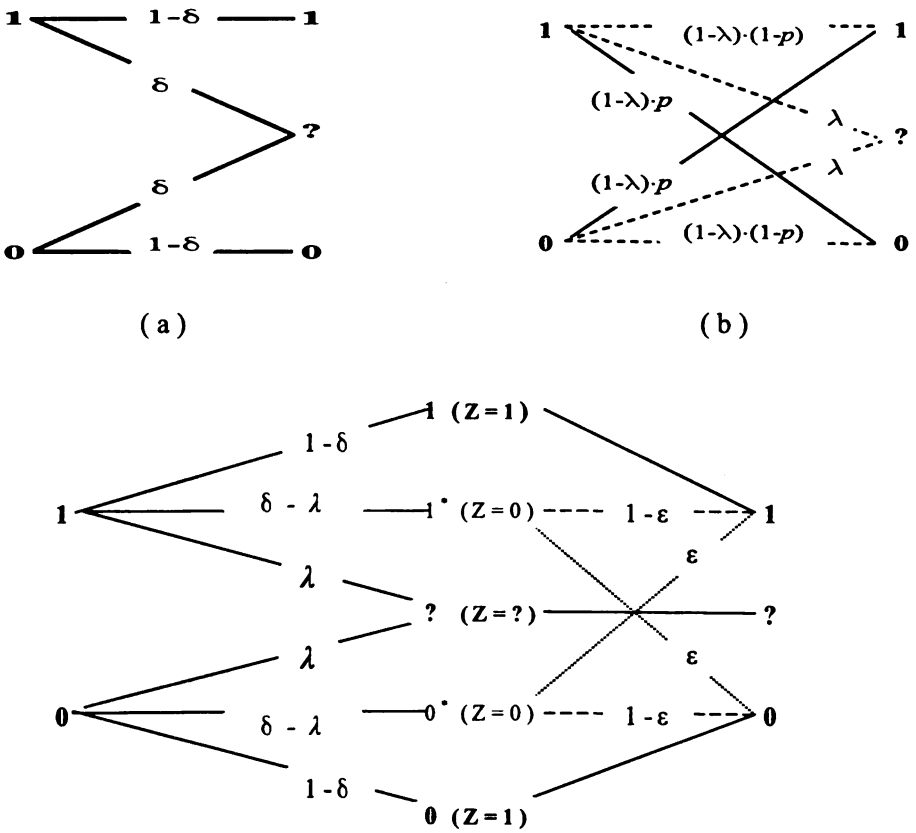


Figure 4 (a) Binary Erasure Channel (BEC) representing the UDP channel (b) Hybrid Binary Symmetric/Erasure Channel (BSEC) (c) BSEC with Side information Z.

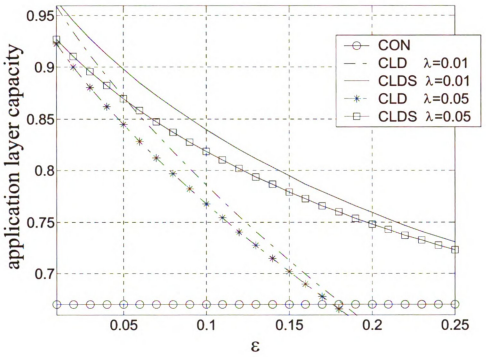


Figure 5 Comparison of channel capacity for CON, CLD and CLDS, $\delta = 0.33$.

Figure 4 provides a comparison between the capacities for various channel conditions. Some important conclusions that can be derived from the above Figure are as follows:

- There exist a variety of channel conditions under which the performance of any cross-layer scheme is better than CON. In Figure 4 observe performance for $\varepsilon \leq 0.17$.
- There exist channel conditions under which the performance of CON is better than CLD. In Figure 4 observe performance for $\varepsilon \geq 0.18$.
- The performance of CLDS is always better than either of CON or CLD.

- The relative performance of CLDS over CLD increases as corruption level increases.

Above conclusions are formally proved in a publication associated with this work [19]. The same paper provides extensions of the above conclusions to multi-hop communication. We encourage the reader to refer to [19] for additional details.

Can we do better?

The above discussion clearly exhibits the utility of side-information from a capacity perspective. However, the CLDS scheme presented employs a rather simplistic side-information In the CLDS mechanism all the corrupted packets are treated equal, however in practice the channel impairments experienced by each packet may be distinct. So a natural question, to ask, is whether we can identify methodologies that allow us to differentiate between corrupted packets at finer resolution.

The answer to the above question is a YES!. There are multiple methods of providing additional side-information, however one effective method that we have identified is based on the observation that: Radio hardware used for reception of 802.11b frames is capable of recording and associating a SSR to each received frame. Thus the relationship of SSR to BER can be used to provide a robust CSI to the cross-layer error recovery mechanism.

For notational convenience let's name the CLDS scheme that utilizes SSR as a side-information as an SSR_aware scheme, while the CLDS scheme that does not utilize the SSR information can be referred as SSR_unaware scheme. Let $f(SSR)$ denote the probability of receiving a packet with a particular SSR indication. For a packet received with a particular SSR indication let $\delta(SSR)$ represent the probability of a packet being corrupted and $\varepsilon(SSR)$ represent the probability of a bit error in the corrupted packet. Also, for convenience let's assume that we do not see any packet drops due to header corruption. Then from equation (3) we already know that the capacity of the SSR_unaware scheme is given by:

$$C_{SSR\_unaware} = \left(1 - \sum_{ssr} f(SSR)\delta(SSR)\right) + \left(\sum_{ssr} f(SSR)\delta(SSR)\right)\left(1 - h_b\left(\sum_{ssr} f(SSR)\delta(SSR)\varepsilon(SSR)\right)\right) \quad (4)$$

where $\delta = \sum_{ssr} f(SSR)\delta(SSR)$ & $\varepsilon = \sum_{ssr} f(SSR)\delta(SSR)\varepsilon(SSR)$ represent the overall probability of a packet being corrupted and the overall probability of bit error in a corrupted packet respectively.

The capacity of an SSR_aware scheme can be found as:

$$C_{SSR\_aware} = E_{SSR}\left[C_{CLDS}\left(\delta(SSR), \varepsilon(SSR)\right)\right] \quad (5)$$

where, $C_{CLDS}(SSR)$ is the CLDS capacity for a particular SSR value, which can be

obtained by employing equation (3) for channel parameter values obtained for a particular SSR value, i.e.

$$C_{CLDS}\left(\delta(SSR),\varepsilon(SSR)\right)=\left(1-\delta(SSR)\right)+\delta(SSR)\cdot\left(1-h_b\left(\varepsilon(SSR)\right)\right)$$

Thus, we obtain:

$$C_{SSR\_aware}=(1-\delta)+\sum_{SSR} f(SSR)\cdot\delta(SSR)\left(1-h_b\left(\varepsilon(SSR)\right)\right) \tag{6}$$

At this stage it should be noted that, equation (4) can be rewritten as

$$C_{SSR\_unaware}=C_{CLDS}\left(E_{SSR}\left[\delta(SSR)\right],E_{SSR}\left[\varepsilon(SSR)\right]\right). \tag{7}$$

Thus by convexity of the capacity function [30] and comparing (5) and (7) we have:

$$C_{SSR\_aware}\geq C_{SSR\_unaware} \tag{8}$$

The above deduction can be repeated for a variety of feasible side-information mechanisms. Thus at this stage, at least from capacity perspective, the motivation for developing CLDS protocols has been clearly established.

2.2.2. FEC for CLD/CLDS Protocols

The FEC schemes we use are realized by interleaving codewords across packets. Thus an FEC scheme can be completely characterized by N, the packet block length, the code length n, code rate R and the packet size s in bits and the symbol size b (in bits) on which the code has been implemented. Thus each FEC block consists of

$(N.R)$ message packets and $(N \cdot s)/(b.n)$ codewords. The FEC decoding algorithm uses erasure decoding when used in conjunction with traditional protocols and erasure-error decoding when used with cross-layer protocols.

For Reed-Solomon based FEC the symbol size is typically the size of a byte i.e. $b = 8$. With the CON scheme we employ the conventional FEC decoding as discussed in [27]. For CLD and CLDS schemes, we modify the algorithm, the decoding algorithm used for simultaneous erasure and error decoding of RS code is the Berlekamp-Massey Algorithm (see e.g. [25], [26] for background).

For cross-layer schemes, a significant improvement can be achieved by employing LDPC based FEC schemes. For Low-Density Parity Check (LDPC) code based FEC schemes the symbol size is typically chosen to be the size of a bit i.e. b=1. The decoding algorithm used for LDPC is based on the Log-Likelihood Ratio (LLR) domain (see e.g. [28]-[29] for relevant background on LDPC codes) implementation of the sum-product decoder.

2.2.2.1. Side-Information for Improved decoding efficiency:

Performance over CLDS can be improved by taking advantage of the side-information provided by CK-DATA for FEC decoding. This can be done as described below

A. FEC block decoding failure: On occurrence of a block decoding failure the channel decoder is unable to recover all the dropped/corrupted packets. However as the FEC block is systematic, message packets that can be identified as uncorrupted can still be forwarded to the eventual application. In a cross layer design like CLD as we do not have information about CK-DATA, if a decoding failure is encountered, the entire packet block is dropped. However in a CLDS scheme the CK-DATA information is used to forward the correctly received message packets to the application layer.

B. Apriori estimates for improved FEC decoding: CK-DATA side-information can be utilized to acquire improved apriori estimate of channel impairments and thus improve FEC decoding:

   a. In case of RS based FEC, we use this information rather simplistically; if the number of packets for which CK-DATA fails is less than the total number of redundant packets, then even if all the corrupted packets are treated as erasures the entire FEC packet block is recovered by pure erasure decoding. Thus the CLDS scheme can switch to the pure erasure decoding whenever possible to avoid decoding failure due to a high corruption level in the packets.

b. In the LDPC decoding algorithm the LLR $L(c_i)$ associated with code bit c_i is initialized at the start of the decoding on the basis of the received bit y_i and an apriori assumption of the bit error probability of the ith bit being P_e as

$$L(c_i) = (-1)^{y_i} \log\left(\frac{1-P_e}{P_e}\right) \tag{9}$$

The message-passing algorithm (log-domain sum-product algorithm) is then iteratively used to update $L(c_i)$. A bit is decoded to 1 if $L(c_i) \leq 0$ and 0 if $L(c_i) > 0$. An accurate initialization of LLR plays a key role in the performance of LDPC codes and thus improved CSI can help in improving the LDPC decoding, in particular if a packet is received without any errors then $L(c_i)$ corresponding to the bits in that packet is obtained by setting $P_e = 0$ and similarly if a packet is dropped $L(c_i)$ is obtained by setting $P_e = 0.5$. Thus we initialize the LLR for various cross-layer protocols as described below:

A. CLD:

$$\begin{aligned} L(c_i) &= (-1)^{y_i} \log\left(\frac{1-p}{p}\right) && \text{if bit is not erased,} \\ L(c_i) &= 0 && \text{if bit is erased} \end{aligned} \tag{10}$$

B. CLDS:

$$
\begin{aligned}
L(c_i) &= 0 && \text{if bit belongs to a dropped packet} \\
L(c_i) &= \infty && \text{if bit belongs to an uncorrupted recieved packet} \\
L(c_i) &= (-1)^{y_i} \log\left(\frac{1-\varepsilon}{\varepsilon}\right) && \text{if bit belongs to a corrupted recieved packet}
\end{aligned}
\tag{11}
$$

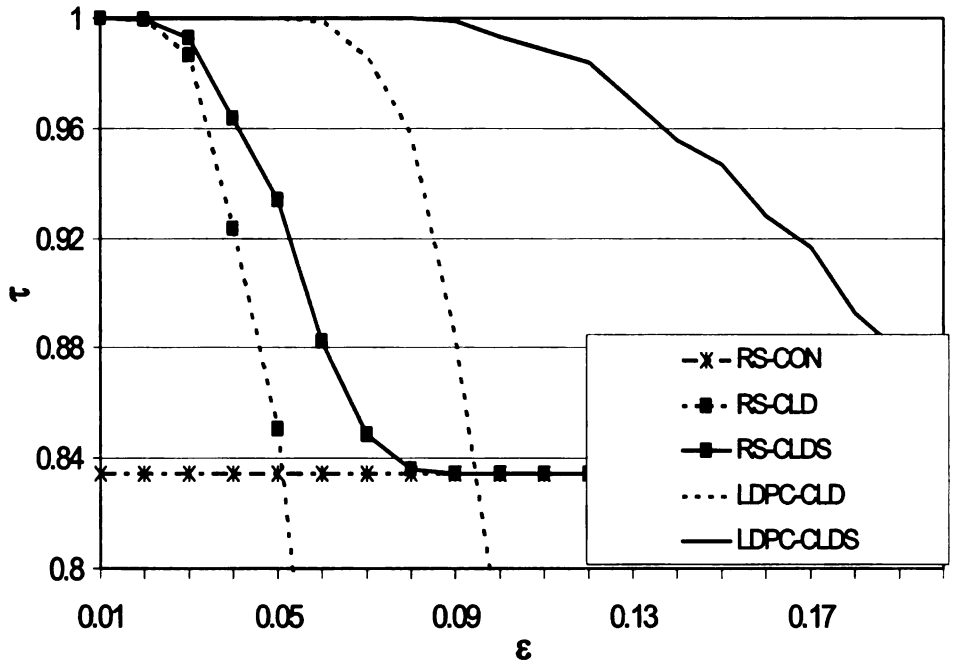


Figure 6 Message packet throughput τ, after channel decoding, for RS/LDPC based FEC schemes over CON/CLD/CLDS. Channel conditions are given by $\delta = 0.33$, $\lambda = 0.01$.

Figure 6 shows the performance of RS/LDPC based FEC schemes in which each FEC block consists of 30 packets (20 message, 10 parity/redundancy) of 500 bytes. Performance trends predicted by the capacity deductions are observed in FEC performance also. In particular it can be seen that the performance of CLDS schemes are the best. Even though CLD schemes can perform better than CON, as the corruption level increases the performance of CLD is worse than CON.

With regards to the performance of a specific FEC scheme, it can be clearly seen that LDPC based FEC schemes can provide significantly improved performance when compared with RS based FEC. This improvement at least in part should be attributed to the fact that we employ a soft-decoding algorithm for LDPC based FEC. When the corruption level is high, the performance improvement of LDPC with side-information is significantly better than all other combinations. Thus, in practical deployments side-information may play a pivotal role in increasing the robustness of video communication architecture.

Just as had been commented in the capacity deduction, we highlight that the CLDS scheme explored above is a simplistic one. Improved performance can be obtained by including better indicators of link quality. In particular in SSR_aware CLDS scheme we initialize the LDPC algorithm as follows:

$$\begin{aligned} L(c_i) &= 0 && \text{if bit belongs to a dropped packet} \\ L(c_i) &= \infty && \text{if bit belongs to an uncorrupted packet} \\ L(c_i) &= (-1)^{y_i} \log\left(\frac{1-\varepsilon(X)}{\varepsilon(X)}\right) && \text{if bit belongs to a corrupted packet} \\ & && \text{with with SSR indication X} \end{aligned} \tag{12}$$

CHAPTER 3
MULTI-TIER MODEL FOR BER PREDICTION

In this chapter, we describe the method of collecting residual error traces which represents various and realistic wireless channels; and we analyze MAC-to-MAC bit-error characteristics of these wireless MAC-layer channels. Based on our analysis, we develop a multi-tier model (MTM) for BER estimation and prediction. Here it is important to explain what we mean by BER estimation and prediction. In contemporary wireless networks [43], [44], a wireless receiver's link layer performs a packet-level checksum to determine whether a packet is error-free or corrupted. Obviously, the BER is zero if a packet passes the checksum, thereby eliminating the need for BER estimation. For a packet that fails the checksum, the receiver does not know how many bits within the packet are corrupted. This knowledge is important analytically and in practice, and it has a direct implication on the effective channel capacity and the choice of certain cross-layer wireless protocols [45]-[47]. Thus for a corrupted packet, one needs a scheme that can render an accurate estimate of the number of errors in the packet. Once the BER of

the current packet is estimated, the next problem is to predict the number of errors in the following packets.

The proposed MTM leverages Signal to Silence Ratio (SSR) indication and checksum side-information to estimate the BER in the current packet and to predict the BER in future packets.

3.1. Trace Collection

Wireless network simulators often rely on channel models to recreate the complexities of the real-world. Channel models that fully take into account channel memory that produces persistence and clustering of errors in transmissions are often based on Markov chains [8]-[10]. Unfortunately, the complexity of Markov models grows as $O(n^2)$ with increase in the memory length n of wireless channel errors. To provide an appreciation of the increasing richness of the inhabitants of the 2.4 GHz Industrial-Scientific-Medical (ISM) band, consider that IEEE 802.11b Wireless Local Area Networks (WLAN) [57], microwave ovens and cordless phones all operate at these frequencies. Additional factors which further complicate channel modeling include effects of interference, differences in environment (open space, office, residential, public

etc.) and the question of what constitutes "typical" background traffic and noise. Of course, it is possible to create interference scenarios by placing additional interference sources in simulated settings and hope that the simulator does a good enough job at modeling interference effects. However, the validity of assumptions made about traffic patterns and background noise due to interfering sources is unreliable. Since simulations [58] are often based on simple channel models, the validity and accuracy of simulator results is doubtful. The environments are often simplistic and unrealistic in comparison to what is observed in real-life. To that end, we collect a comprehensive set of residual bit-error traces representing realistic wireless channels and use them in our experiments to verify the performance of our developed schemes.

3.1.1. Data Collection

We collected error traces simultaneously on five IEEE 802.11b wireless receivers (or sniffers). The receivers were located at different places in a lab, while the access point (AP) was placed in a room across a hallway from the receivers to simulate a realistic classroom/office setting [Figure 7]. The receivers' MAC layer device drivers were modified to capture corrupted packets. Each experiment comprised of one million packets with a payload of 1,000 bytes each, i.e., each trace has approximately 1 GB of

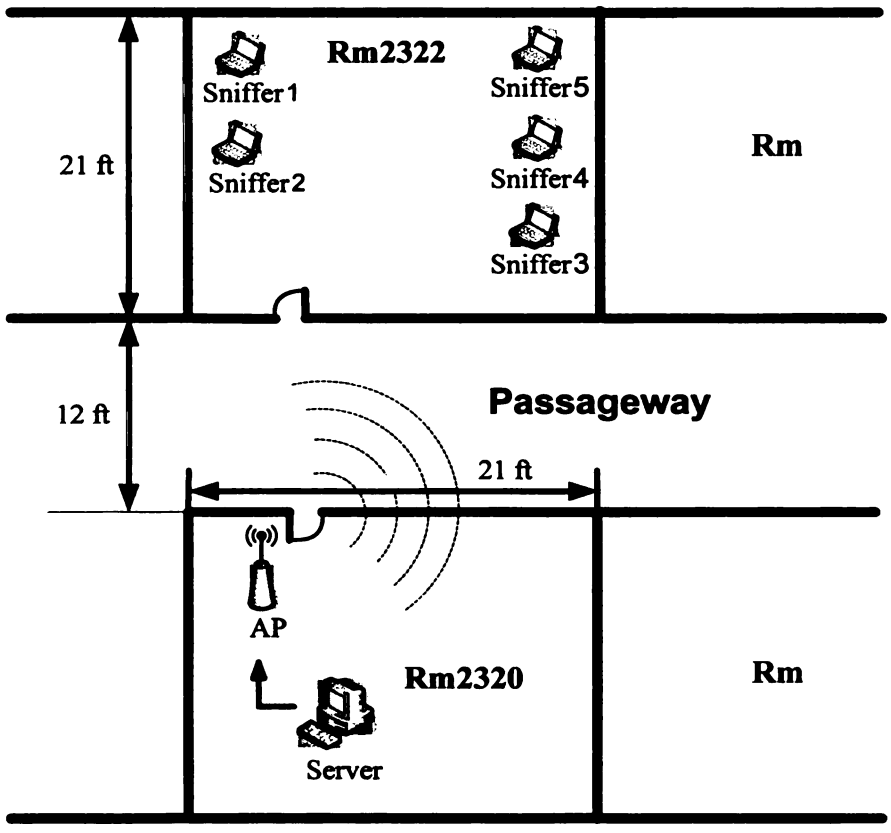


Figure 7 Topologies used for wireless trace collection.

data, which corresponds to approximately 4 and half hours of error trace collection when packets are transmitted at 500kbps. Note that the error traces were collected in a university lab environment so it is not possible to precisely account for movement activity in the hallway. However, even though the receivers were located in the same room, they were place in different locations, and consequently they encountered very different channel state or loss pattern. For instance, it was clearly observed that BER of the traces collected by the receiver 5 in Figure 7, which was placed near by the window (which was close to a street) and surrounded by metal bookshelves, was considerably higher than that of other traces collected by other receivers. Therefore, the error traces can represent the various and realistic channel states of wireless environment.

A wired sender was used to send multicast packets, in which a multicast IP address is defined, with a predetermined payload on the wireless LAN; multicasting disabled MAC layer retransmissions. In addition to a packet's header and payload information, we logged signal to silence ratio (SSR) for each packet. The sender used four different transmission rates which are 500, 750, 900 Kbps and 1 Mbps. Note that a packet rate (packets per second) differs with transmission rates. At the physical layer, the auto rate selection feature of the AP was disabled and for each experiment the AP was forced to transmit at a fixed data rate (2, 5.5 or 11 Mbps). For each trace collection experiment, we

set a transmission rate and a physical layer data rate, and hence 12 different experiments were conducted at different times of day, i.e., 60 error traces were collected.

3.1.2. Average Statistics of the Traces

TABLE I provides some statistics of the traces collected for this study. Since the physical layer robustness decreases with an increase in data rate, the average packet error rate increases with an increase in the physical layer data rate. In particular, the average packet error rate increases from approximately 10% at 5.5 Mbps to almost 40% at 11 Mbps. Since the wireless receivers were placed at different locations, the receivers experienced different packet error rates. The overall minimum and maximum error rates in TABLE I outline that the receivers under consideration were experiencing both good and bad link conditions.

TABLE I Statistics of Traces Used In This Study

Phy. data rate (Mbps)	Avg. PER	Min. PER	Max. PER	Avg. SSR (dB)	Min. SSR (dB)	Max. SSR (dB)
2	5.97%	0.75%	14.31%	14.75	0	34
5.5	9.79%	0.61%	22.74%	15.27	0	32
11	39.5%	10.99%	77.83%	16.51	0	35

TABLE II Error Statistics For Varying SSR Values At 11 Mbps

SSR (dB)	Average Packet-Error Rate	BER of all packets (error-free & corrupted)	BER of corrupted packets
5	0.701	0.0253	0.0361
13	0.6248	0.0157	0.0251
20	0.2166	0.0048	0.0223
26	0.0384	0.0023	0.0591

The average, minimum and maximum SSR values are also shown in TABLE I. Note that the minimum SSR value is zero at all three data rates. From a prior analysis [21] [35], we know this SSR range is of interest for the protocols considered in this paper.

The relationship between SSR values and the channel error rate is also shown in TABLE I. It is easily observed from the second column of TABLE II that packet error rates increase drastically with a decrease in SSR values. In particular, the packet error rate increases by approximately 18% as the SSR decrease from 26 dB to 20 dB. Similarly, there is a packet error rate increase of about 41% between SSRs 13 and 20.

3.1.3. BER Behavior at Different SSR Values

Figure 8 uses the 11 Mbps residual channel to show that the BER behavior of the channel changes considerably with a change in the received packets' SSR values. (For brevity, in this section we only show results for the 11 Mbps channels because results for 2 and 5.5 Mbps channels were similar. In the modeling sections, we show results for all three channels under consideration.) At an SSR of 5 dB, small number of errors in a corrupted packet are fairly infrequent, as shown by the low normalized frequency of errors in the [1,200] range of Figure 8 (a). This is mainly because at such low SSR values, most of the received packets are corrupted, and a large number of bits in these packets are corrupted. As the SSR increases, the skew of the histogram changes and the corrupted packets have fewer bit-errors. This trend can be observed in Figure 8 (b), (c) and (d), which show that the frequency of small number of bit-errors increases with SSR. For instance, at an SSR of 26 dB, almost 25% of corrupted packets have less than five bit-errors. Comparison of Figure 8 (a), (b), (c) and (d) clearly shows that an increase in SSR decreases the mean number of bit-errors in a corrupted packet.

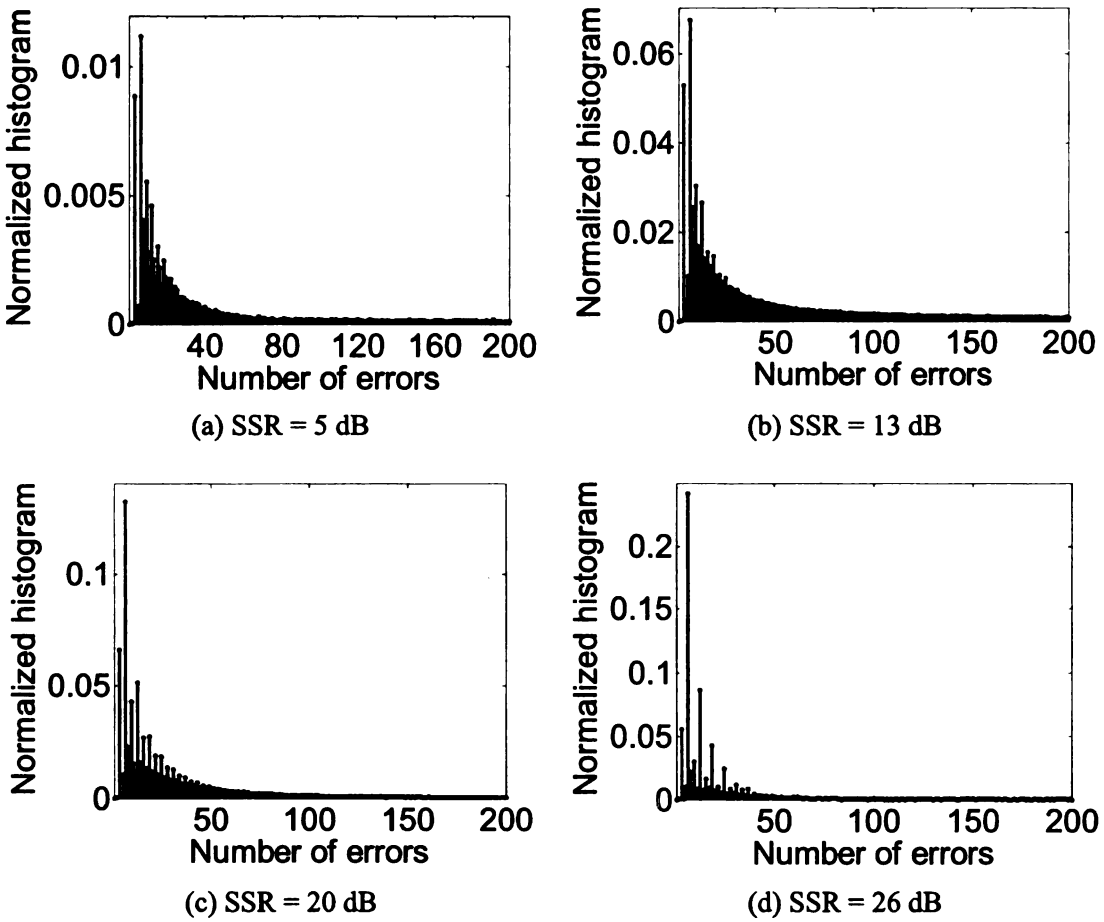


Figure 8 Normalized histograms of the number of bit-errors in a corrupted packet for varying SSR values; histograms are averaged over all 11 Mbps residual traces, and only number of errors between [1,200] are shown.

The relationship between SSR values and the channel error rate is also shown in TABLE II. It is easily observed from the second column of TABLE II that packet error rates increase drastically with a decrease in SSR values. In particular, the packet error rate increases by approximately 18% as the SSR decrease from 26 dB to 20 dB. Similarly, there is a packet error rate increase of about 41% between SSRs 13 and 20. To avoid repetition, we defer discussion on columns 3 and 4 of TABLE II to a later

section.

Based on the results presented so far, we deduce that SSR is a robust and effective side-information of a wireless link's condition. The following section leverages this side-information to accurately estimate and predict the BER of the channel.

3.2. A Markov Model for PER Prediction

We first emphasize an important point highlighted by columns three and four of TABLE II. Note that for low SSR values (i.e., poor link conditions,) the BER computed using all (error-free and corrupted) packets is somewhat similar to the BER computed using only corrupted packets. However, when we compare the BER at higher SSR (20 and 26 dB in TABLE II), the BER computed using all packets is orders of magnitude different from the BER computed using corrupted packets only.

The BER difference for high SSR value is very stark because: (i) most of the received packets are error-free; and (ii) the relatively small number of corrupted packets has relatively-high BER. In general, and based on our extensive study of this issue, it became evident that the relatively-small number of corrupted packets, at relatively-high SSR/SNR values, provides good (yet conservative and slightly biased) estimates of BER.

Meanwhile, including the relatively large number of error-free packets in the BER estimates (especially at high SSR/SNR values), provides highly optimistic (very biased) estimates of BER in actually corrupted packets. A surprising result shown in TABLE II is that the BER in the corrupted packets is very high for high SSR values. We observed that while the total number of corrupted packet at high SSR values is quite small, the corrupted packets contained a large number of bit-errors. We believe that this phenomenon occurs because the corrupted packets at high SSR values are mostly due to packet collisions, and therefore these packets contain are highly corrupted.

High SSR values are quite important because over real-life wireless channels, many packets are received with high SSR values. (For instance, as shown in TABLE I, we observed maximum SSR values of 34, 32 and 35 dB for the 2, 5.5 and 11 Mbps channels, respectively.) Therefore, we propose that at the first tier, an accurate predictor should solely focus on packet-error prediction. In turn, BER prediction should only be done for packets which are predicted to be corrupted by the packet-error model.

3.2.1. Correlation of the Packet-Error Process

To determine an accurate first-tier packet-error model, we first analyze the correlation coefficient of the packet-error process. Let X_n denote a discrete binary packet-error

random process. Thus for a given n, $X_n \in \{0,1\}$ is a binary random variable with $X_n = 0$ and $X_n = 1$ respectively representing that packet n is error-free and corrupted.

For each physical layer data rate, we treat the packet-error sequences obtained from the traces as realizations of the packet-error process for that data rate. Then the random process' autocorrelation function is [39]

$$R[\eta] = \mathrm{E}\{X_0 X_\eta\}, \tag{13}$$

where $\mathrm{E}\{.\}$ is the sample expected value computed using the process realizations. Using the autocorrelation, we compute the sample correlation coefficient of a packet-error process:

$$\rho[\eta] = \frac{\mathrm{E}\{X_0 X_\eta\} - \mathrm{E}\{X_0\}\mathrm{E}\{X_\eta\}}{\sigma_{X_0}\sigma_{X_\eta}}, \tag{14}$$

where σ_X represents the sample standard deviation of random variable X. This correlation coefficient provides a normalized measure of the amount of correlation present in the data. Markov chains are generally used to model processes with low-correlation values, whereas highly correlated processes are typically modeled using heavy-tailed models [48].

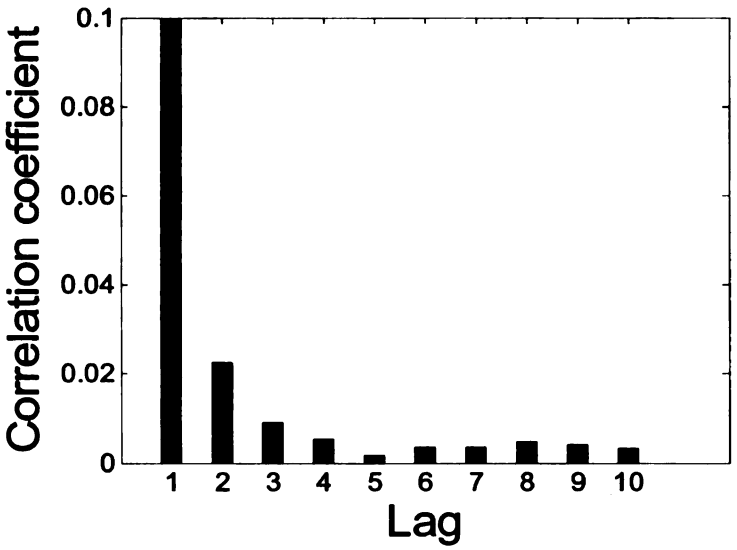


Figure 9 Sample autocorrelation coefficient of an 11 Mbps trace.

Figure 9 shows the correlation coefficient of a sample 11 Mbps residual trace. (Correlation decays for other traces were similar and are therefore skipped.) Clearly, the packet-error process' correlation shows an exponentially decaying trend. The correlation function drops to and stays at an insignificant value at lags of more than three. Thus we deduce that a 3rd order Markov chain can be used to model the packet-error process.

3.2.2. The 3rd order Packet-Error Markov Model

To predict packet errors, we define a discrete-time Markov chain such that each state of the chain corresponds to one of the $2^3 = 8$ possible combinations of three binary symbols; each binary symbol represents an error-free or corrupted packet. Transition

probabilities of the 3rd order Markov chain are computed by sliding a 3-bit window over the wireless traces and by observing the frequency of a binary pattern $\underline{x} = [x_1 x_2 \ldots x_k]$ followed by another bit-pattern $\underline{y} = [y_1 y_2 \ldots y_k]$, for all patterns $\underline{x}$ and $\underline{y}$.

The 3rd order Markov chain model's transition probabilities are computed from the packet-errors observed in the traces of this study. Since the BER behavior changes drastically with respect to the physical layer data rate, for each data rate (2, 5.5 and 11 Mbps), we train a different 3rd order Markov model. After model training, given a packet's checksum side-information (pass/fail), we use the Markov chain to predict whether the next packet will be error-free or corrupted. The prediction process is conducted as follows. From any given state of the 3rd order Markov chain, the process can transit either to a state with an upcoming error-free packet or to a state with an upcoming corrupted packet. There are two transition probabilities associated with these two possible transitions, say p and $1-p$. To predict the checksum of the next packet, we treat these probabilities as a Bernoulli random variable, with probability of success p corresponding to the probability that the next packet will be error-free, and the probability of failure $1-p$ corresponding to the probability that the next packet will be corrupted. Furthermore, from the checksum of the next packet, we know whether or not our prediction was correct. If the prediction was correct then the predicted state of the

Markov chain is used for the subsequent prediction. Otherwise, the Markov chain's current state is changed to the opposite of what was predicted.

As explained earlier, BER estimation is only invoked for corrupted packets. In the following section, we explain the BER estimation using SSR values.

Figure 10 The multi-tier BER prediction model.

3.3. Packet-Level Model for SSR Prediction

Once a corrupted packet is predicted by the tier 1 model, at the second tier we use the SSR of the received packet to predict the SSR of a future packet. For each possible SSR value (which ranges between 0 and 100 dB), we maintain a discrete conditional probability distribution of the next SSR values. Thus a conditional probability value P_{ij} yields the probability that if the SSR value of the current packet is i dB then the next packet will be received at an SSR of j dB. The conditional probability distributions are computed using corrupt packets observed in the traces.

3.4. BER Prediction using BSC Models

The second tier model predicts the SSR of the next packet using a conditional probability distribution. In accordance with prior discussions and as shown in TABLE II, the BER of the channel changes with respect to the channel SSR. In general, we observed a non-linear relationship between SSR and BER. Therefore, after predicting the next packet's SSR, we employ a binary symmetric channel (BSC) model to predict the BER of the next packet. While we acknowledge that the BSC model is somewhat simplistic for the present problem, we observed that this simple model can provide quite accurate

BER prediction, especially at low physical layer data rates. Hence, we use the crossover probability of the BSC model corresponding to the predicted SSR as the BER estimate of the next packet.

3.5. Performance Evaluation of the MTM

We now compare the performance of the proposed MTM model with two leading predictors, namely the Yule-Walker (Y-W) predictor [39] and finite-state Markov Chain (FSMC) predictor [49],[50].

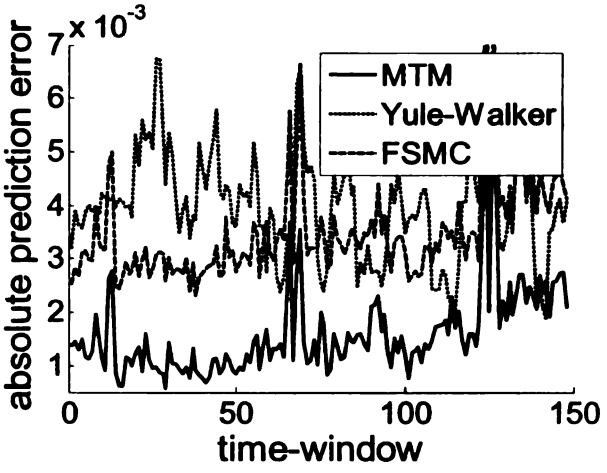
For the Y-W based BER prediction, we first use SSR values of the last k packets to predict the SSR of the next packet. More specifically, let $S_{n-k+1}, S_{n-k+2}, \ldots, S_n$ be the SSR values of the last k packets. The Y-W predictor predicts the next SSR value, S_{n+1}, using a linear filter of the form:

$$S_{n+1} = \sum_{i=0}^{k-1} h_i S_{n-i} .$$

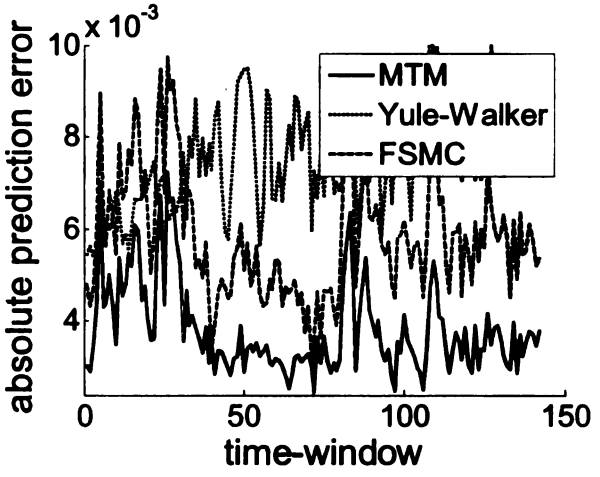
The coefficients of the filter are computed as:

$$\begin{bmatrix} h[0] \\ h[1] \\ \vdots \\ h[k-1] \end{bmatrix} = \begin{bmatrix} R[0] & R[1] & R[2] & \cdots & R[k-1] \\ R[1] & R[0] & R[1] & \cdots & R[k-2] \\ & & \vdots & & \\ R[k-1] & & \cdots & R[1] & R[0] \end{bmatrix}^{-1} \begin{bmatrix} R[1] \\ R[2] \\ \vdots \\ R[k] \end{bmatrix},$$

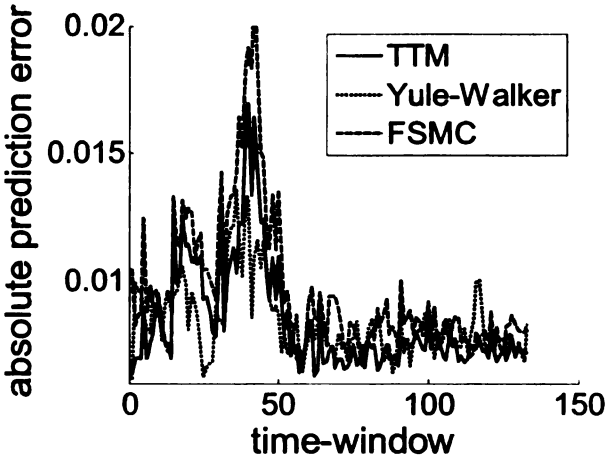
where $R[.]$ is the autocorrelation function of (13). We experimented with different values of k, and obtained the best Y-W prediction performance for $k = 5$. Once the SSR is predicted using the above equations, we use the tier 3 BSC models of Figure 10 to predict the BER of the next packet.



(a) Physical layer data rate = 2 Mbps



(b) Physical layer data rate = 5.5 Mbps



(c) Physical layer data rate = 11 Mbps

Figure11 Absolute error in BER prediction; time-window length $\tau = 50$ seconds, results are averaged over all the traces.

The FSMC predictor [49] employs a Markov chain model of SNR values to predict future SNR values. The FSMC model of SNR values is designed specifically for a fading channel and the model assumes the availability of SNR values for each symbol. The FSMC partitions SNR values into N disjoint intervals, where each interval represents an FSMC state. Different state SNR partitioning strategies have been proposed in prior literature [49],[50]. It has also been shown that under fading conditions, an FSMC model in state i can only transit to its neighboring states $i-1$ and $i+1$. After predicting the next SNR value, BSC models are used to predict the BER of the next packet.

Since on a residual channel we only have packet-level SSR information, the previously-proposed SNR partitioning algorithms are not directly applicable here. Moreover, at times we observed large fluctuations in SSR values. These fluctuations made the constraint of transitioning only to the neighboring states impractical. Therefore, we modify the original FSMC model [49] such that the model can transit from any current SSR value to any other SSR value. For fair comparison with the MTM, we place each SSR value into a different FSMC state.

To clearly show the accuracy of a BER predictor without short-term biases, we compare the predicted and actual BERs in non-overlapping time-windows of length τ

TABLE III Average Absolute Error of BER Predictors.

Phy. data rate (Mbps)	**Three-Tier Model**	**Yule-Walker**	**Finite-State Markov Chain**
2	0.0015	0.0042	0.0028
5.5	0.0044	0.0075	0.0057
11	0.0094	0.0097	0.0101

seconds. In each time-window, we compute the absolute value of the difference between the predicted and actual BERs. This difference is henceforth referred to as absolute prediction error. Clearly, smaller prediction error implies higher prediction accuracy.

Figure11 compares the performance of the proposed model with Y-W and FSMC predictors for $\tau = 50$ seconds; that is, each point shown in Figure11 is the average prediction error in a 50 second time-window. It can be observed that at 2 and 5.5 Mbps the proposed model provides consistently better performance than Y-W and FSMC predictors. At 11 Mbps, the accuracies of all the predictors are comparable. Performance of the MTM at 11 Mbps is less convincing because the bit-error characteristics of the 11 Mbps channel are quite different from 2 and 5.5 Mbps. Specifically, prior studies have shown that 11 Mbps bit-errors exhibit long-range dependence, and therefore multi-scale models are required to characterize these bit-errors [51]. We are currently incorporating

such models in the multi-tier framework to improve BER prediction at 11 Mbps.

In TABLE III, we compare the average accuracies of the predictors under consideration. For the results of this table, the absolute prediction error was averaged over the entire trace. It can be clearly seen that average prediction accuracy of the MTM is consistently higher than both Y-W and FSMC predictors. Thus, irrespective of the physical layer data rate, on-average the MTM renders higher prediction accuracy than both Y-W and FSMC predictors.

CHAPTER 4
RATE PREDICTION AND TUNING

In this chapter, we develop a rate adaptation architecture using the two main contributions: channel capacity prediction and optimal rate tuning. Note that the "channel estimator" in the client and the "rate tuner" in the server shown in Figure 1 employ the channel estimation and tuning schemes developed in this chapter, respectively.

For channel capacity prediction, BER is estimated from the crossover probability, ε, of a BSC model that is inferred from the packet SSR values. The residue-error entropy, $H_b(\varepsilon)$, for a time-window is then calculated using two methods: (i) $H_b(\varepsilon)$ is estimated by the average of the instantaneous entropies (entropy of each packet) over the time-window ($CLDS_1$), or (ii) $H_b(\varepsilon)$ is estimated by the entropy of the average BER over the time-window ($CLDS_2$). These entropy estimates are used as the prediction for the next time-window. Optimal rate tuning is achieved by finding the rate that provides the maximum PSNR over the distribution of the *channel prediction error* process.

4.1. Limitations and Optimal Rate Adaptation Period

In this chapter we consider the following simplifying assumptions: (i) the channel code achieves the capacity (i.e., an ideal channel coder), and a block of packets cannot be recovered at the client if it is coded with an overestimated rate, i.e., a channel coding rate that exceeds channel capacity; (ii) A video encoder provides a bitstream having a bitrate that is exactly the same as the required bitrate, and the bit stream renders PSNR value [38] according to the bitrate. With (i) and (ii), we realize the architecture where for video rates that cannot be supported by the underlying channel capacity, the video quality reduces to zero (i.e., zero PSNR value). Note that without the assumptions (i) and (ii) (i.e., a realistic channel coder or video encoder was used), we should carefully take the performances of the channel coder and video encoder into consideration in finding the optimal rate. Therefore, the above assumptions were made to focus only on the performance of our proposed scheme.

In addition, the time-window for the experiment is selected such that the variation in link-quality from one window to the next is minimized. Allan Variance is a measure which allows us to methodically determine such an optimal size of the window [40], and it computed as follows:

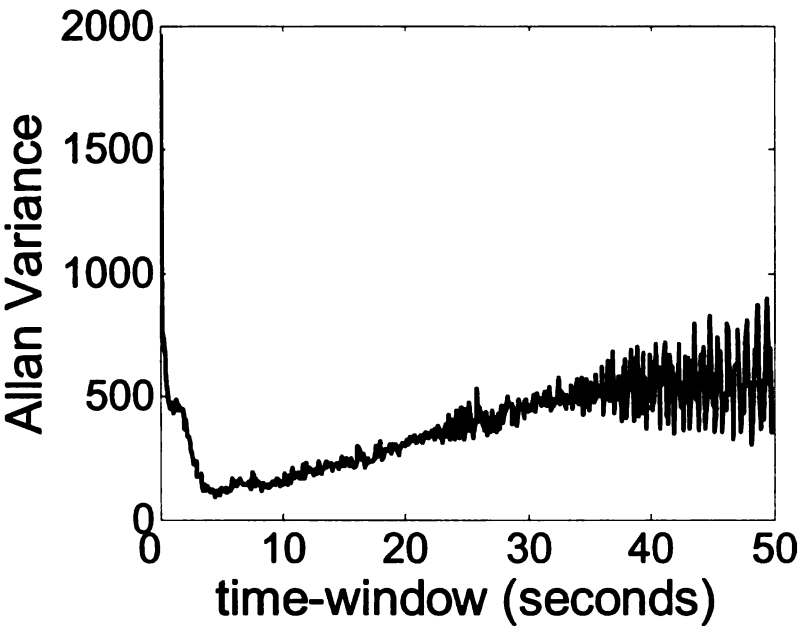


Figure 12 Allan Variance as function of time window.

$$AVAR^2(\tau) = \frac{1}{2(k-1)} \sum_{n=1}^{k} (C(\tau)_{n+1} - C(\tau)_n)^2 \tag{15}$$

where $AVAR^2(\tau)$ is the Allan Variance as a function of averaging time, τ ; $C_n (= 1 - \frac{1}{m} \sum_{i=1}^{m} H_b(\varepsilon_i))$ is the average channel capacity of the measurement in time window n; and k is the total number of time windows. Note that τ in equation (15) is related to the number of packet, m. However, different transmission rates lead to different packet rates, and hence the number of packets, m, in a time window differs with transmission rates. For this reason, we expressed equation (15) in terms of τ, with which we can represent a common time interval for any transmission rate.

For all traces that we have considered, the Allan Variance of BER is minimal for a time-window size of 5 seconds [Figure 12]. Hence, in this study we use the time window with a size of 5 seconds.

4.2. BER and Channel Capacity Estimation

As shown in TABLE II, the BER of the channel changes with respect to the channel SSR. In general, we observed a non-linear relationship between SSR and BER. Here, we employ a BSC model, which utilizes each SSR of a packet to estimate the BER over a given block of packets. While we acknowledge that the BSC model is somewhat simplistic for the present problem, we observed that this simple model can provide quite accurate BER prediction; and more importantly it provides a conservative estimate for the channel capacity due to the lack of a memory model in the channel.

We partitioned the collected traces into training and test data. Next, with training data we define bins over the entire SSR range and determine the average BER for each bin. Note that we regard the average BER as the crossover probability in Binary Symmetric Channel (BSC) model [21], [33], [35]. Hence, a packet renders a BER estimate according to its SSR. During the testing phase we use these estimates of BER to determine channel

capacity over the time window (or the number of packets, m for a given transmission rate) that is defined in equation (15). The channel capacities are calculated as follows:

$$\tilde{C}_n^{CLDS_1} = 1 - \frac{1}{m}\sum_{i=1}^{m} H_b(\tilde{\varepsilon}_i), \tag{16}$$

$$\tilde{C}_n^{CLDS_2} = 1 - H_b(\frac{1}{m}\sum_{i=1}^{m} \tilde{\varepsilon}_i), \text{ and} \tag{17}$$

$$\tilde{C}_n^{CON} = 1 - \frac{1}{m}\sum_{i=1}^{m} Z_i = 1 - PER \tag{18}$$

where $\tilde{\varepsilon}_i$ represents the channel BER estimate for packet i, Z_i is a binary variable representing the status of the checksum of packet i ($Z_i = 1$ if checksum fails). Equation (16) is the capacity estimate $CLDS_1$ computed based on the average of the instantaneous per-packet error-process entropy in a block of m packets, equation (17) is the capacity estimate $CLDS_2$ computed based on the error-process entropy using the average error of packets over the same m-packet block, and equation (18) is the estimate of channel capacity under CON protocols.

4.3. Channel Capacity Prediction

Prior studies have indicated that bit-errors have high temporal correlations, especially in 802.11b wireless networks [34]–[36]. In Figure 13, the correlation coefficients of a number of traces are shown and calculated on the basis of the channel capacity process. The correlation coefficient is computed as

$$\rho = \frac{E[C_n C_{n+1}] - E[C_n]E[C_{n+1}]}{\sqrt{\text{var}[C_n] \cdot \text{var}[C_{n+1}]}}, \tag{19}$$

where $E[.]$ and $\text{var}[.]$ are the sample mean and the sample variance functions. Figure

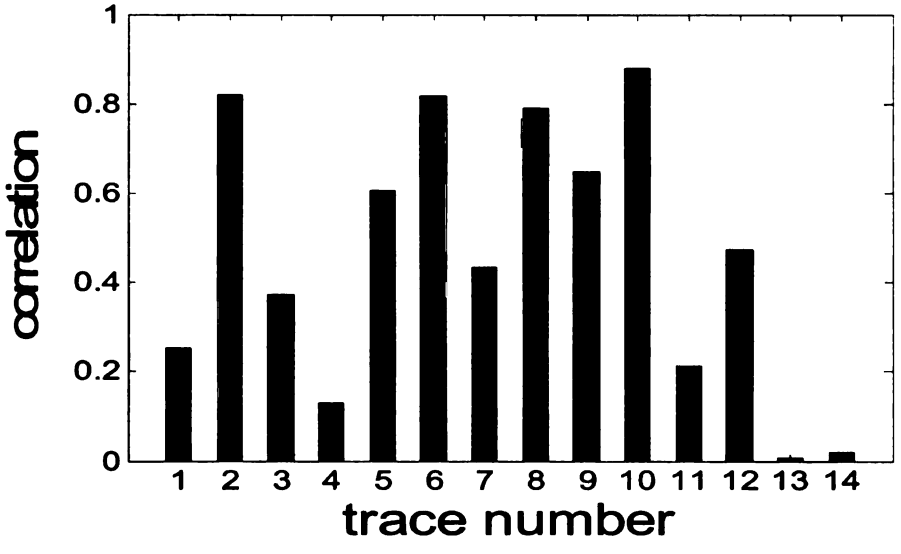


Figure 13 Temporal correlation in channel capacity for 11 Mbps traces (traces from 1 to 5 collected with transmission rate at 500, 6 to 7 at 750, 7 to 10 at 900, and 11 to 14 at 1024Kbps).

13 clearly exhibits the existence of temporal correlation that is non-negligible in all traces, and it is quite significant in most traces. This correlation can be taken advantage of to predict the channel capacity for the next time-window. We exploit this correlation by using the channel capacity estimate of the current packet as an estimate for the next packet's channel capacity:

$$\hat{C}_{n+1} = \tilde{C}_n, \tag{20}$$

which deduces the *channel prediction error,* $e_{n+1} = C_n - \hat{C}_n \simeq \tilde{C}_n - \hat{C}_n = \hat{C}_{n+1} - \hat{C}_n$.

Although we have considered other optimum predictors (e.g., Yule-Walker [39]), our simulation results demonstrate that the above simplistic prediction in conjunction with the optimal rate tuning (described below) provides significant improvement over conventional protocols. Furthermore, the prediction performance of equation (20) is very similar to the optimum Yule-Walker predictor [39] [TABLE III]. Consequently, we focus the remainder of this paper (in the context of the proposed rate prediction architecture) on the predictor determined by equation (20) due it is minimal complexity.

4.4. Optimal Rate Tuning

We need to adhere to a rate that is strictly below channel capacity to avoid excessive

packet drops. Therefore, if the predicted channel capacity is directly employed, there is a good likelihood that the predicted capacity (as a random variable) may exceed the actual channel capacity. A natural workaround to this problem is to make the predicted capacity more conservative by subtracting a small offset Δ from the channel capacity prediction, $\hat{C}_n - \Delta$. Such a strategy can, however, results in considerable long-run bandwidth wastage, and therefore judicious selection of the Δ parameter is extremely important. We propose to find the "optimal" value of Δ (leading in turn to the optimal video rate) such that the average video peak signal-to-noise ratio (PSNR) over some period of time (or over a set of blocks of packets) is maximized:

$$\overset{*}{\Delta}_n = \arg\max_{\Delta} \left[\frac{1}{N} \sum_{n=1}^{N} (I(C_n > \hat{C}_n - \Delta)) \cdot Q((\hat{C}_n - \Delta) \cdot T) \right] \tag{21}$$

$$\text{and } \overset{*}{R}_n = \hat{C}_n - \overset{*}{\Delta}_n$$

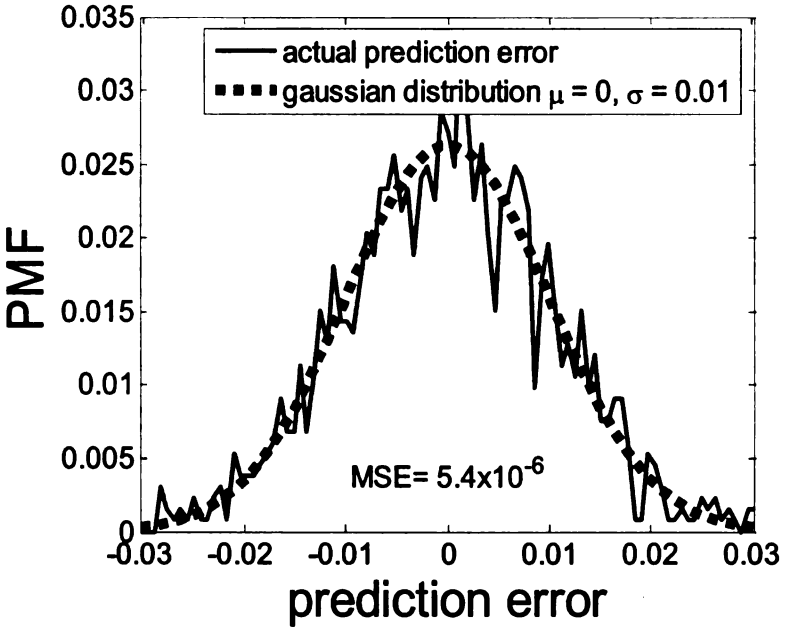


Figure 14 The *channel prediction error* process (e) resembles the probability distribution of Gaussian.

where, C_n and $\hat{C}_n$ are the actual and predicted channel capacities, and $I(\cdot)$, $Q(\cdot)$ and T respectively represent an Indicator function[3], an RD (video quality) function and a transmit rate[4]. Thus the above objective function assumes a rather simple binary quality-indicator that forces the estimated PSNR value to zero when the rate exceeds the capacity. Note that based on the present objective function defined in equation (21), the "optimal" rate can be computed only when the overall statistics and the actual channel capacity values are available. Since the optimal rate determined using (21) is based on the entire

---

[3] $I(\zeta) = \begin{cases} 0 & \text{if } \zeta \text{ is not satisfied} \\ 1 & \text{otherwise} \end{cases}$

[4] $T = \dfrac{pkts\ in\ \tau \cdot pkt\ size}{\tau}$ *where* τ *is time – window* (bits per second), and hence $\dot{R} \cdot T$ is the effective transmission rate which is actually transmitted for the underlying channel.

trace, it cannot be utilized in real-world applications. We resolve this issue by observing that the probability distribution of the *channel prediction error* process, e, is very close to a Gaussian distribution, $N(0,\sigma_e^2)$, as shown in Figure 14. Furthermore, we take into account the normal distribution, which is well known to have the highest entropy, and hence provides the most conservative estimate. Based on the Gaussian assumption for the prediction error, this leads to an expression that replaces the indicator function with the normal distribution function. The predicted channel capacity can then be optimally tuned by finding a rate $\overset{*}{R}_n$ to maximize $PSNR_n$ as

$$\overset{*}{R}_n = \underset{R_n(0\le R_n\le 1)}{\arg\max} \; Q(R_n\cdot T)\cdot \Pr\{C_n \ge R_n\} + Q'\left(\left|R_n - C_n\right|\cdot T\right)\cdot \Pr\{C_n < R_n\}$$

$$= \underset{R_n(0\le R_n\le 1)}{\arg\max} \; Q(R_n\cdot T)\cdot \frac{\int_{R_n}^{1} \frac{1}{\sqrt{2\pi}\sigma_e}\exp\left(\frac{-(C_n-\hat{C}_n)^2}{2\sigma_e^2}\right)dC_n}{\int_{0}^{1} \frac{1}{\sqrt{2\pi}\sigma_e}\exp\left(\frac{-(C_n-\hat{C}_n)^2}{2\sigma_e^2}\right)dC_n} \qquad (22)$$

$$+ \; Q'\left(\left|\frac{R_n-\hat{C}_n}{R_n}\right|\right)\cdot \frac{\int_{0}^{R_n} \frac{1}{\sqrt{2\pi}\sigma_e}\exp\left(\frac{-(C_n-\hat{C}_n)^2}{2\sigma_e^2}\right)dC_n}{\int_{0}^{1} \frac{1}{\sqrt{2\pi}\sigma_e}\exp\left(\frac{-(C_n-\hat{C}_n)^2}{2\sigma_e^2}\right)dC_n}$$

where $Q(\cdot)$ is the RD (quality) function of the video sequence (as a function of total number of bits used to code the sequence); $Q'(\cdot)$ is the quality function of the video sequence for rates above capacity; and $\int(\cdot)$ represents the probabilities of rate below capacity (the first term in equation (22)) and of rate exceeding capacity (the second term in equation (22)) based on the distribution of *channel prediction error* process. As we highlighted in the previous subsection, we assume that $Q'(\cdot)$ equal to 0. Thus the predicted channel capacity can be fully utilized and the video quality can be optimized when the product of the RD function and the probability distribution of the *channel prediction error* process is maximized. In the subsequent section, we use the above objective function to compute the optimal video rate.

4.5. Performance Evaluation

We now compare the performance of the proposed rate prediction architecture, $ORPA_{CLDS}$, with $ORPA_{CON}$. For notational convenience, henceforth we refer to the architectures of equation (16) and equation (17) as $ORPA_{CLDS_1}$ and $ORPA_{CLDS_2}$, respectively. For $ORPA_{CON}$, we first use checksums for a time-window to find the PER, which is in turn used to estimate the channel capacity for the time-

window. As explained earlier, this channel capacity estimate is then used as the predicted capacity for the next time-window. For $ORPA_{CON}$, the optimal rate adaptation scheme is also employed in the same way as the $ORPA_{CLDS}$ schemes.

To compare the performance of each architecture, experiments in this study were conducted as follows:

1. Estimate the channel capacities of k time windows in a trace by using BER estimate of each packet and equation (16), (17), (18), and (20).
2. Find the optimal rates of k time windows by using equation (22).
3. Calculate the average PSNR values over all time windows of a trace (when the chosen optimal rate is greater than the actual capacity for a given time window, we set the PSNR of the time window to zero) as follows:

$$\text{Average PSNR} = \frac{1}{k}\sum_{n=1}^{k} Q(\overset{*}{R}_n \cdot T) \cdot I(C_n - \overset{*}{R}_n > 0).$$

4. Repeat 1-3 for different error traces.

As for channel capacity prediction, Figure 15 clearly shows that $ORPA_{CLDS_1}$ and $ORPA_{CLDS_2}$ perform better than $ORPA_{CON}$. TABLE IV also compares the performance of channel capacity prediction in terms of MSE for all three architectures. It can be observed that at any physical data rate $ORPA_{CLDS_1}$ provides consistently better

performance than $ORPA_{CLDS_2}$ and $ORPA_{CON}$. Note that exploiting of PER results in less accurate prediction than the prediction based on a BER estimate, and because of the convex property of the entropy function, $ORPA_{CLDS_2}$ provides more conservative and hence less accurate measure (in MSE sense) than $ORPA_{CLDS_1}$.

From TABLE V, we can observe that accurate prediction alone does not necessarily imply better overall rate adaptation. Specifically, if capacity prediction is employed (directly) without optimal rate selection, over-predicted channel capacities cause significant packet drops, thereby leading to considerable PNSR degradation. Therefore, it is imperative that any capacity estimation and prediction framework is complemented with rate tuning.

The excellent performance of the proposed optimal rate tuning scheme can be clearly seen in TABLE VI and Figure 16. By comparing TABLE VI (optimum rate selection) and TABLE V (no optimum rate selection), one can observe that the proposed scheme significantly improves the overall performances of $ORPA_{CLDS_1}$ and $ORPA_{CLDS_2}$ as well as $ORPA_{CON}$. For $ORPA_{CON}$, it should be observed that the optimal rate tuning degrades the overall performance at 11 Mbps. This is because $ORPA_{CON}$ predicts the channel capacity rather conservatively at 11 Mbps due to the fact that large number of packet drops are often introduced. Hence, the predictions of $ORPA_{CON}$ are consistently

and considerably lower than the actually available channel capacity. As a result, the performance of $ORPA_{CON}$ at 11 Mbps is slightly degraded.

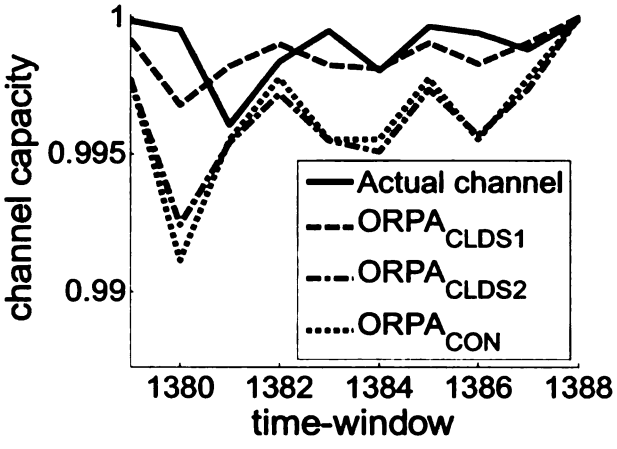
In TABLE VI, we also compare the overall performances of $ORPA_{CLDS_1}$, $ORPA_{CLDS_2}$, and $ORPA_{CON}$ in terms of the average PSNR at different physical data rates and transmit rates. It can be observed that at any physical data rate, the proposed rate adaptation architectures, $ORPA_{CLDS_1}$ and $ORPA_{CLDS_2}$, perform consistently better than $ORPA_{CON}$. However, at 2 and 5.5 Mbps channels, $ORPA_{CLDS_2}$ on average performs better than $ORPA_{CLDS_1}$ whose prediction performance is better than $ORPA_{CLDS_2}$. It is true that $ORPA_{CLDS_1} \geq ORPA_{CLDS_2}$ because of the convexity property of the entropy function, $H_b(\varepsilon)$[41]. Additionally, for 2 and 5.5 Mbps channels, the variance of channel capacity is considerably smaller than the 11 Mbps channel as shown in Figure 15. Thus, $ORPA_{CLDS_1}$ has more probability of selecting the optimal rate exceeding the channel capacity than $ORPA_{CLDS_2}$ even after applying the rate tuning scheme. This results in the performance degradation of $ORPA_{CLDS_1}$ in the video quality although $ORPA_{CLDS_1}$ provides a better performance in prediction of the channel capacity.

To further illustrate the performance of proposed architecture, we have thoroughly tested it using a comprehensive set of RD functions from various CODECs (H.264,

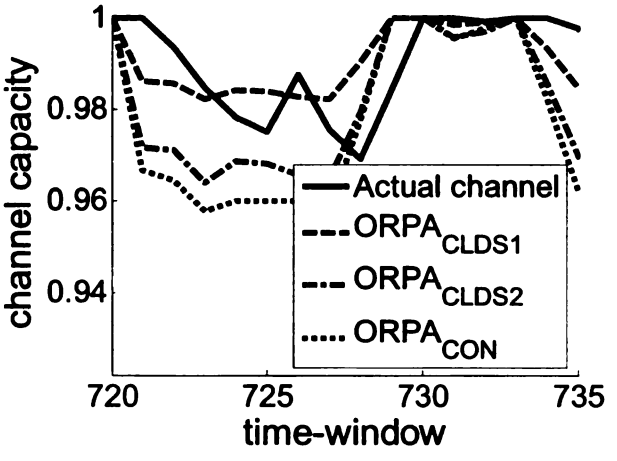
Dvix5.1, WMV9, MC-EZBC, and XivD0.9), and a variety of video sequences (Foreman, Head, Mobile, and Stefan) [38].

TABLE VII and TABLE VIII show that the proposed architecture can achieve up to 5 dB improvement in the average PSNR comparing to $ORPA_{CON}$ and provides consistently good performance with all types of CODECs and video sequences.

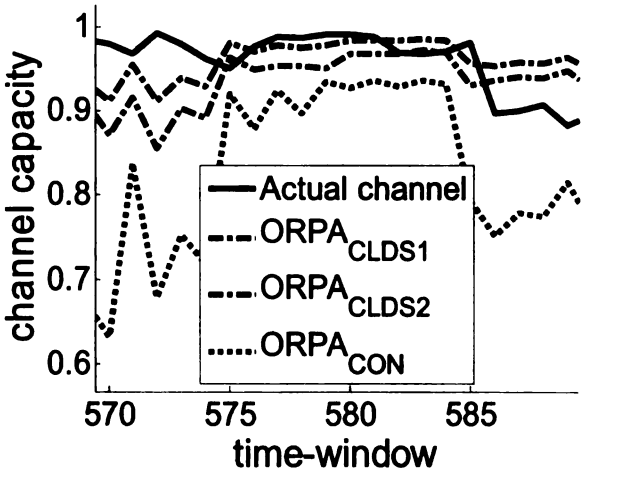
In summary, higher channel capacity and more accurate channel capacity prediction are achieved by $ORPA_{CLDS}$ than $ORPA_{CON}$ at any physical layer data rate; and this leads to better channel utilization and more optimum video rate selection. Moreover, as described in Chapter 3, the collected traces can be considered as the representation of realistic wireless channels, and consequently the trace-driven experiment results represent the outstanding performance of our proposed scheme in the realistic wireless environment.



(a) Phy. rate- 2 Mbps & Xmit rate- 750Kbps

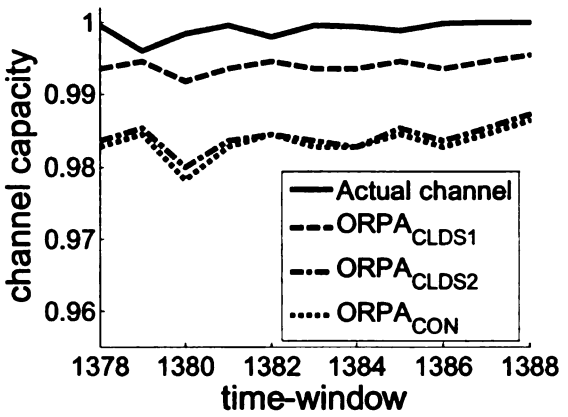


(b) Phy. rate-5.5 Mbps & Xmit rate- 750 Kbps

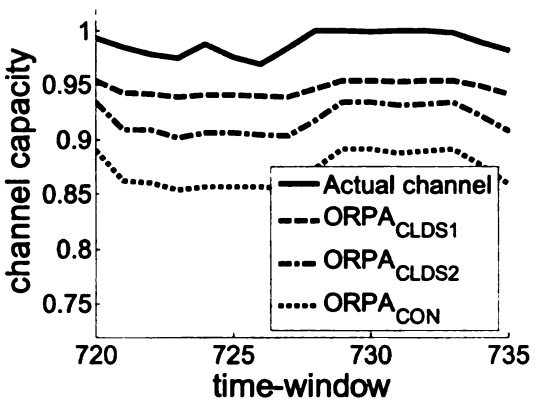


(c) Phy. rate- 11 Mbps Xmit rate- 1 Mbps

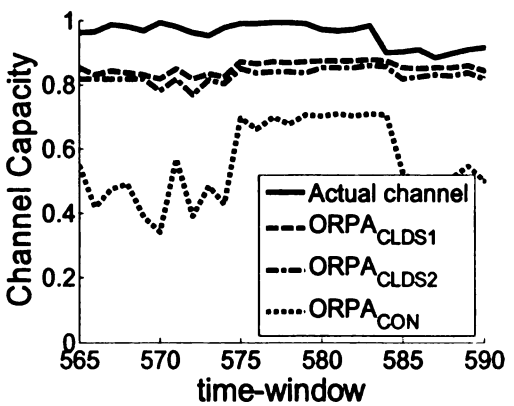
Figure 15 The channel capacity prediction (zoomed in) results.



(a) Phys. rate - 2 Mbps & Xmit rate - 750 Kbps



(b) Phys. rate – 5.5 Mbps & Xmit rate - 750 Kbps



(c) Phys. rate - 11 Mbps & Xmit rate - 1 Mbps

Figure 16 (a), (b), and (c) show the optimally allocated channel capacities (or channel coding rates) based on predictions Figure 15, by $ORPA_{CLDS_1}$, $ORPA_{CLDS2}$ and $ORPA_{CON}$.

TABLE IV Average MSE Of Channel Capacity Prediction.

Phy. data rate (Mbps)	**Packet transmit rate** (Kbps)	$ORPA_{CLDS_1}$ (Info.Bits^2)	$ORPA_{CLDS_2}$ (Info.Bits^2)	$ORPA_{CON}$ (Info.Bits^2)
2	500	0.0004	0.0007	0.0035
	750	0.00002	0.00009	0.0002
	900	0.0002	0.0006	0.0037
	1024	0.0002	0.0009	0.0039
	Overall avg	0.0002	0.0006	0.0028
5.5	500	0.0005	0.0012	0.0053
	750	0.0007	0.0021	0.0121
	900	0.0002	0.0006	0.0019
	1024	0.0017	0.0043	0.0172
	Overall avg	0.0008	0.0020	0.0091
11	500	0.0014	0.0040	0.0563
	750	0.0011	0.0056	0.0168
	900	0.0033	0.0063	0.1267
	1024	0.0056	0.0129	0.2462
	Overall avg	0.0029	0.0072	0.1115

TABLE V Prediction Performance (In Terms Of Overall Video Quality) Before The Optimal Rate Tuning.

Phy	Pkt Xmit Rate (Kbps)	Actual Chan. (PSNR) (dB)	$ORPA_{CLDS_1}$ (dB)	$ORPA_{CLDS_2}$ (dB)	$ORPA_{CON}$ (dB)
2	500	33.98	12.54	30.95	33.30
	750	35.63	21.45	28.04	28.24
	900	38.63	21.45	28.04	33.31
	1024	39.36	27.33	34.80	37.61
	avg	36.90	20.69	30.46	33.12
5.5	500	33.97	17.50	29.36	32.15
	750	35.36	20.25	30.45	31.26
	900	38.53	21.48	33.51	34.64
	1024	39.37	32.15	37.89	37.31
	avg	36.81	22.85	32.80	33.84
11	500	33.98	28.34	32.05	33.25
	750	35.44	32.96	34.61	34.88
	900	37.95	32.96	34.61	34.68
	1024	38.95	38.02	37.39	33.56
	avg	36.58	33.07	34.66	34.09

TABLE VI Prediction Performance (In Terms Of Overall Video Quality) After The Optimal Rate Tuning.

Phy	Pkt Xmit Rate (Kbps)	Actual Chan. (PSNR) (dB)	$ORPA_{CLDS_1}$ (dB)	$ORPA_{CLDS_2}$ (dB)	$ORPA_{CON}$ (dB)
2	500	33.98	33.55	33.70	33.69
	750	35.63	34.83	35.34	35.30
	900	38.63	37.78	37.56	36.84
	1024	39.36	38.60	38.94	38.34
	avg	36.90	36.19	36.39	36.04
5.5	500	33.97	32.32	33.82	33.74
	750	35.36	34.81	34.81	34.46
	900	38.53	37.78	37.56	37.06
	1024	39.37	38.83	38.49	37.45
	avg	36.81	35.94	36.17	35.68
11	500	33.98	33.63	33.71	32.61
	750	35.44	34.79	34.77	34.55
	900	37.95	35.35	35.36	32.68
	1024	38.95	37.49	36.71	32.80
	avg	36.58	35.32	35.14	33.16

TABLE VII Prediction Performance With Foreman Sequence And Various CODECs (pkt rate = 1 Mbps & phy. rate = 11 Mbps).

Codec	Actual channel (PSNR)	$ORPA_{CLDS_1}$ (dB)	$ORPA_{CLDS_2}$ (dB)	$ORPA_{CON}$ (dB)
H.264	38.95	37.49	36.71	33.56
Dvix5.1	36.93	36.54	36.36	31.55
WMV9	37.12	36.69	36.50	32.37
MC	38.13	37.11	36.87	34.08
XivD0.9	36.52	36.05	35.84	33.22

TABLE VIII Prediction Performance With H.264 And Various Video Sequences (pkt rate = 1 Mbps & phy. rate = 11 Mbps).

Sequence	Actual channel (PSNR)	$ORPA_{CLDS_1}$ (dB)	$ORPA_{CLDS_2}$ (dB)	$ORPA_{CON}$ (dB)
Foreman	38.95	37.49	36.71	33.56
Head	39.23	38.79	38.60	35.66
Mobile	31.92	31.28	30.98	27.30
stefan	31.51	31.18	31.06	27.46

CHAPTER 5
RATE ADAPTTION FOR WIRELESS SCALABLE VIDEO

On the basis of the rate adaptation architecture described in the previous Chapter, in Chapter 5 we outline the following research topics: i) a Rate Distortion (RD) model for above-capacity video and ii) Unequal Error Protection (UEP). The proposed topics are selected to bring Optimal Rate Prediction Architecture under CLDS ($ORPA_{CLDS}$) to a certain level of completion and to support utilization of $ORPA_{CLDS}$ in different types of operational networks.

5.1. An RD Model for Above-Capacity Video

In Chapter 4, for practical video streaming over wireless LANs $ORPA_{CLDS}$ is developed. We also show that an accurate source and channel coding rate prediction can be achieved by utilizing the Rate Distortion (RD) of video and the distribution of *channel prediction error* process. However, it is important to note that in Chapter 4 we made the

following assumptions in $ORPA_{CLDS}$: (i) the channel code achieves the capacity (i.e., an ideal channel coder), and (ii) a block of packets cannot be recovered at the client if it is coded with an overestimated rate, i.e., a channel coding rate that exceeds channel capacity. Thus, to optimally realize $ORPA_{CLDS}$ in real world scenarios, we must incorporate i) an operational rate for a specific channel code which is not an ideal code and ii) an RD function for above-capacity video.

5.1.1. Operational Rate

In Chapter 4 we assume that the channel code achieves the capacity (i.e., an ideal channel coder) and a block of packets for a time window cannot be recovered at the client if it is coded with an overestimated rate; here an overestimated rate is a channel coding rate that exceeds channel capacity. However, the optimal rate, which is calculated based on (22), should be adjusted in conjunction with a specific channel code, which is not an ideal code. Therefore, we formulate the *operational rate*[5] as:

$$R^{op} = 1 - \alpha \cdot H(\varepsilon), \quad 1 \le \alpha \le \frac{1}{H(\varepsilon)}, \tag{23}$$

where ε is the actual channel BER. Note that the channel capacity for the considered

[5] The *operational rate* in this study is equivalent to the rate which embodies inferior performance of a practical code to an ideal code.

BSC channel can be estimated as $C = 1 - H_b(\varepsilon)$ [53]. For reliable communication (i.e. distortion free communication,) the *operational rate* has to satisfy $R^{op} \leq C$. While it is theoretically possible to satisfy this constraint, in practice the performance of a code is inferior to the theoretical predictions because of finite length and other design limitations. We capture this performance drop by introducing the parameter α. Thus, we use a stricter constraint $R^{op} + (1-\alpha)H(\varepsilon) \leq C$, where a hypothetically optimal code can be represented by $\alpha = 1$.

As explained above, the value of α plays a critical role in the rate selection. Therefore, we first deduce a suitable value for this parameter. Analytical deduction of α, requires us to consider finite length analysis of LDPC codes. This is a challenging and reasonably open area of research. In this study, we circumvent this issue, and provide a practically workable solution, by empirically evaluating the value of α. Hence, for this purpose, we conducted a comprehensive set of experiments with LDPC channel code [52] and test scalable video sequences. For example, we observed that for $\alpha = 1.8$, 99% of the video packets are decoded successfully. Thus, moving forward, we use this empirically deduced value in our simulation and performance studies. Further, for notational convenience, we refer to the R^{op} as the *operational channel capacity*, C^{op}, which is the maximum achievable rate for the given channel code and for the underlying

channel. Note that to complete equation (22) we develop an empirical model for the quality distortion function, $Q'(\cdot)$ of video sequences for rates exceeding capacity.

5.1.2. The Video Rate-Distortion Model

It is well known that a channel coding rate that exceeds the capacity leads to unreliable communication and increased distortion. The precise increase in distortion depends on a number of factors, such as: (i) the specific video content, (ii) the error concealment/resilience and robustness of a particular source codec, and most importantly (iii) the difference between the overestimated rate and the actual channel capacity. Commercially available video systems are complex and consist of a number of sub-parts. Hence, it is difficult (if not impossible) to deduce a closed form analytical expression that can precisely capture the impact of the overestimated rate on the video quality. Consequently, we develop an empirical model, which can in turn be used in equation (22), to express the video quality for *normalized rates*[6]. Note that in this study a video test sequence (foreman) was encoded by using Scalable Video Coding (SVC) [54] to easily adapt bitrates rather than re-encoding for different rates. When encoding, the following

[6] Rate difference between the over-estimated rate and the operational channel capacity, which normalized by the over-estimated rate, $\frac{R - C^{op}}{R}$.

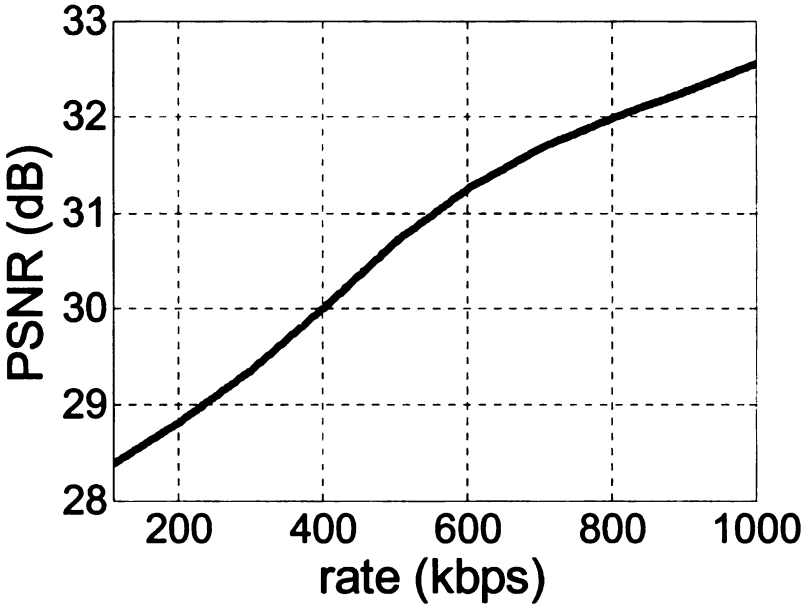
encoding configurations were used: two-layered spatial scalability and three-layered Fine Granularity Scalability (FGS); 30 frame per-second (fps) only with I and P frames; and Group of Pictures (GOP) of 64. Note that the above encoding configurations were used to reduce the computational complexity in wireless mobile terminals, on which our study focuses. In addition, the LDPC code [52] was used as a channel coder.

To deduce the empirical model, we conduct the following measurement procedure:

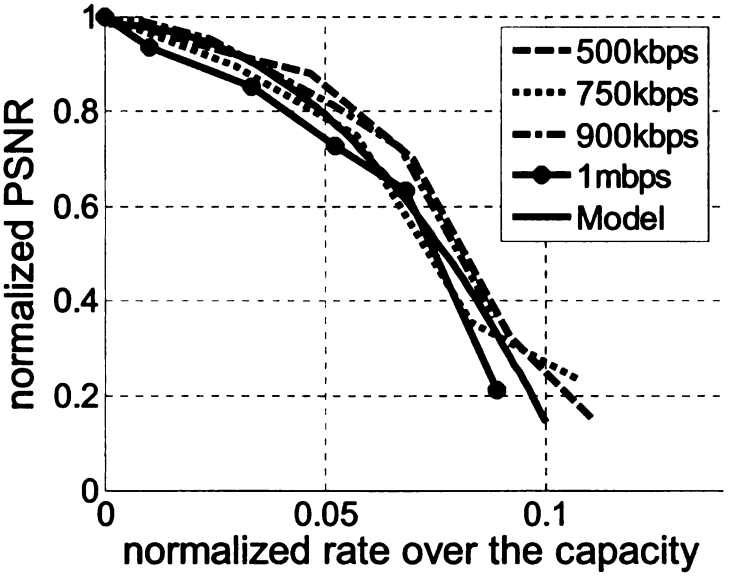
1. Compute the operational channel capacity of the underlying channel (i.e., the average capacity of 11 Mbps traces).
2. Extract the test SVC bitstream at the rate of the operational channel capacity and compute the PSNR of the stream.
3. Encode the extracted bitstream with the LDPC encoder and transmit it to the underlying channel.
4. Decode the bitstream packets using the LDPC decoder and the SVC decoder (with error concealment[7]).
5. Compute the normalized PSNR[8] from the successfully decoded packets.
6. Repeat 3-5 with the increased rate and for four different transmission rates.

[7] A lost frame was replaced with the previous frame.

[8] $Q'\left(\frac{R-C^{op}}{R}\right)/Q'(0)$



(a)



(b)

Figure 17 The RD (quality) function ($Q(\cdot)$) of SVC test video sequence for rates below capacity (a) and the empirical model ($Q'(\cdot)$) for video rate distortion for rates exceeding the channel capacity (b).

From a comprehensive set of measurements, it can be observed that a normalized PSNR decreases as a function of normalized rates [Figure 17]. Consequently, we derive the empirical model to embody the distortion of video quality for rates over the channel capacity as

$$f(x) = ax^b + c, \quad 0 \le x \le 0.12 \tag{24}$$

where $a = -1.18 \times 10^2; b = 2.148;$ *and* $c = 0.9898$. Note that a, b and c were found by a curve fitting and were specified for the foreman sequence which was encoded with SVC [54].

We leverage this empirical video quality distortion model, $Q'(\cdot)$ in equation (22), to optimally select the source and channel rate, i.e., the channel coding rate [53]. The performance of $ORPA_{CLDS}$, which fully employs the model, is described in the following section.

5.1.3. Performance Evaluation

We now evaluate the performance of the proposed video rate-distortion model by simulating $ORPA_{CLDS}$ which employs the proposed model. The simulation procedure with the video quality functions of the pre-encoded bitstream [Figure 17] is as follows:

1. Predict the operational optimal rates, R_n^{*op} and R_{n+1}^{*op}, for randomly chosen two consecutive time windows of a randomly chosen trace (by using equation (22)).
2. Extract a bitstream based on the rates from 1.
3. Encode the extracted bitstream packets with the LDPC encoder and transmit the FEC-encoded packets to the underlying channel.
4. Decode the transmitted packets with the LDPC and the SVC decoder (with error concealment).
5. Repeat 1-4 for different traces with transmission rates.

The simulation results are show in TABLE IX. In TABLE IX, the first and the second columns represent underlying channels (or traces collected at the given physical data rates and transmit rates), the third column represents time window indices, which were randomly chosen from a randomly chosen trace, and the predicted (operational optimal) rates by $ORPA_{CLDS}$; the forth column represents bitrates and PSNRs of the video source, which was extracted based on the predicted rates; and PSNRs of video source which was preserved at a client are shown in the fifth column.

We can observe from TABLE IX, by utilizing the proposed model in conjunction with the CLDS protocol, that there is a small amount of PSNR reduction relative to an error-free decoded bitstream. This distortion is due to burst errors in wireless channel

which cause severe corruption on a few packets. However, the distortion is very small, and the bitstream at the clients preserves most of the original video quality (PSNR) for randomly selected rate adaptation intervals (or time windows) and for different transmission rates [TABLE IX]. This implies an accurate rate prediction performance by $ORPA_{CLDS}$ and a viable empirical model for video rate-distortion. Note that since the test bitstream is 10 second long (300 frames), two consecutive time windows were used for simulations. Thus, the first 150 frames and the rest were extracted according to two different rates. On the contrary, the video quality was computed from the entire bitstream.

Based on the accurate performance of the proposed video rate-distortion model, we extend our analysis on the performance of $ORPA_{CLDS}$ to the entire time windows of all traces by using the following equation:

$$avg\ PSNR = \frac{1}{k}\sum_{n=1}^{k}\left[\begin{array}{l} Q(R_n^{*op}\cdot T)\cdot I(R_n^{*op} \le C_n^{op}) \\ +Q'\left(\left|\dfrac{R_n^{*op} - C_n^{op}}{R_n^{*op}}\right|\right)\cdot I(R_n^{*op} > C_n^{op}) \end{array}\right] \tag{25}$$

where $I(\cdot)$ is an Indication function[9]; R_n^{*op} is the rate computed by using equation (22); C_n^{op} is the operational channel capacity; and k is the number of time windows in a trace.

[9] $I(\zeta) = \begin{cases} 0 & \text{if } \zeta \text{ is not satisfied} \\ 1 & \text{otherwise} \end{cases}$

Additionally, we compared the performance of $ORPA_{CLDS}$ with $ORPA_{CON}$ to highlight the efficiency of $CLDS$ protocols in practical rate adaptation applications. The conventional protocols can only observe the number of packet drops. Thus, for $ORPA_{CON}$ we estimated the channel capacity for a time window using PER as

$$\tilde{C}_n^{CON} = 1 - \frac{1}{m}\sum_{i=1}^{m} Z_i = 1 - PER, \tag{26}$$

and used the same simulation procedure as $ORPA_{CLDS}$.

As shown in TABLE X, $ORPA_{CLDS}$ provides an average PSNR performance that is very close to the maximum PSNR for any given channel. Note that the first column represents the best video quality to be possibly achieved for a transmission rate and for the given physical data rate. These results remark the performance of the proposed video rate-distortion model, with which $ORPA_{CLDS}$ is able to provide an outstanding performance. It should be also noticed that the average PSNRs of both $ORPA_{CLDS}$ and $ORPA_{CLDS}$ for the error trace (or channel), at which is collected the transmit rate at 750 transmit rate and the physical rate at 11 Mbps, have significant gains relative to others. This is due to the fact that almost no rates, which are predicted from the error trace, exceed the maximum achievable rate, while keeping close to the maximum rates.

Moreover, it can also be observed that $ORPA_{CLDS}$ outperforms $ORPA_{CON}$. This

performance difference is due to how the channel is estimated, i.e., BER estimate using side-information provides more accurate channel state estimate than PER.

TABLE IX Performance of $ORPA_{CLDS}$ with an ideal channel code and with a LDPC code (In Terms Of Overall Video Quality).

Phy (Mbps)	Xmit Rate (Kbps)	Time Window Index (Optimal Rates) $(R_n^{*op}, R_{n+1}^{*op})$	Extracted Bitstream PSNR(dB) Bitrate(Kbps)	Decoded Bitstream PSNR(dB)
11	500	437, 438 (0.774,0.744)	29.87 (379.7)	29.20
		1703, 1704 (0.877,0.886)	30.28 (440.3)	29.56
	750	227, 228 (0.904,0.901)	31.57 (677)	30.67
		1837, 1838 (0.890,0.926)	31.59 (681)	31.21
	900	384, 385 (0.638,0.558)	30.96 (538.9)	30.96
		1014, 1015 (0.849,0.779)	31.80 (732)	31.70
	1024	422, 423 (0.843, 0.818)	32.10 (849.1)	32.05
		739, 740 (0.841,0.826)	32.12 (853.5)	32.02

TABLE X Performances of $ORPA_{CLDS}$ and $ORPA_{CON}$ (In Terms Of Overall Video Quality).

Phy (Mbps)	Xmit Rate (Kbps)	Actual Channel (dB)	$ORPA_{CLDS}$ (dB)	$ORPA_{CON}$ (dB)
11	500	29.0564	28.9583	28.7263
		29.0133	28.8043	24.3700
		29.0509	28.8675	24.3804
		29.0422	28.6611	27.8340
		29.0317	28.8711	23.3747
	avg	**29.0389**	**28.8325**	**25.7371**
	750	30.9815	30.7850	30.3802
		30.9465	30.6392	30.0243
	avg	**30.9640**	**30.7121**	**30.2022**
	900	31.8695	31.4052	27.0839
		31.9313	31.6742	27.0339
		31.8198	31.3434	18.5583
	avg	**31.8735**	**31.4743**	**24.2254**
	1024	32.1503	31.5112	0.1117
		32.4389	31.8527	11.5011
		32.4996	32.0516	24.3254
		32.5214	32.0582	30.2564
	avg	**32.4026**	**31.8684**	**16.5486**

5.2. Unequal Error Protection

For multimedia streaming applications, it is essential to accurately estimate/predict the channel capacity to provide QoS as emphasized in previous chapters. However, it is also important to minimize distortion of video quality when video packets are corrupted due to burst errors by severe interference which often occurs in wireless channels. A workaround to this problem is to employ an Unequal Error Protection (UEP) scheme. Under UEP, parts (e.g., Intra-coded frames) of the video bit-stream, which significantly affects video quality, are provided with more protection (i.e. a lower channel coding rate) and vice versa.

5.2.1. UEP Utilizing an LDPC Code

It is well known that for an LDPC code the size of the packet (or message bits) plays an important role in the channel coding performance, i.e., a lager packet size has a better chance of successful decoding than a smaller packet with a given coding rate. As shown in Figure 18 we can observe that the probability of successful decoding is a function of both the size of packet and the redundancy parameter α which is used in (23) in section 5.1.1.

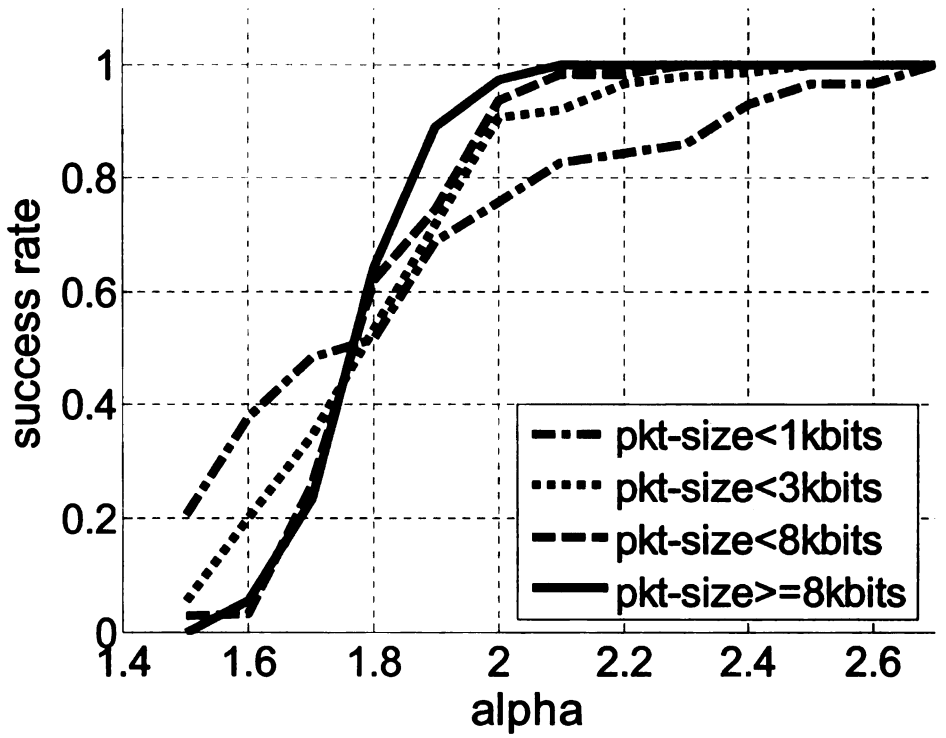


Figure 18 The probability of successful decoding, which is generated with LDPC codes [52] and SVC bit-stream [54], depends on both the size of packet and the parameter α.

Consequently, when a packet containing I frames is encoded with α which leads to the probability of success decoding close to 1 (i.e., $\alpha \simeq 2.1$ for a packet size of 8 Kbits or more), it is highly likely that the packet is successfully decoded. Note that in the previous subsection we apply $\alpha = 1.8$ for every video (I- and P- frame) packets where almost every packet can be successfully decoded. However, channel impairments (e.g., burst errors) can be introduced in any wireless channel, and hence, there is always a chance of a sever packet corruption (or unsuccessful decoding). This results in enormous

distortion of video quality in case of I-frame packet loss (rather than P-frame packet). To that end, we differentiate α for I-frame packet from that for P-frame packet.

After α for each I-frame packet is chosen, α for each P-frame packets can be simply calculated as

$$\alpha_P = \frac{\alpha \sum_{i=1}^{n} L_i - \sum_{i=1}^{k} \alpha_i^I \cdot L_i^I}{\sum_{i=k+1}^{N} L_i^P} \tag{27}$$

where $\alpha = 1.8$; n represents the total number of video packets; k represents the number of I-frame packets; α_I represents the redundancy parameter for I-frame packets; α_P represents the redundancy parameter for P-frame packets; L_i represents the length of ith packet; L_i^I represent the length of ith packet containing an I-frame; and L_i^P represent the length of ith packet containing a P-frames. Note that the total number of redundant bits for a given time-window can be expressed as

$$\alpha \cdot H(\varepsilon) \cdot \sum_{i=1}^{n} L_i = H(\varepsilon) \cdot \sum_{i=1}^{k} \alpha_i^I \cdot L_i^I + \alpha_P \cdot H(\varepsilon) \cdot \sum_{i=k+1}^{n} L_i^P \tag{28}$$

where $\alpha \cdot H(\varepsilon) = 1 - R_*^{op}$, and α_P is used for FEC encoding of P-frame packets.

5.2.2. Performance Evaluation

We now evaluate the performance of the proposed UEP scheme by simulating $ORPA_{CLDS}$ which employs the proposed model. The simulation procedure with the video bit-stream [Figure 17 (a)] is as follows:

1. Predict the operationally optimal rate, R_n^{*op} for a given time window.
2. Extract a SVC bitstream based on the rates from 1.
3. Choose α_I and encode each I-frame packets with the LDPC encoder.
4. Calculate α_P and proceed FEC encoding for P-frame packets by using equation (27).
5. Transmit the FEC-encoded packets over the underlying channel and decode the transmitted packets with the LDPC and the SVC decoder (with error concealment)
6. Compute the *normalized* PSNR from the SVC decoded packets
7. Repeat 1-6 with an increased channel BER.

The excellent performance of the proposed UEP scheme can be clearly seen in Figure 20. We can observe from Figure 20, by utilizing the proposed UEP scheme in conjunction with the CLDS protocol, that there is a small amount of PSNR reduction

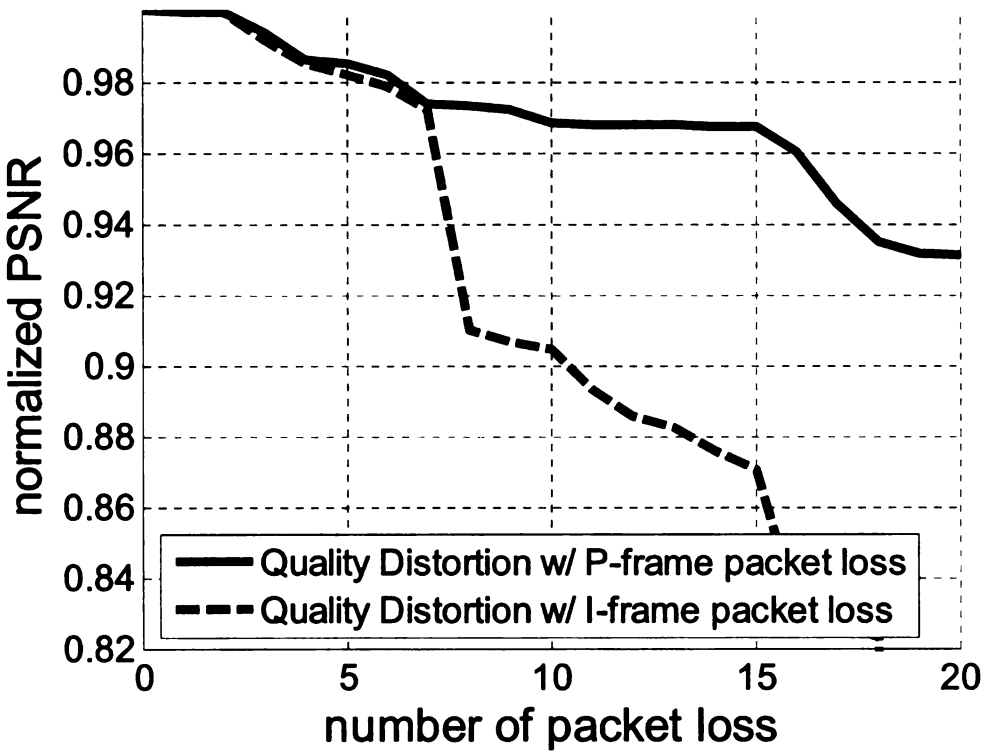


Figure 19 The performance comparison in video qualities with the proposed UEP scheme and without UEP (when only I-frame packets are lost, i.e., the worst case).

relative to an UEP-free decoded bitstream (worst case occurs when I-frame packets are lost) as the number of frame loss increases. This PSNR reduction difference is due to the number of successfully decoded I-frame packets. In other words, the proposed UEP scheme protects I-frame packets more securely so as to minimize the amount of video quality distortion.

CHAPTER 6
SYNDROME PARTIAL DECODING

In general, a client within an ad-hoc network may receive video content after being transmitted over multiple wireless hops. Due to the cascading effect of noise and interference, the overall channel capacity of a multi-hop wireless path drops progressively over each hop, and hence, without optimal rate adaptation, the video quality is expected to degrade significantly at any client located at a far-edge of an ad-hoc network. To overcome this limitation, Decoding and Forwarding (DF), which fully decodes codewords at each intermediate node, can be employed to provide the best video quality. However, complexity and memory usage for DF are significantly high. Consequently, we propose Syndrome-based Partial Decoding (SPD). In the SPD framework an intermediate node partially decodes a codeword and relays the packet along with its syndromes if the packet is corrupted. We exhibit the efficacy of the proposed scheme by simulations using actual IEEE 802.11b wireless traces. The trace-driven simulations show that the proposed SPD framework, which reduces the overall

processing requirements of intermediate nodes, provides reasonably high goodput[10], when compared to simple forwarding, and less complexity and memory requirements, when compared to DF.

6.1. Limitations

Some studies (e.g., [59]-[60]) have focused on wireless multimedia communication over an ad-hoc network which consist of multiple wireless hops from a server to a client. It is well known that each wireless channel between intermediate nodes suffers from impairment due to interference, fading, and multi-path effects. Hence, when a multi-hop wireless network is employed, it can be easily observed that channel capacity decreases over each hop, and hence, the End-to-End (E2E) channel capacity can become very low. This leads to significant degradation of goodput and video quality for rate adaptation applications.

An ideal workaround to this problem is to employ Decoding and Forwarding (DF). Under DF, channel coding is employed over every hop of the end-to-end path. Hence, an intermediate node decodes the transmitted packets (or codewords) to suppress noise on

[10] In this study the goodput is defined as the application level throughput, i.e. the number of information bits per total number of bits forwarded by the network from a certain source address to a certain destination.

the channel between two intermediate nodes, and re-encodes the packets, potentially with a different codebook, for transmission towards the destination [61]. Thus, DF can achieve optimal performance with regard to network capacity. However, complexity and memory usage for DF are significantly high. Moreover, intermediate nodes, which may be participating by only forwarding the content toward a receiver further-down a multi-hop chain, do not have much incentive to perform full decoding-encoding of the channel-coded wireless video content. For such nodes, we need to minimize their burden in terms of the operation they need to perform toward the delivery of the video content to the final receiver.

The above discussion motivates the usage of *partial processing framework* for rate-adaptive wireless video, which reduces the overall processing requirements of intermediate nodes. In particular, in this paper, we propose Syndrome-based Partial Decoding (SPD) architecture. The proposed architecture employs CLDS protocols, which among other things, accurately estimate the channel capacity using simple Binary Symmetric Channel (BSC) [56] model. The two main contributions of this paper are:

1) A syndrome-based partial decoding scheme. Under this scheme, each intermediate node only computes the syndromes of the received packet (i.e., partial decoding) and relays the syndromes along with the packet (if the packet is corrupted) to the next hop.

The client then fully decodes the packet by using the syndromes which are appended to the packet.

2) An optimal packet size selection scheme. As explained later, SPD utilizes channel Packet Error Rate (PER) as an error parameter for its operation; and hence SPD goodput heavily depends on the size of packets. Thus, we derive the optimal packet size selection scheme to maximize the goodput in the proposed architecture.

The proposed SPD architecture is tested using a comprehensive set of wireless residual error traces collected at 2, 5.5 and 11 Mbps physical data rates of an operational 802.11b network. We compare the performance of SPD with DF, E2E decoding (END) and automatic repeat request (ARQ). We show that SPD outperforms END decoding and ARQ and provides relatively good goodput compared to that by DF.

6.2. Syndrome-based Partial Decoding

In this chapter we develop the rate adaptation architecture over an ad-hoc network using the two main contributions of this paper: SPD and optimal packet size selection. SPD is achieved by i) only computing a syndrome for a packet at each intermediate node; ii) forwarding codeword for the syndrome of the packet (only when the packet is

corrupted at a hop) to the next hop; and iii) fully decoding the packet by using its syndromes at a client. SPD embodies BER as well as PER; i.e., the goodput is a function of both BER and PER. Thus, the optimal goodput is achieved by finding an optimal packet size that provides the best goodput in the proposed architecture.

6.2.1. Architecture for Rate-Adaptation

The architecture of a multimedia streaming application depends heavily on a network over which it operates. Therefore, in this section, we define the proposed architecture for rate adaptation. Additionally, we outline the assumptions under which the proposed architecture is to be evaluated.

The proposed architecture consists of a server, intermediate nodes and a client that receives packets over a multi-hop wireless network. In the given architecture, a server, intermediate nodes and a client are designed to support source and channel rate adaptation. Figure 20 illustrates such architecture. The intermediate nodes and client supports CLDS protocols that leverage residue-error-process and side information, which can be relayed to an FEC decoder (for partial decoding at each intermediate nodes or full

decoding at a client)[37], to estimate the current channel capacity[11] for a block of packets (or a rate adaptation period). The current channel capacity, which is estimated by the channel estimator with the entropy of the residue error process, is then transmitted to the server as feedback for rate adaptation. Using the feedback, the rate tuner at the server predicts the optimal source and channel rates for the next block of multimedia packets to be transmitted [56].

For this study, we focus on the performance of SPD, and hence we assume the system employs generic (arguably ideal) channel and source coding schemes. In particular, we consider the following simplifying assumptions: (i) the channel code achieves the capacity (i.e., an ideal channel coder), and a block of packets can be successfully decoded at the client if a source and channel coding rate does not exceed channel capacity, i.e., $R \leq C$ [25]; (ii) A video encoder provides a bit-stream having a bitrate that is exactly the same as the required bitrate, and the bit-stream renders Peak-to-Peak Signal to Noise Ratio (PSNR) value [38] according to the bitrate. With (i) and (ii), we realize the architecture where for video rates that cannot be supported by the underlying channel

---

[11] Here, the channel capacity is estimated as $C_n = 1 - \frac{1}{m}\sum_{i=1}^{m} H_b(\varepsilon_i)$ where ε_i represents the BER estimate for packet i; $\frac{1}{m}\sum_{i=1}^{m} H_b(\varepsilon_i)$ is the instantaneous per-packet error-process entropy in a block of m packets, and n is the time-window index [56].

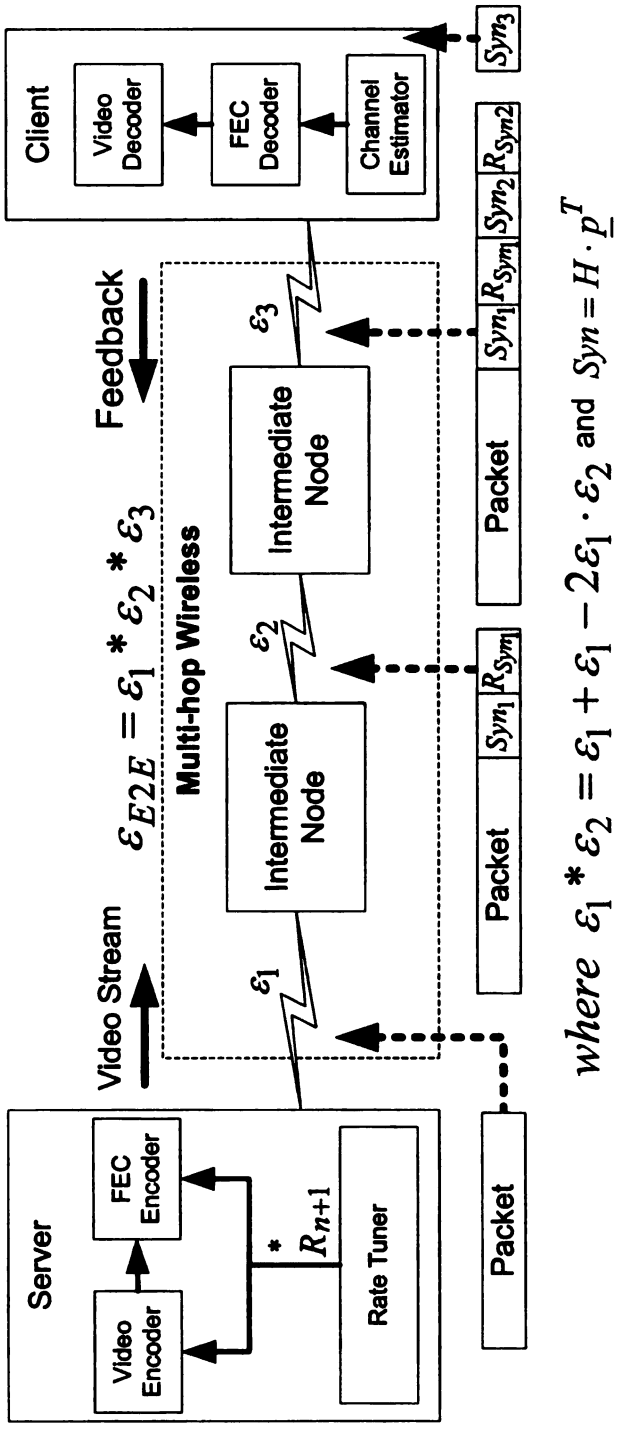


Figure 20 The architecture of the proposed rate adaptation.

capacity, the video quality reduces to zero (i.e., zero PSNR value). Note that without the assumptions (i) and (ii) (i.e., a realistic channel coder or video encoder was used), we should carefully take the performances of the channel coder and video encoder into consideration in finding the rate. Therefore, the above assumptions were made to focus only on the performance of the proposed SPD scheme.

6.2.2. SPD Overview

Each intermediate node computes the syndromes of packet ($Syndrome_{r\times 1} = Check\_matrix_{n\times r} \cdot Packet^{T}_{1\times n}$) from the server or the previous intermediate node and forwards the packet with the packet's syndrome (including the redundancy of the syndrome; i.e., codeword for the syndrome) only if the packet is corrupted. Note that when a packet is not corrupted (the syndromes consist of all 0s) only the packet is relayed to the next hop, Meanwhile, when a packet is corrupted each node adds its syndrome (including syndrome redundancy) to the packet in an embedded manner; and hence, this process is repeated until a client receives the packet. A client then uses the syndromes, which are coded and appended to the packet, to decode the packet. For example, under the scenario shown in Figure 20, when the packet is

corrupted at the first hop, then a client receives the packet and two different syndromes appended to the packet as shown in Figure 20 and computes the syndrome for the last hop. Note that a syndrome is used as the information needed to successfully decode the packet which is corrupted in a hop (or a channel link).

It should be noted that a server uses the maximum channel BER ($\varepsilon_{\max} = \max(\varepsilon_1, \varepsilon_2, \varepsilon_3)$) to select the rates rather than E2E BER ($\varepsilon_{E2E} \geq \varepsilon_{\max}$) [Figure 20] since each syndrome can be used to correct errors occurred at each hop. This leads SPD to suppress noise on each channel between a server and a client and to provide the goodput performance close to DF.

Using Figure 20 as an illustrative example, the packet received at a client is then decoded along with its syndromes as follows:

i) Decode the codewords for the syndromes that is forwarded from the previous nodes

ii) Decode the codeword for the message with each of the decoded codewords for the syndromes

Note that the above procedure specifies the (1 bit error correcting) Hamming code and the similar procedure can be applied to Reed Solomon (RS) code or turbo code, for which the syndrome polynomial can be employed.

6.2.3. Goodput Evaluation

In this section, we develop the expression for the goodput of proposed SPD and other schemes such as DF, End node decoding, and ARQ. As described in the previous subsection, for SPD the total number of bits transmitted to multi-hop wireless network increases as PER and the number of hops increase. Therefore, the goodput for SPD, $Goodput_{SPD}$ can be expressed as

$$Goodput_{SPD} = \frac{k}{n_{total}} = \frac{k}{\left\lceil \frac{(k+r)\cdot h + \sum_{i=1}^{h}\left(PER_i \cdot \left(k_{s_i} + r_{s_i}\right)\cdot h\right)}{h} \right\rceil} \tag{29}$$

$$= \frac{k}{(k+r) + \sum_{i=1}^{h}\left(PER_i \cdot \left(k_{s_i} + r_{s_i}\right)\right)}$$

$$= \frac{k}{\left(k + \frac{k\cdot H(\varepsilon_{\max})}{1-H(\varepsilon_{\max})}\right) + \sum_{i=1}^{h} PER_i \cdot \left(r + \frac{r\cdot H(\varepsilon_{\max})}{1-H(\varepsilon_{\max})}\right)}$$

$$= \frac{k}{k\cdot\left(1 + \frac{H(\varepsilon_{\max})}{1-H(\varepsilon_{\max})}\right) + \sum_{i=1}^{h} PER_i \cdot r \cdot \left(1 + \frac{H(\varepsilon_{\max})}{1-H(\varepsilon_{\max})}\right)}$$

$$= \frac{k}{\left[k + \sum_{i=1}^{h} PER_i \cdot r\right] \cdot \Delta_{\max}} = \frac{k}{\left[k + \sum_{i=1}^{h} PER_i \cdot \frac{k \cdot H(\varepsilon_{\max})}{1 - H(\varepsilon_{\max})}\right] \cdot \Delta_{\max}}$$

$$= \frac{1}{\left[1 + \sum_{i=1}^{h} PER_i \cdot \frac{H(\varepsilon_{\max})}{1 - H(\varepsilon_{\max})}\right] \cdot \Delta_{\max}}$$

where k is the number of information (original source) bits; r is the number of redundant bits for k; $n\,(=k+r)$ is the total number of bits in a packet length; k_s is the number of syndrome (bits) of the packet; r_s is the number of redundant bits for the syndrome; h represents the number of hops from a server to a client; and $\Delta_{\max} = 1 + \frac{H(\varepsilon_{\max})}{1 - H(\varepsilon_{\max})}$. Note that $n_{total} = \frac{(k+r) \cdot h + \sum_{i=1}^{h} \left(PER_i \cdot \left(k_{s_i} + r_{s_i} \right) \cdot h \right)}{h}$ which normalizes the total number of bits by defining per-hop transmission cost; the length of k_s is the same as that of r; and $r_{s_i} = \frac{k_{s_i} \cdot H(\varepsilon_{\max})}{C} = \frac{r \cdot H(\varepsilon_{\max})}{C} = \frac{r \cdot H(\varepsilon_{\max})}{1 - H(\varepsilon_{\max})}$.

DF corrects error bits (full decoding) and re-encoding at every intermediate node; and hence, the goodput for DF, $Goodput_{DF}$, can be expressed as

$$Goodput_{DF} = \frac{k}{n_{total}} = \frac{k}{n} \tag{30}$$

$$= \frac{k}{\left(k + \frac{k \cdot H(\varepsilon_{\max})}{1 - H(\varepsilon_{\max})}\right)} = \frac{k}{k \cdot \left(1 + \frac{H(\varepsilon_{\max})}{1 - H(\varepsilon_{\max})}\right)} = \frac{k}{k \cdot \Delta_{\max}} = \frac{1}{\Delta_{\max}}$$

where $n_{total} = \frac{n \cdot h}{h}$ normalizing the total number of bits by defining per-hop transmission cost.

End node decoding, which we defined in this study, employs CLDS protocols but does not incorporate a partial or full decoding at any intermediate node and uses the E2E channel BER, ε_{E2E}, for finding a source and channel coding rate. Thus the goodput for end node decoding, $Goodput_{END}$, can be expressed as

$$Goodput_{END} = \frac{k}{n_{total}} = \frac{k}{(k+r)}$$

$$\frac{k}{\left(k + \frac{k \cdot H(\varepsilon_{E2E})}{1 - H(\varepsilon_{E2E})}\right)} = \frac{k}{k \cdot \left(1 + \frac{H(\varepsilon_{E2E})}{1 - H(\varepsilon_{E2E})}\right)} \tag{31}$$

$$= \frac{k}{k \cdot \Delta_{E2E}} = \frac{1}{\Delta_{E2E}}$$

where $\varepsilon_{E2E} = \varepsilon_1 * \varepsilon_2 * ... * \varepsilon_n$ and $\Delta_{E2E} = 1 + \frac{H(\varepsilon_{E2E})}{1 - H(\varepsilon_{E2E})}$. Note that BER for two cascaded BSCs is $\varepsilon_1 * \varepsilon_2 = \varepsilon_1 + \varepsilon_2 - 2\varepsilon_1 \cdot \varepsilon_2$.

For ARQ, a packet is re-transmitted until a defined time-out by a server if it is corrupted during its transmission from a server to a client. Therefore, the goodput for ARQ can be expressed as

$$Goodput_{ARQ} = 1 - PER_{E2E} \tag{32}$$

where $PER_{E2E} = PER_1 * PER_2 * ... * PER_n$.

6.2.4. Optimization of SPD

It is well known that PER changes with respect to a size of packet for a given channel BER, and $Goodput_{SPD}$ is a function of BER and PER. Therefore, we capture this aspect to find the optimal packet size (pkt_size), which maximizes $Goodput_{SPD}$ [41] as

$$\begin{aligned} pkt\_size^{*} &= \arg\max_{pkt\_size} Goodput_{SPD} \\ &= \arg\max_{pkt\_size} \frac{1}{\left[1 + \sum_{i=1}^{h} PER_i \cdot \frac{H(\varepsilon_{\max})}{1 - H(\varepsilon_{\max})}\right] \cdot \Delta_{\max}}. \end{aligned} \tag{33}$$

It requires to model the PER (as a function of packet sizes) to fully utilize equation (33). Consequently, we develop an empirical PER model to express the PER for packet sizes. To deduce the empirical model, we conduct the following measurement procedure:

i) Calculate BER of the underlying channel (i.e., the BER of a collected trace).

ii) Calculate PER with respect to a packet size.

iii) Repeat i) and ii) with different traces.

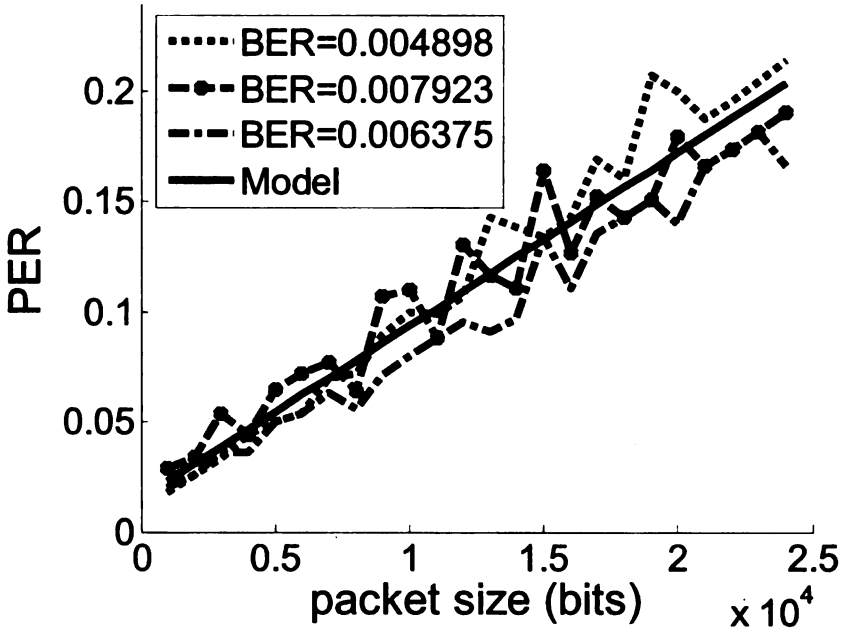


Figure 21 The empirical PER model.

From a comprehensive set of measurements, we observed that PER linearly increases as a packet size increases [Figure 21]. Consequently, we derive the empirical model to embody the PER for packet sizes as

$$PER = ax + b \tag{34}$$

where $a = 1.071 \times 10^{-5}$, $b = 0.01597$, and x is a packet size. Note that a and b were found by a linear fitting.

We leverage this empirical PER model to optimally select the packet size which maximizes $Goodput_{SPD}$. The performance of SPD, which fully employs this model, is described in the following section.

6.3. Performance Evaluation

We now compare the performance of the proposed SPD with DF, End node decoding, and ARQ in terms of goodput and the total number of bits transmitted in an ad-hoc network. To compare the performance of each scheme, experiments in this study are conducted as follows:

1. Calculate the BER and PER of each channel (or hop) for a rate adaptation period. Note that we use 2 Mbps traces for this simulation and treat each trace as the channel between two nodes in the network.
2. Find a source and channel coding rate, for a rate adaptation period. Note that we assume to use an ideal channel code; and hence, we use the capacity as the rate.
3. Calculate the goodput for SPD, DF, End node decoding, and ARQ by using equations (29), (30), (31), and (32).
4. Find the optimal packet size by using equation (33) which incorporates the empirical PER model [see (34)] and the corresponding goodput.

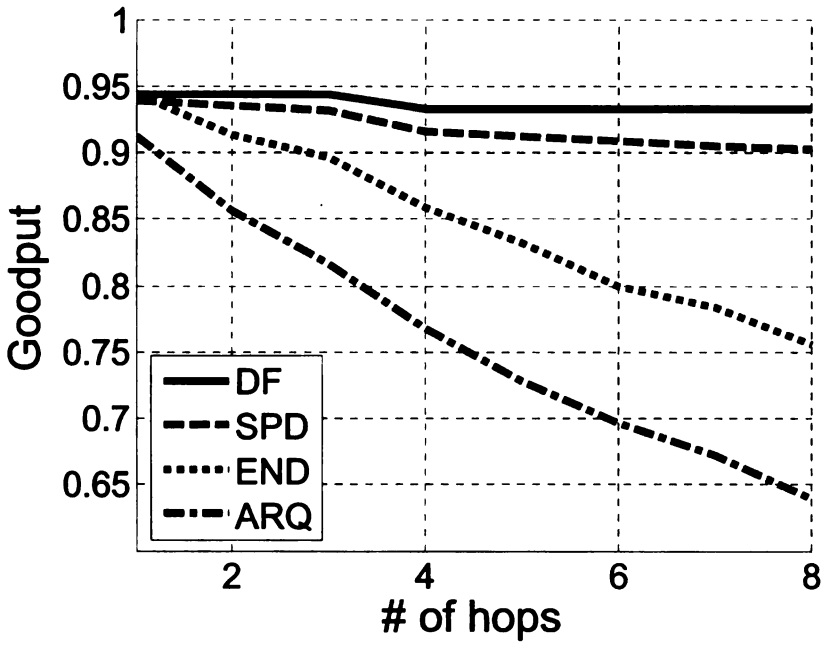


Figure 22 Performance comparison among four different schemes in terms of goodput. For this experiment, a packet size of 8000 bits is used.

5. Calculate the PSNR using the rate-distortion (RD) function of a scalable video coding (SVC) sequence, $Q(\cdot)$ [Figure 17 (a)][54], for the given goodput as $PSNR = Q(Goodput_{SPD} \cdot T)$ where T represents a transmission (Xmit) rate in bits-per-second (bps).

As for goodput, Figure 22 shows that SPD outperforms End node decoding and ARQ and provides the performance, which is a little inferior to DF over hops. However, it should be noticed that although DF provides the best performance in terms of goodput, the complexity and memory usage at each intermediate node are significantly high since

each intermediate node proceeds full decoding and re-encoding of packets. Additionally, the result shown in Figure 22 does not employ the optimal packet size selection scheme.

The excellent performance of the proposed optimal packet size selection scheme for SPD can be clearly seen in TABLE XI. The first column of TABLE XI represents the number of hops; the second, third, and fourth column represent the optimal packet size, goodput, and PSNR achieved by SPD under the actual channel (i.e., the collected traces used for simulations in this study); the fifth column represents the optimal packet size estimated by equations (33) and (34); and the sixth and seventh column respectively represent the goodput and PSNR with the optimally estimated packet size under actual channel. As shown in TABLE XI, the optimal packet size selection scheme estimates very accurate packet size that maximizes both goodput and PSNR. In fact, they are very

TABLE XI The performance of SPD with optimal packet size selection scheme in terms of goodput (Xmit rate=500Kbps)

hop	Actual			Optimization		
	Optimal pkt_size (bits)	Goodput	PSNR (dB)	Optimal pkt_size (bits)	Goodput	PSNR (dB)
2	28555	0.9126	30.38	26700	0.9124	30.38
3	18806	0.8999	30.34	19100	0.8960	30.32
4	14487	0.8760	30.26	14456	0.8759	30.25
5	13754	0.8669	30.23	12649	0.8646	30.21

close to the optimum.

CHAPTER 7
CONCLUSION

In this thesis, we developed an optimal rate prediction architecture using CLDS protocols, $ORPA_{CLDS}$ to support QoS for wireless scalable video. $ORPA_{CLDS}$ combines the following schemes: i) a BER prediction which leads to an accurate channel capacity estimation and prediction; ii) an optimal source and channel coding rate prediction and tuning; iii) Unequal Error Protection (UEP); and iv) Syndrome Partial Decoding (SPD).

As for BER prediction over wireless networks, a multi-tier model (MTM) is developed. The MTM relies on the premise that packet-error prediction should be performed before and in isolation from BER prediction. For predicted error-free packets, BER prediction is not required. The MTM achieved packet-error prediction using a 3rd order Markov chain model. For predicted packet-errors, the MTM uses a second-tier model of Signal to Silence Ratio (SSR) values to predict the next packet's SSR. These predicted SSR values are in turn used in a third tier for BER prediction. We showed that the MTM renders higher prediction accuracy than existing Yule-Walker and finite-state

Markov chain predictors. Consequently, it is revealed that packet level side information can be utilized to accurately estimate/predict channel conditions.

We observe that accurate channel prediction alone does not necessarily imply better overall rate adaptation. Specifically, if capacity prediction is employed (directly) without optimal rate selection, over-predicted channel capacities cause significant packet drops. To that end, on the basis of the BER prediction we develop an optimal rate tuning scheme which leverages both the probability distribution of channel capacity prediction error process and an RD (quality) function of a video sequence. Further, a Rate Distortion (RD) model for above-capacity video and an operational rate for a specific channel code are deduced to employ the optimal rate tuning scheme in a practical wireless environment.

Our experimental results showed that $ORPA_{CLDS}$ provides higher prediction accuracy and better multimedia quality than $ORPA_{CON}$ (optimal rate prediction architecture under conventional protocols) at any physical layer data rate. another interesting observation is that the proposed optimal rate tuning scheme performs considerably well on traces with very low temporal correlation, thus highlighting that the proposed scheme can efficiently cater for bad channel predictions, i.e., the proposed scheme provides good performance even for relatively memoryless and isolated errors.

When burst error occurs over wireless networks, it is highly likely that video quality faces severe degradation. To resolve this problem we develop an UEP scheme which utilizes characteristics of a LDPC code. It is shown that the proposed UEP scheme reduces the degradation of video quality. It should be noticed that the impact of such a scheme is potentially quite significant, not only on the overall video rate selection, but rather on the overall coding strategy of the video encoder (in terms of rate allocation among different frame types).

Finally, we proposed the new *partial processing framework* for multi-hop rate-adaptive wireless video, i.e., SPD. From our experiments, SPD, which reduces the overall processing requirements of intermediate nodes, provides reasonably high goodput, when compared to End node decoding and Automatic Repeat request (ARQ). It is also observed that SPD provides less goodput than Decoding and Forwarding (DF). However, the performance difference between DF and SPD is relatively small if we take complexity and memory requirements into consideration. From this study, we exhibit the applicability of $ORPA_{CLDS}$ in multi-hop wireless networks by introducing SPD.

The current research shows that $ORPA_{CLDS}$ provides the outstanding performance as a rate-adaptive video streaming application over wireless LANs. However, it should be noted that in order to employ $ORPA_{CLDS}$ as a real-time application the followings are

required: a real-time Rate Distortion (RD) model for both above-capacity video and below-capacity video. As described in section 5.1.2, the precise increase in distortion depends on a number of factors, such as the specific video content, the error concealment/resilience and robustness of a particular source codec, and etc. Thus, it is very difficult (if not impossible) to deduce an RD model which accurately estimates a PSNR of any video bit-stream in real-time. Additionally, if SPD incorporates entropy coding (e.g. Hoffman or run-length coding) for syndromes at each intermediate node, the goodput can be further improved. These are the possible future topics to possibly utilize the optimal rate prediction architecture as a real-time rate-adaptive video streaming application over any kind of wireless LANs.

APPENDIX

List of papers published or submitted

Journal Papers

- **Y. Cho**, S. S. Karande, K. Misra, H. Radha, J. Yoo, and J. Hong, "*On Channel Capacity Estimation and Prediction for Rate Adaptive Wireless Video,*" accepted for publications in IEEE Transactions on Multimedia.
- S. Karande, U. Parrika, K. Misra, H. Radha, **Y. Cho**, J. Yoo, and J. Hong, "*Side-Information Aware Non-Homogenous Markov Models For The 802.11b MAC-to-MAC Channel,*" submitted to IEEE Communications (under review), 2008.
- **Y. Cho**, H. Radha, and J. Yoo, "*Channel Capacity Prediction for Rate Adaptive Wireless SVC,*" submission to European Association for Signal Processing (EURASIP) on Wireless Communications and Networking.
- **Y. Cho**, H. Radha, and J. Seo, "*Multi-hop Rate Adaptive Wireless Scalable Video Using Syndrome-based Partial Decoding,*" submission to Electronics and Telecommunications Research Institute Journal.

Conference Paper

- **Y. Cho**, H. Radha, J. Yoo, and J. Hong, "*A Rate-Distortion Empirical Model for Rate Adaptive Wireless Scalable Video,*" Conference on Information Sciences and Systems (CISS), Princeton University, March 2008.
- **Y. Cho**, S. A. Khayam, S. S. Karande, H. Radha, J. Kim, and J. Hong, "*A Multi-Tier Model for BER Prediction over Wireless Residual Channels,*" Conference on Information Sciences and Systems (CISS), John Hopkins University, March 2007.
- S. Karande, S. Khayam, **Y. Cho**, K. Misra, H. Radha, J. Kim, and J. Hong, "*On Channel State Inference and Prediction using Observable Variables in 802.11b Networks,*" IEEE International Conference on Communications (ICC), June 2006.

REFERENCES

[1] S. A. Khayam, S. 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," Signal Processing: Image Communication, vol. 18, Aug 2003.

[2] S. A. Khayam, S. Karande, M. U. Ilyas, and H. Radha, "Header detection to improve multimedia quality over wireless networks," IEEE Trans. On Multimedia, to appear.

[3] S. A. Khayam, S. Karande, M. U. Ilyas, and H. Radha, "Improving wireless multimedia quality using header detection with priors," IEEE ICC, 2005.

[4] S. A. Khayam, M. U. Ilyas, K. Porsch, S. Karande, and H. Radha, "A statistical receiver-based scheme for improved multimedia throughput over wireless networks," IEEE ICC, 2004.

[5] H. L. Van Trees, Detection, Estimation, and Modulation Theory, Part I, Wiley: New York, 2001.

[6] J. G. Kemeny and J. L. Snell, Finite Markov Chains, Springer-Verlag: New York, 1976.

[7] R. Riedi, M. Crouse, V. Ribeiro, and R. Baraniuk, "A multifractal wavelet model with application to network traffic," IEEE Trans. on Info. Theory, (45)3, pp. 992–1018, 1999.

[8] S. A. Khayam and H. Radha, "Linear-complexity models for wireless MAC-to-MAC channels," ACM WINET, (11)5, pp. 543–555, 2005.

[9] S. A. Khayam and H. Radha, "Markov and multifractal wavelet models for wireless MAC-to-MAC channels," Elsevier Performance Evaluation Journal, to appear.

[10]S. A. Khayam and H. Radha, "Constant-complexity models for wireless channels," IEEE Infocom, April 2005.

[11]A. Servetti, J.C. De Martin, "Link-Level Unequal Error Detection for Speech Transmission over 802.11b Networks," Proc. Special Workshop in Maui (SWIM), Maui, Hawaii, January 2004.

[12]E. Masala, M. Bottero, J.C. De Martin, "Link-Level Partial Checksum for Real-Time Video Transmission over 802.11 Wireless Networks", Proceedings of 14th International Packet Video Workshop (PVW), Irvine, CA, December 2004.

[13]E. Masala, M. Bottero, J.C. De Martin, "MAC-Level Partial Checksum for H.264 Video Transmission over 802.11 Ad-hoc Wireless Networks" Proceedings of IEEE 61st Semiannual Vehicular Technology Conference (VTC), May 2005.

[14]E. Masala, A. Servetti, J.C. De Martin, "Standard Compatible Error Correction for Multimedia Transmissions over 802.11 WLAN", Proc. of IEEE International Conference on Multimedia & Expo (ICME), July 2005.

[15]S. Yi, Y. Shan, S. Kalyanaraman and B. Azimi-Sadjadi, "Header Error Protection for Multimedia Data Transmission in Wireless Ad-hoc Networks", ..

[16]R. Riemann and K. Winstein, "Improving 802.11 Range with Forward Error Correction", MIT-CSAIL-TR-2005-011

[17]F. Pauchet, C. Guillemot, M. Kerdranvat, S. Pateux, P. Siohan, "Conditions for SVC bit error resilience testing" Joint Video Team (JVT) 17th Meeting: Nice, FR, 14-21 October, 2005

[18]W. Jiang, "A Bit Error Correction without Redundant Data: a Mac layer technique for 802.11 Networks", WinMee 2006.

[19]S. Karande and H. Radha, "Hybrid Erasure-Error Protocols for Wireless Video," IEEE Transactions on Multimedia, Feb. 2007.

[20]S. Karande, U. Parrikar, K. Misra and H. Radha, "On Modeling of 802.11b Residue Errors," Conference on Information Sciences & Systems (CISS), March 2006.

[21]S. Karande, U. Parrikar, K. Misra and H. Radha, "Utilizing Signal to Silence Ratio indications for improved Video Communication in presence of 802.11b Residue Errors ", IEEE International Conference on Multimedia & Expo (ICME), July 2006.

[22]Utpal Parrikar, "On SNR Aware Analysis and Modeling of 802.11b Link-level Residue Errors", Master's Thesis, Electrical and Computer Engineering Department, Michigan State University.

[23]S. Karande, K. Misra and H. Radha, "CLIX: Network Coding and Cross-layer Information Exchange of Wireless Video", IEEE ICIP 2006.

[24]S. Karande, K. Misra, U. Parrikar and H. Radha, "Utility Of Signal To Silence Ratio's As A Predictive Tool For Enhancing Cross-layer Multimedia Protocols" under review, IEEE Transactions on Multimedia.

[25]S. A. Vanstone and P. C. van Oorschot, "An Introduction to Error Correcting Codes with Applications," Kluwer Academic Publishers.

[26]R. H. Morelos-Zaragoza, "The Art of Error Correcting Coding," John-Wiley & Sons Ltd.

[27]L. Rizzo, "Effective Erasure Codes for Reliable Computer Communication Protocols", Computer Communication Review, April 1997.

[28]W.E. Ryan, "An introduction to LDPC codes," in CRC Handbook for Coding and Signal Processing for Recoding Systems (B. Vasic, ed.), CRC Press, 2004.

[29]X. Hu, E. Eleftheriou, D. M. Arnold, "Regular and Irregular Progressive Edge-Growth Tanner Graphs", IEEE Transactions on Information Theory, vol. 51. no. 1, pp. 386-398, 2003.

[30]T. M. Cover and J. A Thomas, "Elements of Information Theory," Wiley Series in Telecommunications.

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

[32]A. Singh, A. Konrad, A. D. Joseph, "*Performance Evaluation of UDP Lite for Cellular Video*," NOSSDAV, 2001.

[33]S. Karande, K. Misra, and H. Radha, "*Survival Of The Fittest: An Active Queue Management Technique For Noisy Packet Flows*," Advances in Multimedia, vol. 2007, Article ID 64695, 2007.

[34]S. A. Khayam, "Wireless Channel Modeling and Malware Detection using Statistical and Information Theoretic Measures," Ph.D dissertation, Dec 2006.

[35]Y. Cho, S. A. Khayam, S. S. Karande, H. Radha, J. Kim, and J. Hong, "*A Multi-Tier Model for BER Prediction over Wireless Residual Channels*," CISS 2007, March 2007.

[36]D. Aguayo, J. Bicket, S. Biswas, G. Judd, R. Morris, "*Link-level Measurements from an 802.11b Mesh Network*", SIGCOMM 2004, Aug 2004.

[37]S. Lin and D.J. Costello, Jr., "*Error Control Coding: Fundamentals and Applications*", Englewood Cliffs, NJ: Prentice Hall, 1983.

[38]P. Lambert, W. D. Neve, P. E. Neve, and I. Moerman, "*Rate-Distortion Performance of H.264/AVC Compared to State-of-the-Art Video Codecs*," IEEE Transactions on Circuits and Systems for Video Technology, Jan 2006.

[39]A. Leon-Garcia, "Probability and Random Processes for Electrical Engineers," Addison-Wesley, 2nd ed., 1994.

[40]W. Allan, "*Time and Frequency (Time-Domain) Characterization, Estimation, and Prediction of Precision Clocks and Oscillators*," IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control, UFFC-34, 647-654, 1987.

[41]S. Boyd and L. Vandenberghe, "*Convex Optimization*," Cambridge, 2006.

[42]T Wiegand, H Schwarz, A Joch, F Kossentini, G.J. Sullivan, *"Rate-constrained coder control and comparison of video coding standards,"* IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, p688-703, 2003.

[43]IEEE 802 Working Groups, http://grouper.ieee.org/groups/802/11/.

[44]3GPP Official Website, http://www.3gpp.org/.

[45]S. Karande and H. Radha, "Does relay of corrupted packets increase capacity?," IEEE WCNC, Mar. 2005.

[46]F. Long, K-L. Lo, W-C. Siu, and F. Long, "Effect of channel quality estimation error on the performance of interactive mobile video system," ISIMP, May 2001.

[47]S. Ray, M. Médard, L. Zheng, and J. Abounadi, *"On the sublinear behavior of MIMO channel capacity at low SNR,"* ISITA, Oct. 2004.

[48]R. Adler, R. Feldman, and M. Taqqu (Eds.), A Practical Guide to Heavy Tails: Statistical Techniques and Applications, Birkhäuser, 1st ed., Oct. 1998.

[49]H. S. Wang and N. Maoyeri, "Finite-state Markov channel – A useful model for radio communication," IEEE Trans. Veh. Tech., (44)1, 163–171, Feb. 1995.

[50]Q. Zhang and S. A. Kassam, "Finite-state Markov model for rayleigh fading channels," IEEE Trans. Comm., (41)11, 1688–1692, Nov. 1999.

[51]S. A. Khayam and H. Radha, "On long-range dependence in wireless residual channels," CISS, Mar. 2005.

[52]Software for Low Density Parity Check Codes, "http://www.cs.toronto.edu/~radford/ftp/LDPC-2006-02-08/index.html," last accessed on Nov 10.

[53]R. W. Yeung, "A First Course in Information Theory," Kluwer Academic/Plenum Publishers.

[54]ITU-T Rec. H.264 | ISO/IEC 14496-10 AVC, "Advanced Video Coding for Generic Audiovisual Services," version 8.9, 2007.

[55]Y. Cho, S. S. Karande, K. Misra, H. Radha, J. Yoo, and J. Hong, "*On Channel Capacity Estimation and Prediction for Rate Adaptive Wireless Video s*," to be appeared in IEEE Transactions on Multimedia.

[56]Y. Cho, H. Radha, J. Y, and J. Hong, "A Rate-Distortion Empirical Model for Rate Adaptive Wireless Scalable Video," CISS 2008, March 2008.

[57]"ANSI/IEEE Std 802, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," 1999.

[58]"ANSI/IEEE Std 802, Part 15.4: Low Rate Wireless Personal Area Networks," Sep 8, 2006.

[59]Y. C. Hu and D. B. Johnson, "Design and demonstration of live audio and video over multihop wireless ad-hoc networks," in: Proceedings of the MILCOM2002 (2002).

[60]C.M. Cordeiro, H. Gossain, and D.P. Agrawal, *"Multicast over Wireless Mobile Ad-hoc Networks: Present and Future Directions,"* IEEE Network, vol. 17, no. 1, 2003, pp. 52-59.

[61]J. N. Laneman and G. W. Wornell, "*An Efficient Protocol for Realizing Distributed Spatial Diversity in Wireless Ad-Hoc Networks,*" in ARL FedLab Symposium on Advanced Telecommunications and Information Distribution (ATIRP-2001), College Park, MD, March 2001.