SOURCE LEVEL SEQUENCE BASED PROCESSING FOR SIGNAL COMPRESSION AND RELIABLE TRANSMISSION

by

Raghavendra Singh

A Dissertation Presented to the

FACULTY OF THE GRADUATE SCHOOL

UNIVERSITY OF SOUTHERN CALIFORNIA

In Partial Fulfillment of the

Requirements for the Degree

DOCTOR OF PHILOSOPHY

(Electrical Engineering)

Sept 2001

Abstract

In this thesis we are proposing algorithms in the areas of signal processing and networking to enable efficient and reliable transmission of multimedia data. This work can be divided into three parts, the first part deals with source encoding, the second with channel decoding and the third with channel modeling.

Multimedia data is correlated and many popular source coders take advantage of this correlation to improve their performance, for example, predictive quantization, context based entropy coding and spatially adaptive thresholding for denoising. Clearly for these memory based systems, performance can be improved by permitting the processor to lookahead for several samples, i.e., permitting the processor to make decisions based not only on past samples but also on future samples.

In the source coding community the advantages of this "delayed decision" approach have long been recognized, but popular usage of this approach has been limited by the associated increase in complexity of the source encoder. In this thesis, we propose a general framework for reducing the complexity of the lookahead search for memory based source coders. We have applied our algorithm to rate distortion optimization in an adaptive context based entropy coding environment. A compression and a denoising algorithm for images have been developed; these algorithms achieve better results than the state of the art coders and denoisers. In addition, the results are comparable to results obtained by the traditional, more complex, lookahead search algorithms.

On the other hand, in channel decoding the delayed decision approach is very popular. However the memory of the system is mostly provided either by the intersymbol-interference in the channel or by the forward error correction codes like convolution codes. There has been limited work on incorporating source properties and using the memory of the source in channel decoding. In this thesis we propose a channel decoder based on maximum likelihood sequence estimation (MLSE) for predictive coding environments, where the source information is supplied by using the multiple description coding (MDC) format. MDC is a novel forward error correction scheme which does not require retransmission and provides graceful degradation, hence it is very popular for multimedia data. Our MLSE channel decoder exploits the correlation across descriptions and within a description to estimate data lost due to noise in a description. This allows the MDC decoder to decode at a higher SNR and also reduce error propagation.

The last part of work is on network modeling. Given the best-effort nature of the current Internet, it is up to the application to adapt to a wide range of network conditions. To allow the applications to be adaptive, simple and efficient network loss models are needed so that the applications can predict the future and adopt the optimal strategy. We have developed a model for the temporal dependence between losses in network traffic, based on the universal modeling concept of Risannen which shows a marked improvement over the traditional models based on the Markov chains.

Acknowledgment

I would like to first and foremost express my gratitude to my advisor Professor Antonio Ortega. His support and patience helped me tremendously in all these years at USC. His insights and the long discussions I had with him made work both fun and interesting.

I would also like to thank Professor Keith Chugg and Professor Christos Papadopoulos for serving on my dissertation committee. Professor Vijay Kumar and Professor Jay Kuo were on my qualifying committee and they helped me define my research problems. I would like to thank all of them for giving me their valuable time.

I would like to thank my group mates, some of whom I collaborated with, for helping in various stages of my work. They let me listen to loud music in the lab. which made work so much more easier! I would like to thank all my friends they have listened to me patiently over the years espousing one of my many theories of life. I will dearly value all the new friendships that I made in Los Angeles.

Last but not the least, I thank my parents and my sisters for the encouragement and support they have given me over the years.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Advances in the speed and degree of integration of digital circuits have led to a steady increase in the popularity of digital signal communications. The ever growing Internet, high definition television broadcast and mobile wireless telephony are examples of digital communication systems that have had considerable commercial success in the past decade.

To motivate the work proposed in this thesis, let us first examine a generic communication system (Fig. 1.1). It is composed of five main elements, a source encoder, a channel encoder, a communication channel, a source decoder and a channel decoder. The source encoder's task is to minimize the amount of information needed to reproduce the source at the decoder, enabling efficient transmission and storage of the source. The channel encoder's task is to facilitate reliable transmission over noisy channel conditions. It is the case with all communication systems that the received signal can differ from the transmitted signal due to various channel impairments (attenuation, thermal noise, or plain congestion). For digital signals, bit errors are introduced in transmission, i.e., a binary 1 is transformed into 0 and vice versa. These bit errors can lead to a severe degradation in the quality available at the receiver; the channel encoder provides the means to combat these channel errors. The channel decoder and the source decoder perform the opposite task of the encoders and will not be discussed here.

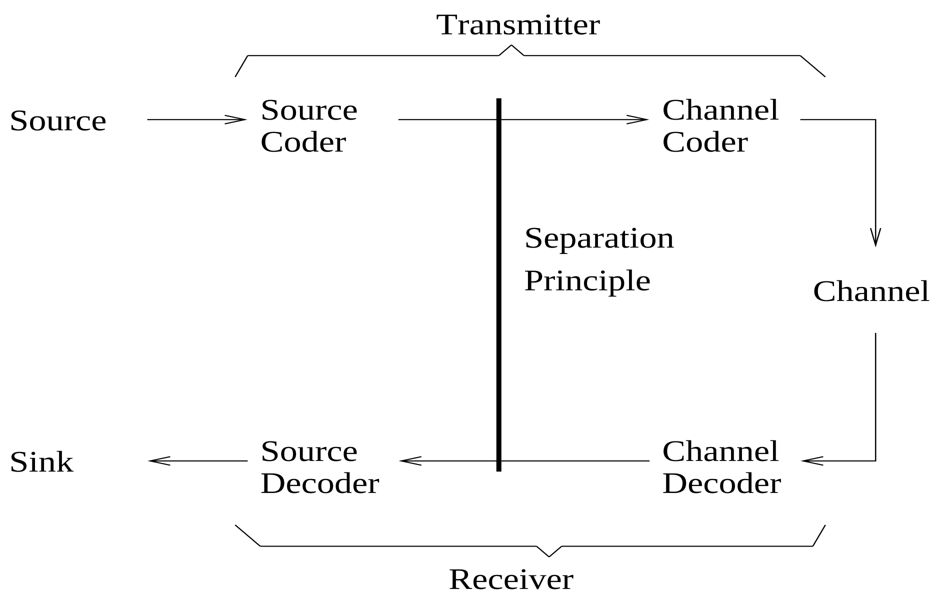


Figure 1.1: A communication system illustrating Shannon's separation principle.

The number of users and the volume of data on these communication systems has been growing steadily, often taxing the limited resources. But it is the change in nature of the traffic that has raised several new challenges for digital communication system design. Multimedia traffic has exploded in the past couple of years; already MP3 is the most searched keyword on the Internet and it was reported [6] that in November 1999 listeners spent nearly a total of 2 million hours tuned to Internet audio. With 3G wireless technology, the promise of multimedia data delivered to a hand-held device is also becoming a reality.

Multimedia data have two important characteristics that differentiate it from traditional (textual) data. First, there is a strict transmission delay imposed on it, such that if data is not received at its associated playback time, it would be considered lost. This is a problem especially in the present day Internet where there are no guarantees on the quality of service and delays are common due to congestion over the network. Second, the nature of the multimedia data is such that it is possible to lose some information and still decode it to a visually acceptable quality. Loss can occur randomly due to channel errors or be introduced in a controlled manner as in a lossy compression algorithm.

The huge amount of "data" contained in a multimedia signal implies that there is a need for compression of multimedia signals. For example, an uncompressed bi-level document would require a storage space of about 1.2 Mbytes and about 30 minutes for transmission over standard telephone lines. Lossless source coding, as employed used in the FAX standard, reduces the storage space by a factor of 40 along with a proportional decrease in transmission time. In another instance, a 45 second piece of uncompressed *Handel's Water Symphony* is about 4Mbyte and requires 40 seconds to download from a source in Columbia University, New York, to a receiver in Univ. of Southern California, Los Angeles. However, by using the MP3 standard this signal can be lossy compressed, without any perceptible loss in quality, to about 0.3Kbytes! Thus powerful compression techniques have been developed and standardized for multimedia signals, e.g., JBIG [7] for bi-level image coding, JPEG [8] for color image coding, MPEG [9] for video coding and the popular MP3 format for audio coding.

One of the disadvantages associated with compression is that in a compressed signal, the bit errors due to channel imperfections can be catastrophic. For example, in predictive coding, often used for audio signals, a single error will propagate to the end of the signal due to the feedback loop in the decoder. To combat such channel errors and make efficient use of the limited bandwidth, a channel encoder either adds redundancy to the transmitted data as in forward error correction (FEC) codes or allows for retransmission of the lost data. Significant progress has been made in the area of channel encoding. The development of turbo codes [10] and rate compatible punctured convolutional codes [11] are important achievements of the 1990s. At the same time, the networking community has been developing reliable protocols, some of which use FEC as the means for reliable communication [12], while others employ automatic retransmission request (ARQ) [13] to reliably transmit information.

Despite these advances in compression and transmission for digital communication systems, the explosion in usage of multimedia data has necessitated continued research in these areas. Bandwidth and disk storage space, though cheaper, still comes at a premium. Algorithms which can use properties of the source to efficiently process data and improve their performance are of considerable interest to both the academic and the commercial world.

A property common to many multimedia sources is that they are correlated, i.e., given the past and present samples, the future samples can be predicted with a high probability. Many popular source processors take advantage of this memory to improve their performance. Predictive quantization [14], context based entropy coding [3, 7] and spatially adaptive thresholding for denoising [15] are examples of applications where memory based systems have been used. For these memory based systems, performance can be improved by permitting the processor to lookahead for several samples, i.e., permitting the processor to make decisions based not only on past samples but also on the future samples [16].

In the source coding community the advantages of this "delayed decision" approach have long been recognized, examples include trellis and tree quantizers [16], smoothed DPCM codes [17] and entropy constrained halftoning [18]. However the popular usage of this approach has been limited by the associated increase in complexity of the source encoder [16]. In this thesis, we propose a general framework for reducing the complexity of the lookahead search for memory based source coders. We have applied our algorithm to rate distortion optimization in an adaptive context based entropy coding environment. Compression and denoising algorithms for images have been developed; these algorithms achieve better results than the state of the art coders and denoisers. In addition, the results are comparable to results obtained by the traditional, more complex, lookahead search algorithms.

On the other hand, in channel decoding the delayed decision approach is very popular. However memory of the system is provided either by the inter-symbol-interference in channel [19] or by the forward error correction codes like convolution codes [20]. There has been limited work on incorporating source properties and using the memory of the source in channel decoding. An exception is the work by Sayood and Borkenhagen [21], where the "residual" memory in the DPCM generated prediction errors is used for maximum likelihood channel estimation (MLSE) based channel decoding. A similar algorithm has been developed by Miller and Park [22]. (A disadvantage of these algorithms is that sub-optimal assumptions, like use of training data sets, have to be made because sufficient source information is usually not available at the decoder.

In this thesis we propose a MLSE channel decoder for predictive coding environments, where the source information is supplied using the multiple description coding [23] (MDC) format. MDC is a novel forward error correction scheme which does not require retransmission and provides graceful degradation, making it very popular for multimedia data [24]. In MDC two or more descriptions of the source are generated and sent over independent channels, each description is independently decodable and each additional description improves decoding performance. In MDC systems, if there are channel errors in a description, for data encoded using a predictive coder, then the description is discarded [5]. Given that the descriptions are correlated this is clearly a waste of resources. Our MLSE channel decoder exploits the correlation across descriptions and within a description to estimate data lost due to noise in the description. This allows the MDC decoder to decode at a higher SNR and reduce error propagation.

Another contribution of this thesis is towards modeling of temporal dependencies in packet loss over the Internet. For Internet based multimedia applications, e.g., video conferencing tool, radio etc., since the Internet does not provide a guarantee on quality of service and packet losses are common, the application has to adapt to the changing network conditions so as to provide good quality to the end-user. To allow the applications to be adaptive, simple and efficient network loss models are needed so that the applications can predict the future and adopt the optimal strategy. We have developed a model for the temporal dependence between losses in network traffic, based on the universal modeling concept of Risannen [25], which shows a marked improvement over conventional models based on the Markov chains [1].

The sequence based source encoder and channel decoder is discussed in detail in Sections 1.2 and 1.3 respectively. The network modeling algorithm is discussed in Section 1.4 and the chapter concludes with an overview of contributions of the thesis in Section 1.5.

1.2 Source Encoding of multimedia dat

Context based entropy coding is an example of a memory based system. Let us use this example to motivate the advantages of a lookahead search in memory based encoders. In a context based entropy coder, the quantized samples are entropy coded (usually using arithmetic coding) based on different probability models. These models, static or adaptive, are selected based on the values of the previously coded samples, i.e., on the *context*. In most compression algorithms that use a context based entropy coder, the mode of operation is to quantize the data first on a sample by sample basis and then entropy code it. This is clearly suboptimal as the number of bits required to code a sample depends upon its context, i.e., upon the past encoded samples. In other words, the quantization decisions on the past samples will affect the coding bits for the present sample. Thus, if quantization decisions are taken independently for each sample, the quantization that is best for a sample (in terms of distortion) could negatively affect the coding of future samples.

The advantages of a sequence search for memory based systems have long been recognized [16], but it has not been used much because of the associated increase in complexity. Without getting into details here, the increase in complexity can be considered to be proportional to the memory of the system. For source coders with infinite memory, i.e., memory which expands with each encoding operation, such as those using predictive coding, the complexity of an optimal sequence search is prohibitive. Even for systems where the memory is finite, but large, the cost of an optimal lookahead search can be very high [16]. Thus suboptimal sequence search algorithms like the M-L algorithm [18] and the iterative decision feedback equalization (DFE) [2] are usually used to provide a reduced complexity solution.

In this thesis, we propose an alternative suboptimal sequence search algorithm for the memory based encoders. The novelty of our work is to demonstrate that the per survivor processing (PSP) principle [26] can be used to perform a *reduced complexity, one-pass* lookahead search for source coding environments, with its performance being comparable to that of traditional, (M-L, DFE), suboptimal algorithms. Our approach performs a search where the number of *states*, where state is a variable which completely captures the memory of the system, is reduced through pruning. The basic idea is to preserve the structure of the state space (i.e., the memory of the state is maintained) but to compare states that have some common information (e.g., compare states that have "similar" values for recent intervals in time, while they differ for past intervals). Then only the best states among these states are kept while the others are pruned. Obviously this approach precludes achieving the optimal solution but the rationale for it is that if the pruning is done in sets of similar states then the loss in performance will be small. PSP has been popular in adaptive MLSE channel decoding when the channel parameters are unknown. To the best of our knowledge it has only been applied in source coding in Predictive TCQ [27], where it has been used to estimate the prediction error at any state of the trellis. In our work we present a systematic approach to apply PSP to a source coding environment; the extension is non-trivial as each scenario a new and appropriate state model has to be formulated.

We have specifically considered the context based adaptive entropy coding environment as an example of non-finite memory coding systems. However, this work should serve as motivation towards the application of PSP to other source coding applications involving lookahead [28]. Three different applications have been considered in this thesis. First, we consider bi-level images where quantization implies flipping of pixels. Our sequence based quantizer could be used within the JBIG standard to introduce losses in the image. This not only reduces the bit-rate but is often helpful in removing noise in the image, in particular pepper noise. The second application that we have considered is the quantization of wavelet coefficients in a gray-level image coder. The underlying coder is based on the work by Chrysafis and Ortega [3]; our results show an improvement over the symbol based quantizer used by the authors. Removal of random noise from a gray-level image is the third application. We work in the wavelet domain and use the principle of Occam filter [29] to denoise the image, i.e., we use signal compression for denoising the image. In addition, we have also implemented a one pass algorithm for denoising of images using spatially adaptive thresholding, with results that are comparable to the multi-pass algorithm of Chang *et al.* [15].

1.3 Reliable Transmission of multimedia data

As shown in Fig. 1.1, source and channel coders are often designed *separately* in communication systems. This is on the basis of Shannon's separation principle [30], which states that in a point-to-point transmission and for infinite source length, source and channel can be separately optimized, as long as the source code produces a bit rate that can be carried by the channel capacity. However, in practical systems, due to constraints on decoding delay and complexity, arbitrarily long sequences cannot be used; this limits the applicability of the principle but not its usage. Researchers have started recognizing that coupling between the source and channel coder can give substantial gains in performance or a reduction in complexity of the coder and there is extensive ongoing work in the area of joint source channel coding/decoding (JSCC) [31, 32].

An example of a JSCC is a scheme that recognizes the priorities that occur naturally in multimedia signals and accordingly offers higher protection to the data with higher priorities. For example in scalable schemes like layered coding (LC), e.g., progressive JPEG, the signal is coded hierarchally into different resolutions (layers) and each resolution can improve the decoding quality if and only if it is decoded in the same hierarchal order in which it was encoded. Thus there is a dependency between layers and there is a prioritization where the layer which was encoded first has the highest priority while the one which was encoded last will have the least priority. A JSSC scheme like priority encoding transmission (PET) [33] will assign more channel coding bits to the lower layers (layers encoded first), and less channel coding bits to the higher layer.

Another example of JSSC is MDC where the redundancy is added in the source domain. Two or more descriptions of the source are generated and sent over different channels to the receiver, as shown in Fig. 1.2. If only one description is received, the decoder can reconstruct the signal to a minimum level of distortion. However, if both descriptions are received, information from one channel augments information from the other and it is possible to achieve a lower level of distortion than with a single channel. Thus MDC is **robust** due to the redundancy of the multiple descriptions of the same source and it is **scalable** as each correctly received description improves the decoder performance. Also MDC **does not require prioritized** transmission, as each description is independently decodable. In the figure we have shown only two descriptions but this can be easily extended to multiple (more than two) description systems.

The first contribution of our work is to compare the two scalable schemes, LC and MDC. Clearly if there is no channel noise, LC will outperform MDC. However

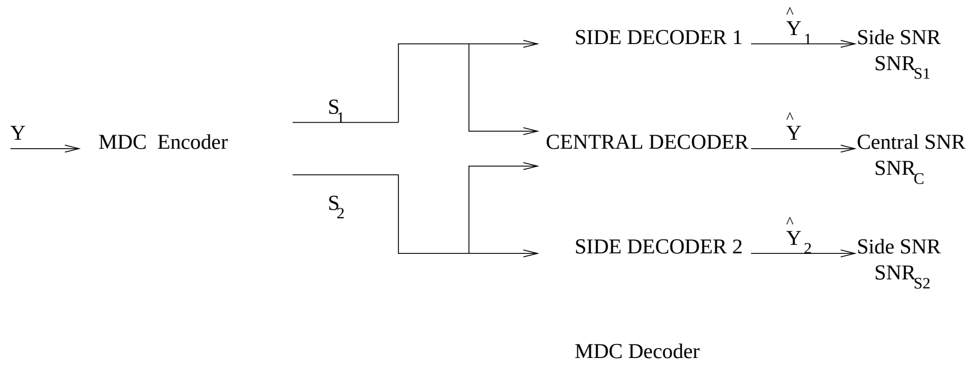


Figure 1.2: Multiple Description coding and decoding. S_1 and S_2 are the two descriptions of the source Y.

in presence of noise and with no protection, especially to the higher priority layers, the performance of LC will degrade sharply. A JSCC scheme for LC based on automatic retransmission (ARQ) is proposed in this thesis. The base layer, i.e., the layer which was encoded first, is sent first with ARQ and, given the time constraints associated with multimedia data, the higher layers are transmitted next without any protection. Thus a form of selective retransmission [34] is used to ensure that the base layer gets across error free before other layers are transmitted.

This LC based JSCC scheme is compared with THE MDC scheme of Jiang and Ortega [4] under lossy packet network conditions. The cost of a retransmission in JSCC varies across different situations and will determine whether ARQ can be used. For example, in a point-to-multipoint communication even a single retransmission can be prohibitive due to the NAK implosion at the server. For point-to-point links, the number of allowable retransmissions will depend upon the round-trip-time (RTT) of the link and the latency of the application. In our simulations we have varied these parameters to find scenarios under which MDC would be better than ARQ. The results show that ARQ is not a good solution for links with long RTT (intercontinental) or for applications with very short latency. In addition, we have compared layered coding to MDC for a receiver driven layered multicast scenario [35]. The results show the advantage of using MDC in a multicast environment.

The second contribution of this thesis towards reliable transmission of multimedia sources is in developing a sequence based channel decoding algorithm. In particular, predictive encoding environments are considered for two reasons. First, these environments are commonly used in multimedia compression schemes, e.g., motion prediction in video coding or DPCM in audio coding. Second, in these environments channel noise can have catastrophic effect at the decoder due to the feedback loop in the decoder. Most traditional schemes limit error propagation by periodically restarting the prediction loop but do not allow recovery of lost data.

MDC has been proposed as a means of robust transmission of predictive encoded multimedia data [24]. However, in most MDC schemes for predictive source coders, if there is an erasure in a description the description is discarded, i.e., no attempt is made to estimate data lost due to channel noise [5]. This is true even if the second description is received error free. Given that the descriptions are correlated and that most sources have memory, this is a waste of resources and motivates the need for an error recovery algorithm.

The major contribution of our work is an algorithm which estimates the lost data through processing at the decoder. The algorithm is based on maximum likelihood estimation of the erased samples, in say S_2 (Fig. 1.2), where likelihood is defined in terms of a distance measure between the *estimated* $\hat{Y}_2$ and $\hat{Y}_1$ with the added constraint that the *estimated* $\hat{Y}_2$ sequence be *consistent* with all the error-free data that has been received. Thus, in the estimation algorithm we exploit both redundancy between descriptions as well as the encoder memory. As consistency is a key part of our algorithm we will call it consistent sequence estimation algorithm (CSE).

Forward error correction (FEC) codes could also be used for local recovery at the decoder. The main disadvantage of FEC codes is that they experience the *cliff effect* [24]; their performance is constant for up to some e erasures and then drops very sharply for more than e losses. Our scheme, on the other hand, results in *graceful degradation*; as the length of a burst of erasure grows, the performance of the CSE degrades gracefully.

1.4 Modeling of temporal dependency in packet loss using information theory concepts

A number of studies have shown that network traffic is correlated, i.e., the packet losses exhibit a finite dependence in time [1] and thus it can be modeled as a correlated random process. Specifically, if a network traffic trace is represented as a binary message, with the symbol one representing a lost packet and the symbol zero a received packet, then the channel can then be modeled as a binary Markov process, i.e., a process where the conditional probabilities of a symbol depend upon a function of a finite number of contiguous past observations. This function of the past observations will be referred to as the "context" of the symbol in this thesis. Hence, to define the random process and thus model the channel, a set of contexts and their associated conditional probabilities have to be found.

One such popular model for modeling the packet loss, i.e., the channel, is the Markov chain model [1]. In a Markov chain model of order k, the context of a symbol is the string of past k observations. Thus there are $N_C = 2^k$ contexts in this model. An advantage of the Markov chain is that a finite state machine (FSM) can be associated with it, where each state corresponds to a context and its associated conditional probabilities.

However, when fitting Markov chain models to the data by estimating the conditional probabilities of the contexts, a number of difficulties arise [36]. First, there is an explosive increase in the number of states (2^k) if the order of the model is increased to find the best fit. In the paper by Yajnik *et al* [1], their analysis shows that models of up to order 40 may be required to best fit the trace. Clearly the computational cost of such a large model will be very high.

Another problem with the Markov chain model is that some contexts occur very rarely in the data and when they are used for modeling the process they may not provide enough information about the process. This is referred to as the "context dilution" problem. Thus it may be more efficient to either delete these set of contexts from the model or alternately "lump" them together to form a new context. However, if the redundant parameters are removed by arbitrarily lumping together equivalent states, the result may not be a finite state machine implementation of the process, let alone one of a Markov type [36].

In this thesis, we use the universal modeling concept [36] to alleviate the problems of Markov chain models. A universal model is one which represents an entire class of probability distributions such that it is able to capture the behavior of any model in that class. From this set of models, the "best" model for the observed data is found. To define the "best" model we need a measure of the performance of a model. We use the concept of the "shortest description length" [37], where description length is defined as the negative logarithm of the probability distribution estimated for the given data, a concept similar to Shannon's entropy [30]. Thus to evaluate a model, the entropy of the observed data is calculated with respect to the model and the best model is the one which gives the lowest entropy.

An example of universal model is the tree model proposed by Rissanen [25]. This model will be referred to as the Markov tree model. A simplistic view of the model is that it is a collection of all possible Markov models, from which the most appropriate model is chosen for prediction of of the source [38]. In a kth order Markov chain model, memory of all the contexts is of the order k, in a kth order Markov tree model, all possible contexts with memory of order $1 \ldots k$ will be arranged in a tree. The nodes of the tree represent a context and its associated probability distribution. The best set of contexts, i.e., a set of connected nodes, for the observed data can be efficiently found using the Context algorithm developed by Rissanen [25]. Thus, rather than deleting or lumping contexts of the Markov chain, all possible contexts are grown and the best set is chosen, i.e., from an "overcomplete" model, a model which captures essential information of the source is chosen. We propose to use this concept for modeling of network traces, an area, where to the best of our knowledge, it has not been used before. Our results show the improvement in modeling performance of the tree models over Markov chain models.

1.5 Overview and Contribution of the thesis

The main contributions of this thesis are,

- *Reduced complexity lookahead search with applications to image coding and denoising.* The main novelty of our scheme is that it proposes a method to make sequence based decision in environments where traditionally lookahead had been computationally expensive. The proposed algorithm has been used for coding and denoising of images, using a adaptive context based encoder. This is presented in Chapter 2 of the thesis.

- *Comparison of MDC to layered coding.* We have compared the performance of MDC to layered coding in delay constrained environments using the network simulator *ns*. This is presented in Chapter 3 along with a brief review of existing MDC techniques.

- *Error recovery scheme in predictive coding environments using MDC.* In predictive coding, errors are catastrophic due to the feed back loop in the decoder. In other MDC based DPCM schemes, if there is an error in a description, the description is discarded. We show that the correlation between two descriptions along with the memory of the source can be used to recover the errors. We have proposed a novel sequence estimation algorithm for our error recovery algorithm. Chapter 4 covers this part of our work.

- *Modeling of temporal dependence in packet losses* We have presented a novel method of analyzing network traffic traces based on information theoretic concepts. We have also introduced new Markov models based on a tree structure which model the channel, i.e, the packet losses, in a better manner than traditional Markov chain models. This is presented in Chapter 5 of this thesis.

Chapter 2

Reduced complexity lookahead search with applications in image compression and denoising

2.1 Introduction

In spite of the cheap availability of bandwidth and storage space, due to the explosion in usage of multimedia data, efficient compression of the data has become a necessity. Given the correlation often present in the data, compression approaches that exploit this inherent memory to improve their performance have become very popular. In many of these approaches, the quantization and/or entropy coding operation depends not only on the current input but also on the past history of the output, i.e., a sample is coded (quantized or entropy coded) differently depending on the values of the past coded samples. Predictive quantization [14], context based entropy coding [3, 7] and spatially adaptive thresholding for denoising [15] are examples of applications where memory based systems have been used.

In these approaches, which have been called *recursive coding systems* [16], the influence of past outputs on current encoder operation can be completely captured by a *state* variable. Given an input sequence, the encoder produces both a sequence of channel symbols and a sequence of *states* [16]. These state based approaches can be broadly classified into the following categories:

- *Finite state coding systems.* In this case, the present encoder operation depends upon a finite number of past outputs. Hence a finite, constant, number of states can capture the memory in the system. Examples include the finite state vector quantizer [16], where the current state determines the codebook to be used to quantize the next vector.

- *Non finite state coding systems.* The present encoder operation in this system, is dependent on all past operations. Hence the number of states grows with each encoder operation for these systems. Predictive quantization and context based entropy coding with adaptive probability models are examples of such systems.

In most of the popular compression schemes where state based coders are used, the quantizer operates in an "instantaneous decision" fashion, i.e., given the present state and the present input it will produce an output with the minimum possible distortion. However this mode of operation is "greedy" in nature and could lead to a deterioration in performance. Due to the memory of the system it is possible that good short term decisions can lead to bad long term behavior. Let us take the example of a context based entropy coder. In a context based entropy coder the quantized samples are entropy coded (usually using arithmetic coding) based on different probability models. These models, static or adaptive, are selected based on the values of the previously coded samples, i.e., on the *context.* In most compression algorithms that use a context based entropy coder, the mode of operation is to quantize the data first on a sample by sample basis and then entropy code it. This is clearly suboptimal as the number of bits required to code a sample depends upon its context, i.e., upon the past encoded samples. In other words, the quantization decisions on the past samples will affect the coding bits for the present sample. Thus, if quantization decisions are taken independently for each sample, the quantization that is best for a sample (in terms of distortion) could negatively affect the coding of future samples.

Clearly, for these state based coders, the coding performance, in terms of bit-rate and distortion, can be improved by permitting the quantizer to look ahead for several samples, thereby permitting a better long term minimum distortion fit at the expense of increased complexity and delay. The advantages of this "delayed decision" approach have long been recognized, with examples including trellis and tree quantizers [16], smoothed DPCM codes [17] and entropy constrained halftoning [18].

The main disadvantage of using lookahead is the complexity of the encoder [16]. For the finite state coding systems, with limited memory, optimal sequence search can be performed efficiently by using the Viterbi Algorithm (VA) [39] over the associated trellis. The Trellis Coded Quantizer (TCQ) [40] is a classic example of a delayed decision finite state quantizer. However, if the memory is large then the cost of an optimal lookahead search will be very high and for non-finite state coding systems, the cost could be prohibitive. Hence suboptimal algorithms like the M-L algorithm [18] or approximate iterative algorithms [2] are traditionally used to perform a reduced complexity search.

In this chapter, we propose an alternative suboptimal algorithm for lookahead coding in state coding systems. The main novelty of our work is to demonstrate that the per survivor processing (PSP) principle [26] can be used to perform a *reduced complexity, one-pass* lookahead search for these environments with its performance being comparable to that of M-L and iterative algorithms.

In a PSP based lookahead search, the basic idea is to merge "similar" states (e.g., states that have same values for recent intervals in time, while they differ for past intervals) so as to form a trellis of a finite and small number of "merged-states". VA can then be used to find the optimal sequence in the trellis, with the modification that any uncertainty in the transition metrics are removed or reduced by data-aided estimation techniques [26]. In a typical example, this uncertainty could be due to imperfect knowledge of source parameters, such as the probability model needed for entropy coding or uncertainty could also be due to the truncation of memory associated with merging of states. The novelty of PSP is that it allows the reduction of uncertainty within the calculation of each transition metric in the Viterbi algorithm. The estimation is based on the data sequence associated with the survivor path of a state, i.e., the path with the least cost into the state. The motivation behind PSP is that the survivor sequence is the best approximation of the data seen so far [26].

PSP has been popular in adaptive MLSE channel decoding when the channel parameters are unknown. To the best of our knowledge it has only been applied in source coding in Predictive TCQ [27], where it has been used to estimate the prediction error at any state of the trellis. In our work we present a systematic approach to apply PSP to source coding environment; the extension is non-trivial, for each scenario a new appropriate state model has to be formulated.

We have specifically considered the context based adaptive entropy coding environment as an example of non-finite state coding systems. However, our work should serve as motivation towards the application of PSP to other source coding applications involving lookahead [28]. Three different applications have been considered in this paper. First, we consider joint entropy coding and quantization of bi-level images, where quantization consists of flipping of pixels. Our sequence based quantizer could be used within the JBIG standard to introduce losses in the image. This not only reduces the bit-rate but is often helpful in removing noise in the image, in particular pepper noise. The second application that we have considered is the quantization of wavelet coefficients in a gray-level image coder. The underlying coder is based on the work by Chrysafis and Ortega [3], our results show an improvement over the symbol based quantizer used by the authors. Removal of random noise from a gray-level image is the third application we have considered. We work in the wavelet domain and use the principle of Occam filter [29] to denoise the image, i.e use signal compression to denoise the image. In addition, we have also implemented a one pass algorithm for spatially adaptive thresholding for denoising, the results being comparable to the multi-pass algorithm of Chang *et al.* [15].

The chapter is organized as follows. We start with the general design algorithm in Section 2.2. In Section 2.3 we introduce the basic concept behind context based adaptive entropy coding and Section 2.4 motivates lookahead search in a context coding environment. Then, in Section 2.5, we provide a series of experimental results to demonstrate the benefits of our approach and compare with other traditional approaches. A review of relevant work is given in the associated sections.

2.2 PSP based reduced state space lookahead search

In this section we present an algorithm for a reduced state sequence search for state based systems using the PSP principle.

We start with some definitions, let y represent the past encoding operations, where each y_i can take any of N_q possible values. If the memory of the state system is M, then a state, at the ith stage, can be written as the M-tuple

$$s_i = (y_{i-1}, y_{i-2} ... y_{i-M}). \tag{2.1}$$

Given this definition of state, the transition rule $s_i \rightarrow s_{i+1}$ is well defined for all y_i. State explosion can occur if $M = i$, as in a non-finite state system. It can also occur if $M < i$ but N_q is large in a finite state system.

We can also write the state s_i as,

$$s_i = (f(y_{i-1} ... y_{i-\hat{M}}), g(y_{i-1} .... y_{i-M})) \tag{2.2}$$

where $\hat{M} \leq M$ and $f()$, $g()$ are functions defined such that equations (2.1 & 2.2) are equivalent. Let us further define ms_i as,

$$ms_i = f(y_{i-1}, y_{i-2} ... y_{i-\hat{M}+1}) \tag{2.3}$$

and $\mathcal{P}_{ms_i}$ as,

$$\mathcal{P}_{ms_i} = g(y_{i-1} ... y_{i-M}). \tag{2.4}$$

The basic idea behind PSP is to compare, at every instant, states which have the same value for ms_i; these are what we referred to as similar states. From among these similar states, keep the one which has the lowest cost and prune the rest thus allowing a reduced state search. Note that though the number of states is being reduced, the memory of each state is not affected.

The easiest way to implement this reduced state search is to use the function $f()$ to merge similar states, i.e., define ms_i as a "merged-state" where all the states which have the same value for $f(y_{i-1} ... y_{i-\hat{M}})$ are merged. The function $f()$

could be defined such that, N_s, the number of different values $f()$ can take is limited to a small number.

A trellis of these merged-states can be defined, however note that these merged-states do not form a finite state machine. With each merged-state there is the "uncertainty" of $\mathcal{P}_{ms_i}$; unless $\mathcal{P}_{ms_i}$ is known we cannot define the transition rule. This is because unless ms_i and $\mathcal{P}_{ms_i}$ are both known the state s_i cannot be defined, hence the associated finite state machine cannot be defined. Here we are making that the state s_i is the smallest unit representing the memory of the system.

Thus in the trellis of merged-states the uncertainity of $\mathcal{P}_{ms_i}$ needs to be estimated. PSP allows this estimation within the Viterbi Algorithm by using the history of the survivor path, i.e., the path with the least cost into the merged state, to estimate the "parameter set" $\mathcal{P}_{ms_i}$. If $(y_{i-1}^{{ms_i}^S} ... y_{i-M}^{{ms_i}^S})$ is the history associated with the survivor path of merged-state ms_i then mathematically parameter set is defined as,

$$\mathcal{P}_{ms_i} = g(y_{i-1}^{{ms_i}^S} ... y_{i-M}^{{ms_i}^S}). \tag{2.5}$$

This can also be written as the recursive equation,

$$\mathcal{P}_{ms_i} = g(\mathcal{P}_{ms_{i-1}^S}, y_{i-1}^{ms_i^S}), \tag{2.6}$$

with the right initializations. Using VA, the survivor state and path are found for a state and its the parameter set is updated using the survivor information. A PSP based trellis is shown in Fig. 2.1.

With the above formulation the problem of reducing complexity in lookahead search amounts to finding the right $f()$. (Given $f()$, $g()$ will be automatically defined). Definition of $f()$ will vary with the environment considered, in the next section we design a $f()$ for context coding environment. Before ending this section we present the modified VA algorithm.

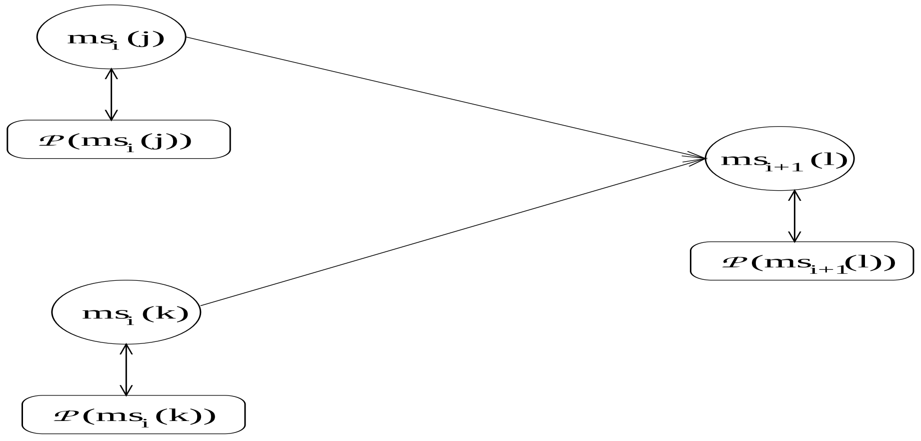


Figure 2.1: A PSP based trellis is shown in the figure. With each merged-state $ms_i(j)$ there is an associated parameter set $\mathcal{P}(ms_i(j))$. The parameter set carries the history of the surviving path into the state. Given the merged-state **and** the parameter set, the transition rule is well defined. VA is used to find the survivor path, i.e., the path with the least cost, for state $ms_{i+1}(l)$. If the survivor path, for example, is $ms_i(j)- > ms_{i+1}(l)$ then the parameter set of $ms_i(j)$ is copied into parameter set of $ms_{i+1}(l)$ and updated with the symbol associated with the transition.

Lookahead search algorithm

Let $BM(ms_{i-1} \to ms_i)$ be the branch metric, associated with transition $ms_{i-1} \to ms_i$. Let the set of merged states $\{ms_{i-1}\}_{ms_{i-1} \to ms_i}$ be all the merged states in stage $i-1$ which transit to the state ms_i in stage i.

```
For each stage $i$
    For $j = 1..N_s$
        For state $ms_i(j)$ find the survivor state:
            (1) $ms^S_{i-1}(ms_i(j)) =$
            $arg\ min_{\forall ms_{i-1} \in \{ms_{i-1}\}_{ms_{i-1} \to ms_i}} J(ms_{i-1}) + BM(ms_{i-1} \to ms_i(j))$
            (2) Compute $J(ms_i(j)) = J(ms^S_{i-1}(ms_i(j))) + BM(ms^S_{i-1}(ms_i(j)) \to ms_i(j))$
            (3) **Update** $\mathcal{P}(ms_i(j))$ **based on the survivor** $ms^S_{i-1}(ms_i(j))$.
            end
        **Derive, using the parameter set, the transition rule** $ms_i \to ms_{i+1}$ **.**
    end
end
```

The above algorithm is the Viterbi algorithm with the steps in bold font being the additional PSP step. Step (4), uses the parameter set $\mathcal{P}(ms_i(j))$ to derive the transition rule for the $i+1^{th}$ stage. With each state we are storing a parameter set, hence the memory requirements are more than a standard VA (with the same number of states). However, using this parameter set we can reduce the number of sets by a significant factor, in our context based scheme from a minimum of 2^{15} states to just 15 states. The additional operations in a PSP-VA are the update of the parameter set, equation (2.6), and the derivation of the transition rule.

2.3 Examples of Context Coding Environment

The encoder and decoder of a typical context based system are shown in Fig. 2.2. Refer to the figure for notation used in this section.

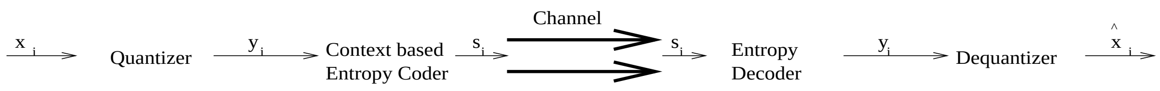


Figure 2.2: A generic context based encoder and decoder are shown. x_i is the input which is quantized to produce the quantization index y_i. y_i is encoded by the entropy coder based on the probability model $P(y_i|c(y_i))$ where $c(y_i)$ is the context information for sample y_i. The binary symbol s_i is transmitted to the decoder where it is entropy decoded to produce y_i. The dequantizer reconstructs the input to $\hat{x}_i$.

In the general case the context information for a sample is defined as the information that can be obtained from the neighboring samples. In this paper we assume that only causal neighbors are used to calculate the context, this avoids having to send any side information to the decoder. Thus, for a one dimension signal the context information corresponding to the index y_i is the function $c(y_i) = f(y_{i-1}, y_{i-2}....)$. The key issue in context coding is to find efficient contexts, i.e., those that approximate the "true" statistics of the source; obviously, the exact nature of the context information will be application specific.

The first step in defining a context is to select a neighborhood template. An example neighborhood template for a two dimensional source is shown in Fig. 2.3. For this template, let $y_i,\ i = 1...K$ be the indexes in one line of the 2D source, where K is the width of the line and let w_i be the context information derived from these indexes. One line of the context information is kept; at the top of the source all w_i are set to zero and *after* encoding of index y_i, w_i is updated. The rule for updating can vary with the environment.

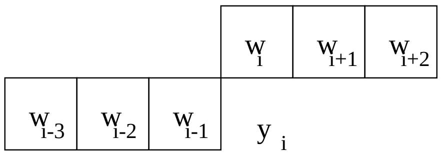


Figure 2.3: An example of a template used in context coding. The sample to be encoded is the quantization index y_i and the context is derived from the previously encoded neighboring indexes y_i using context information $w_i = g(y_i)$. The context could be simply a string of the neighboring pixels as in the bi-level case or it could be a mathematical function of the neighboring wavelet coefficients as in the gray-level image coding case.

The second step in context coding is to derive the context from the context information in the template. Let us explain the above steps using the examples of bi-level and gray-level image coding.

In bi-level image coding the context information is usually derived in the image domain, i.e, x_i are pixels of the image. If no quantization is used then, $\hat{x}_i = y_i = x_i$. Quantization in the bi-level case implies flipping of pixel value, i.e., $\hat{x}_i = y_i = 1 - x_i$. An example of context coding in bi-level images could be that the context is a string formed by concatenating the w_i in the template, e.g., if the indexes in the given template are $w_i = 0, w_{i+1} = 1, w_{i+2} = 1, w_{i+3} = 0, w_{i-2} = 0, w_{i-1} = 0$, then $c(y_i) = (011000) = 48_{10}$ (to the base 10). After encoding y_i, w_i is updated using the rule, $w_i = y_i$.

On the other hand, for gray-level image compression, context based entropy coding is usually done in the transform domain, i.e., the image is transformed, e.g. by wavelet transform [3], quantized and the context is formed from the quantized coefficients. Let x_i, $i = 0..K - 1$ be the wavelet coefficients in one line of a certain subband, where K is the width of the line, then y_i are the corresponding quantization indexes on this line. Starting at the top of the subband with all w_i

equal to zero we update each w_i *after* we encode an index y_i as follows:

$$w_i = \begin{cases} |y_i| & \text{if } y_i \neq 0 \\ w_i/2 & \text{otherwise} \end{cases}$$

Based on the w_i the context $c(y_i)$ for index y_i can be derived as:-

$$\alpha_i = w_i + 2w_{i-1} + w_{i+1} + (w_{i-3}||w_{i-2}||w_{i+2}||w_{i+3}) \tag{2.7}$$

$$c(y_i) = \begin{cases} 14 & \text{if } \alpha_i > 2^{14} \text{ -1} \\ 0 & \text{if } \alpha_i\text{=0} \\ 1 + log_2\lfloor\alpha_i\rfloor & \text{otherwise} \end{cases} \tag{2.8}$$

For more information on the gray-level context, see [3].

A finite number, N_c, of contexts is usually selected in context coding to avoid the problem of context dilution [41]. In the example for bi-level image coding given above, context $c(y)$ can take on at most $2^6 = 64$ values, while for gray-level image coding, quantization ensures that the context can take at most 15 distinct values.

After deriving the context for an index, the probability $p(y_i = q_j|c(y_i) = z_k)$, where z_k, $k = 1..N_c$, are the possible context values and q_j, $j = 1...N_q$, are the possible quantization indexes, is estimated. This probability may be estimated from fixed probability tables, derived either from test images or from a probability distribution function that matches the source characteristics. For example, generalized Gaussian distributions are often used to characterize wavelet coefficients. As mentioned in the introduction, adaptive models are more commonly used. One way of performing adaptive modeling is to store the counts, $n(y_i = q_j|c(y_i) = z_k)$, i.e., the number of times $y_i = q_j$ occurs with context $c(y_i) = z_k$. Each time a y_i follows a given context $c(y_i)$, $n(y_i|c(y_i))$ is increased by one. The probability $p(y_i = q_j|c(y_i) = z_k)$ is estimated by

$$p(y_i = q_j|c(y_i) = z_k) = \frac{n(y_i = q_j|c(y_i) = z_k)}{\sum_{j=1}^{N_q} n(y_i = q_j|c(y_i) = z_k)}$$

To prevent the probability model from saturating and to allow it to continue learning the statistics of the source, either a forget factor is used to forget the past or if the counts exceed a large number the counts are halved. The last step in a context based entropy coder is to entropy code the sample y_i using the probability model $p(y_i|c(y_i))$. In this paper we will be using an arithmetic coder as the entropy coder, though other entropy coders could also be used. Note that in this section, we have introduced the context encoder that is used in our applications, however, the reduced complexity lookahead search algorithm can be used in any fixed template, causal, context coding environment.

2.4 Lookahead in context coding environment

Let us start this section by further motivating lookahead for context coding environment. In Fig. 2.4 two binary bitstreams A and B as shown. Assume that a one dimensional, 1st order (i.e one past pixel) context and fixed probability model is being used. The probability models are such that $p(0|0) >> p(1|0)$ and $p(1|1) >> p(0|1)$, i.e, the probability of seeing "runs" of symbols is high, which is a reasonable assumption to make for image sources.

In a traditional symbol based approach, the cost (distortion and rate) of coding pixels as is or flipping them are compared for each individual sample. At the first one pixel (shown by the arrow), as the probability $p(0|0)$ is very high, the rate gain of flipping the 1 to 0 will offset the distortion cost induced by the flipping. This implies that for sequence A the isolated pixel will be flipped to zero, and for sequence B all one pixels will be flipped (first one pixel is flipped, then the next one pixel, i.e., there will be a cascade effect). In the sequence based approach the cost of coding the string is compared with the cost of coding another string. For this example, there are 2^{12} possible strings to compare with as the length of the string is 12. The string which has the lowest cost is selected. The difference between the outputs of the symbol based and sequence based approaches (Fig. 2.4) is that the symbol based approach does not recognize the difference between the isolated one (in A) and a run of ones (in B) while the sequence based approach, because it looks ahead, is able to do so. Clearly, there is an advantage in lookahead quantization.

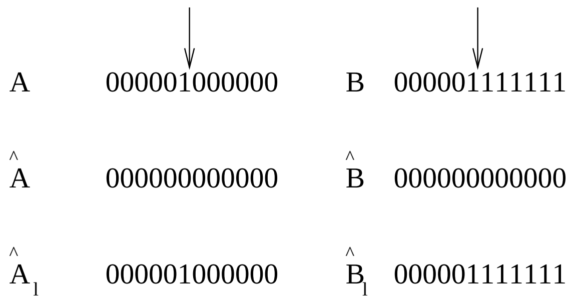


Figure 2.4: Two bi-level sequences A and B are considered. A 1st order context with fixed probability model $p(0|0) >> p(1|0)$ and $p(1|1) > p(0|1)$ is used for coding. The results for symbol based quantization (flipping for bi-level) $\hat{A}$ and $\hat{B}$ are shown alongside the results for a sequence based approach $\hat{A}_l$ and $\hat{B}_l$.

There are two issues that arise in a sequence optimization for adaptive context coding environments:

- (1) In case of adaptive entropy coding, the probability model is updated after each encoding operation, i.e., the probability model depends upon the entire history of operations. Clearly, this is an example of a non-finite state system where a state at stage i can be defined as,

$$s_i = (y_{i-1}....y_0) \tag{2.9}$$

- (2) Even if the memory of the state is finite, say M, the number of states, for the definition given in equation(2.1), will be N_q^M, if there are N_q possible quantization choices for each sample. This could be a very large number, for example in coding gray-level images, the quantized wavelet coefficients can take up to 2^5 different values [3], hence the number of states in this case would be 2^{15} for a memory of order 3. Even with the efficient Viterbi algorithm the cost of lookahead search would be prohibitive.

Both these issues lie within the general framework of the problem for which a PSP based solution has been proposed in section 2.2. Let us take a concrete example, in Fig. 2.5(a) a non-finite state system which models an adaptive entropy coder for bi-level sources is shown. For adaptive models the entire past history is needed, i.e., each path in the tree in has a different probability model. To reduce the non-finite state system to a finite state system, similar states have to be merged together. If $f()$ is defined as $f(y_{i-1}...y_0) = y_{i-1}$ then the similar state can be merged together to form the trellis in part (b) of the figure. For these merged-states the uncertainty is the probability model; for each path into the merged-state there is a different probability model and hence a different transition out of the merged-state. Given this definition of $f()$, $g()$ has to be a function of $y_{i-1}...y_0$, such that the probability models can be calculated for each state. Thus, if the parameter set is defined as the probability model associated with each state, then the transition rule is well defined. PSP could be used to estimate the probability model from the survivor path of the merged-state. Using this definition of merged-state and parameter set a trellis can be defined and modified VA run to find the best sequence. The update operation for each state, equation(2.6), is a single operation of incrementing the probability count with the encoded sample, $y_i^{ms_i^S}$, corresponding to the survivor path.

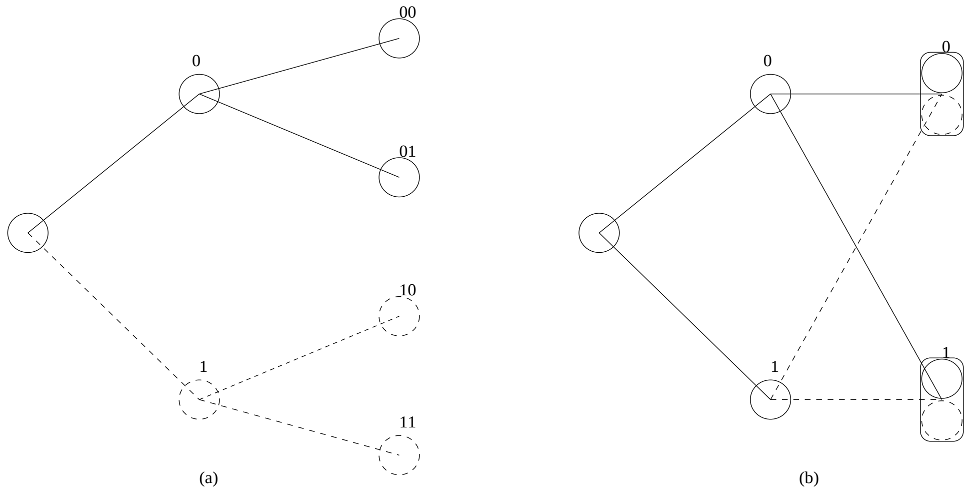


Figure 2.5: In a bi-level adaptive entropy coding, the entire history is needed to calculate the probability models. Hence, the tree shown in (a) is needed for lookahead, with each path having a corresponding history. The non-finite state system can be reduced to a finite state, let the similar states be (00, 10) and (11, 01). They can be merged to form merged-states 0 and 1 respectively. Its clear that the broken line into a merged-state has a different probability model than the solid line. To define the transition rule, i.e., the next encoding operation the probability model is needed, hence the parameter set of each merged-state carries a probability model which is based on the survivor path of the merged-state.

To give a general solution for all adaptive context based environments, and one which will be used in our experiments, we define the merged-state to be a context value, i.e, $ms_i(j) = z_j$, $j = 1 \ldots N_c$. The main reason for using this definition is is that the number of contexts N_c is usually small, hence the number of merged-states in the PSP based trellis is also going to be small. Also a context encoder while encoding a source moves from one context to the next, hence its operation can be modeled as a finite state machine, this will be reflected in the designed trellis.

Given this definition of the merged-state the parameter set can be easily defined. As discussed above for the adaptive entropy model for bi-level coding, the parameter set has to contain the probability models. This parameter set will suffice for the bi-level image coding, however for the gray-level case, y_{i-1} and y_{i-2} (or equivalently w_{i-1},w_{i-2}) are needed along with $c(y_i)$ (ms_i) and y_i to calculate $c(y_{i+1})$ (ms_{i+1}). Thus for the gray-level coding case, y_{i-1} and y_{i-2}, have to be stored in the parameter set and updated with the survivor path. In the next section we present results of our reduced state space search.

2.5 Experimental results

In our experiments we are minimizing $D(\hat{\mathbf{x}}, \mathbf{x})$ for a given rate budget $R(\mathbf{y})$. This can be reformulated into an unconstrained minimization problem using Lagrange multiplier λ, i.e., the goal becomes

$$min_{\mathbf{Y}}(\sum_{i=0}^{L} d(x_i, \hat{x}_i) - \lambda \sum_{i=0}^{n} log(p(y_i|c(y_i))))$$

If x_i is the input, y_i is the index associated with the branch $(s_{i-1}(j) \rightarrow s_i(k))$ and $c(s_{i-1}(j))$ is the context associated with state $s_{i-1}(j)$, then the branch metric can be calculated as:

$$BM(s_{i-1}(j) \rightarrow s_i(k)) = d(y_i, x_i) - \lambda log(p(y_i|c(s_{i-1}(j)))) \qquad (2.10)$$

We are using the adaptive 2D context model introduced in section 2.3 for our experiments. Each line of a 2D source can be considered as an independent 1D sequence by assuming that the previous line of the source has already been encoded, i.e. w_i, w_{i+1}, w_{i+2} are fixed (fig. 2.3). This simplifying assumption has been used in most image coding environments[2]. Better results can be obtained by using near optimal algorithms for the 2D digital least metric problem, like the iterative message-passing algorithm [42, 43], however the improvement comes at the cost of additional complexity.

2.5.1 Lossy Bi-level Coding

In a lossy bi-level image coding, pixels are flipped within the rate-distortion optimization framework. We start by comparing the PSP based sequence optimization algorithm with the two suboptimal sequence algorithms, the M-L algorithm [18] and the iterative algorithm [2] for lossy coding of a bi-level image. In the M-L algorithm, used for sequence optimization of non-finite state systems, the tree is grown with each stage keeping, at most, the best M states. At the L^{th} stage the first symbol is released, the tree is grown again and the second symbol is released at the next stage and the process is continued for each symbol.

Fig. 2.6 compares performance of a PSP based trellis and a M-L algorithm with varying M,L parameters. Note that the trellis is run over a row of length 740 samples, of the image. As M, L increase the performance of tree algorithm improves, but it is clear that to achieve performance comparable to the PSP, a very large L would be required, which would be impossible to implement because of the complexity.

An iterative algorithm has been proposed for adaptive context coding environment used in near-lossless compression of gray-level images [2]. The iteration scheme starts with an initial probability model, finds the optimum sequence in the trellis given the model, and updates the model with the optimum sequence. The procedure is repeated until the gain in entropy becomes negligible. In an earlier paper [44] we have shown, for this near-lossless coding scenario, that PSP based VA achieves the same performance as the iterative scheme but does not require any iteration. In this paper we compare the two schemes for lossy coding

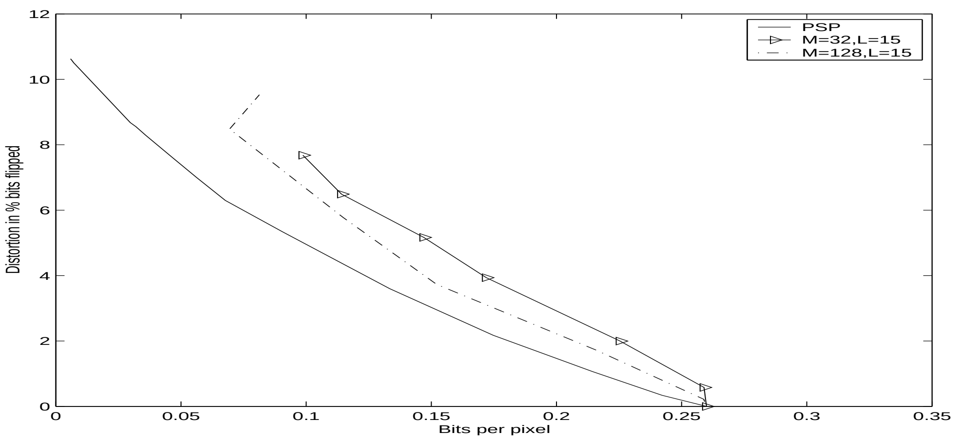


Figure 2.6: Comparison of the PSP based trellis algorithm to a tree algorithm where all the probability models are retained. ML algorithm is used to find the best path in the tree.

of bi-level images, Fig. 2.7. In Iterative 2 the maximum number of iterations is set to 8, in Iterative1 the stopping criteria is the change in bit-rate between the two iterations. If it is less than 0.01% then iterations are stopped. The figure shows that PSP performance is very similar to these iterative algorithms.

Bo Martins and Forchhamer [45] have implemented a greedy look ahead scheme for lossy context coding of binary images. We have shown that our scheme outperforms their scheme especially for 1-D context models in our ICIP paper [44].

2.5.2 Lossy Gray-level Image Coding

For gray-level image coding, we are using the encoder developed by Chrysafis and Ortega [3] and described in section 2.3. We have replaced the symbol based quantizer in their work by our sequence based quantizer based on the PSP trellis. The results shown in Fig. 2.8, show the improvement of sequence quantization over symbol quantization. X, Y, Z are results obtained by a symbol quantizer of different step size. Using the step size corresponding to X, we can lookahead and perform a search to find the path with the best $D + \lambda R$ for different λ. This will trace the PSNR-rate curve shown in Fig. (curve with left triangles). Point W the PSNR is higher than that obtained at Y using a symbol quantizer, showing the improvement in using a lookahead scheme.

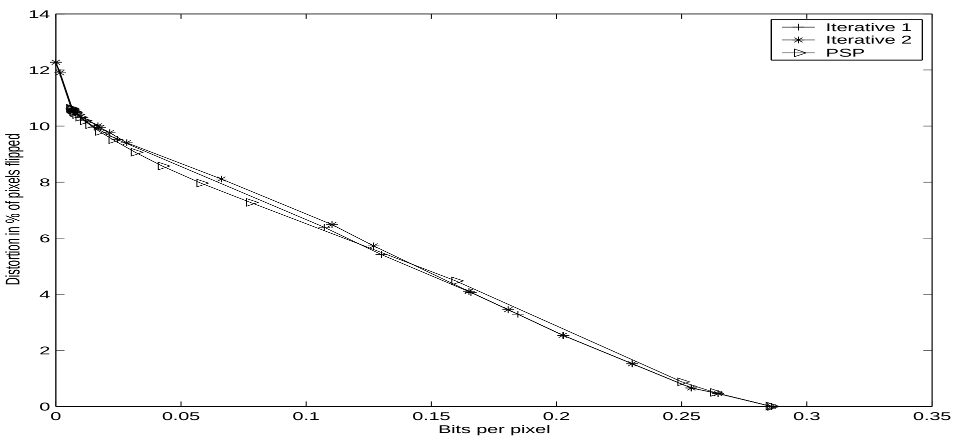


Figure 2.7: Results for bi-level coding. Image is 745x1024 b&w image from the CCITT database. PSP stands for the proposed PSP based modified Viterbi algorithm. Iterative schemes are based on the traditional approach used by [2]. Iterative1 the stopping criteria is change in rate between iterations should be less than 0.01% and Iterative2 8 iterations are performed each time.

Ramchandran and Vetterli [46] have implemented a lookahead scheme for JPEG which is similar to our work in wavelet coding of gray level images. However, their lookahead scheme is for 1D run length coding while here we use 2D context models.

2.5.3 Denoising of Gray-level images

For denoising we follow two separate methodologies. First, we us the principle of Occam's filter proposed by Natrajan [29] to provide joint compression and denoising. The idea is that random noise will be more difficult to compress than the signal (which has some structure). Hence a good compression method can provide a suitable model for distinguishing between signal and noise. In Fig. 2.9, we compare the denoising results of our sequence based encoder with the symbol based encoder of Chrysafis. The PSNR is calculated by using the mean square error between the original and the denoised image [47]. The results clearly show that sequence based encoders are more efficient than symbol based encoders and are better than the best reported results in [47].

In addition, we have also used the soft thresholding method proposed by Chang *et al.* [47]. They have derived soft thresholds, to threshold noisy data in wavelet

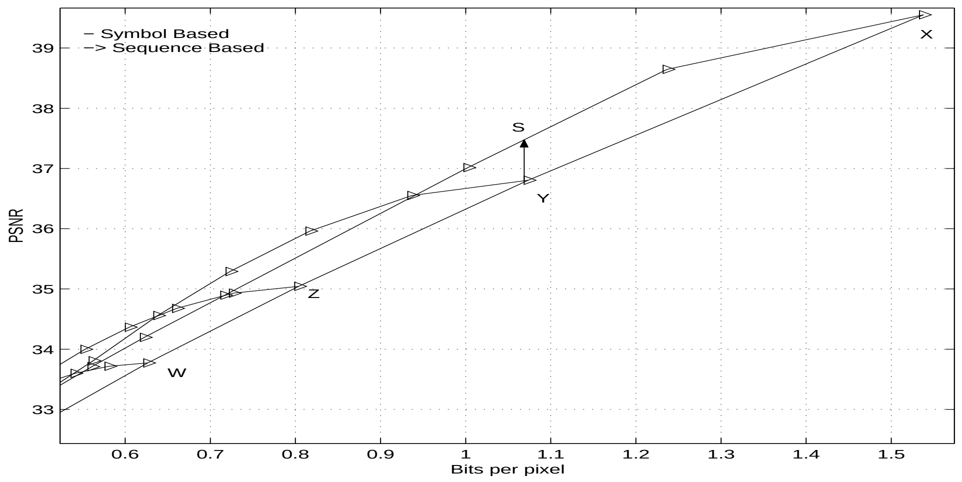


Figure 2.8: Results for gray-level coding. X,Y,Z are results for different quantization step size of the symbol based quantizer, i.e., no lookahead is used for these results. For each of the step sizes, the lookahead algorithm is run, and the sequence with the minimum $D + \lambda R$ cost is found. As λ is increased, the PSNR and bit rate slide down the curves shown in the figure (a different curve for a different step size). Thus we can start with the uniform quantizer step size 0.5 corresponding to result X, increase λ and slide down to point W. The PSNR for W is higher than the PSNR for Y but both are coded with the same bit rate.

domain, based on the estimated noise variance and signal variance. In a later paper [15], the same authors have extended their algorithm such that the threshold for each sample is calculated from local information by using the variance of pixels in the neighborhood. They use a two pass algorithm: the first pass gathers information about the source; the second pass calculates the thresholds and denoises the data. The authors have mentioned the disadvantage of using a single pass algorithm: a run of zero coefficients may cause all subsequent coefficients to be quantized to zero. They have not given their single pass result but have mentioned that the results were much worse the two pass results.

We have implemented a single pass lookahead soft thresholding scheme. The disadvantage of a single pass algorithm, can be avoided by using lookahead. The PSP based trellis is used, with the modification that the parameter set for each state contains the probability model *and* the threshold for denoising based on the past information. The operation at each state is to threshold or not to threshold and each operation has a corresponding path in the trellis. Using rate as the

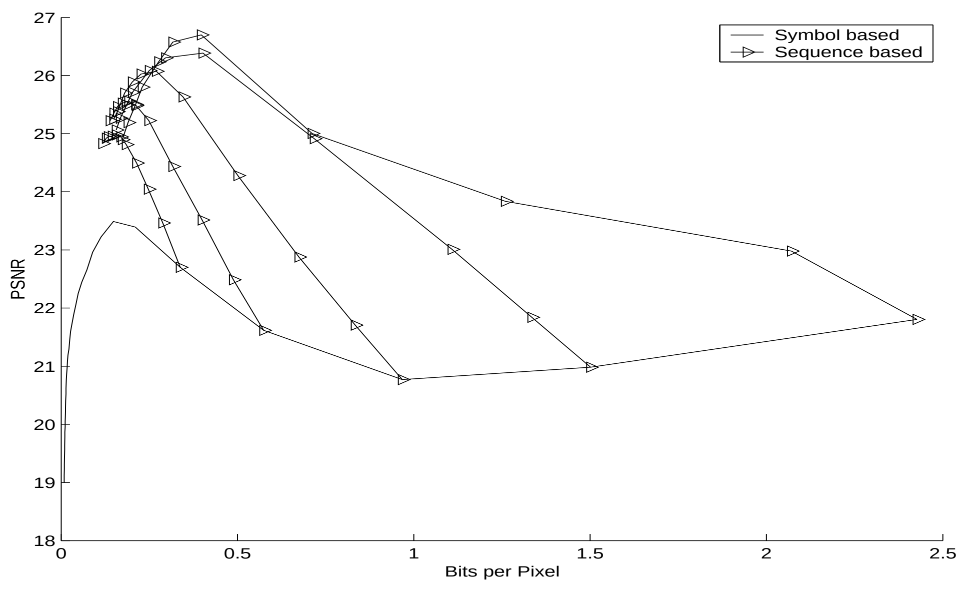


Figure 2.9: Results for Barbara image, with Gaussian noise of std=20. The Occam filter is used, i.e., a compression scheme is used to denoise image. The PSNR results are calculated by finding mean square error between original (clean) and denoised image. We are comparing our sequence based encoder with Chrysafis *et al* [3] symbol based encoder. Clearly the sequence based scheme does much better.

regularization metric [48] we find the path which has the lowest distortion for a given rate. In Fig. 2.10, we show the improvement in a single pass algorithm. A single pass without rate-regularization and lookahead would give about 24.5dB, by using lookahead and rate-regularization we can improve the performance to around 27.4 dB which is very close to performance of the two pass algorithm of Chang *et al.* (using the same causal context).

2.6 Conclusion

In this paper we have proposed a reduced complexity lookahead search algorithm for environments where there can be a state explosion, e.g., where the present encoding operation depends upon all the past operations. The solution is based on per-survivor processing principle. We have taken the example of context based entropy coding environment and shown that a reduced complexity lookahead search

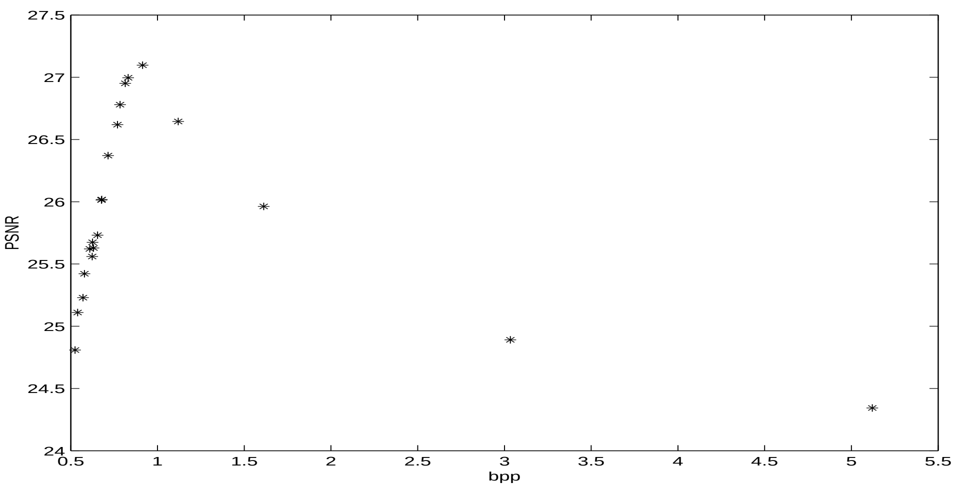


Figure 2.10: Barbara image, Gaussian random noise of std=20. Single pass lookahead algorithm, where soft thresholds are kept at each state. There are two choices, to threshold or not to threshold. The path which has the minimum distortion for a given a rate is chosen. Rate is the regularization metric, to calculate rate a uniform threshold quantizer is used. However, the distortion is calculated between the thresholded and the original samples.

can be performed to improve compression and denoising performance. Our algorithm could be easily extended usually to other state systems with large memory.

Chapter 3

Multiple Description Coding

3.1 Introduction

In the previous chapter we have presented algorithms for efficient compression of multimedia data. As mentioned in the introduction compressed data is more vulnerable to channel noise, hence efficient schemes for reliable transmission of compressed multimedia data are needed. Also it has been recognized that schemes that add redundancy, to protect the data, in a joint source channel coding manner are more efficient. One such scheme, Multiple description coding (MDC), was first proposed in the 1980s [23], and has gained practical significance over the past few years. Since the introduction of multiple description scalar quantizers by Vaishampayan [49], in 1993, many techniques have been developed for MDC, some of which are reviewed later in this chapter.

MDC, as discussed in the introduction, besides robustness also offers scalable streams. A description is independently decodable and each additional description improves the decoding quality. For networks such as the Internet that offer no guarantees on the quality of service, scalability of the application is a desirable feature [50, 35]. The application can adapt to the changing network conditions by scaling the bit-rate, e.g., if high bandwidth is available a high quality, i.e., a high bit-rate version of the data, can be made available. On the other hand, if very low bandwidth is available a coarse quality, low bit-rate, data will be available to the user.

Many methods are available for adjusting the quality/bit-rate of the data, e.g., multi-resolution (also called layered) coding, transcoding, and keeping several versions of the data at different qualities. Of these, layered coding is the most popular because of its low complexity and high compression performance. In layered coding there is an order in which the layers are encoded and if at the decoder a lower layer, i.e., one which was encoded first, is not available the higher layers are rendered useless. In Fig. 3.1, the Lenna image is encoded into two layers using the progressive JPEG coder [8]. Clearly if the base layer, the first layer to be encoded, is lost then the enhancement layer, the second layer to be encoded, will decode to a visually unacceptable quality. Increasing congestion on the Internet means that this scenario of losses in the base layer could happen often, at least as long as there is no infrastructure to give higher priority to the correct delivery of the base layer.

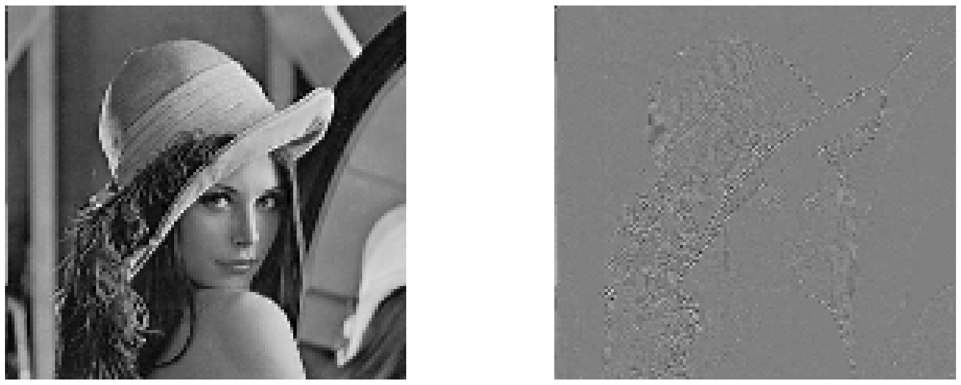


Figure 3.1: Lena image coded using Progressive JPEG coder. Base Layer mean square error (MSE) =59.06, Enhancement Layer MSE =3589.1, Total Bpp =1.8, Total MSE =8.8506, MSE is w.r.t. original Lena image.

In spite of this disadvantage, layered coding has become popular for realtime multimedia transmission over Internet. Receiver driven layered multicast (RLM) is an application developed by McCanne *et al.* [35] for video conferencing over multicast networks. Here layered coding is used to cope with receiver heterogeneity; the receiver individually adapts its reception rate by adjusting the number of layers that it receives. Another application is the recently proposed protocol for real-time streams over the Internet, rate adaptation protocol (RAP) [50], which is an end-to-end TCP friendly protocol, employing layered coding for quality adaptation at the sender.

To ensure reliable transmission of layered coding data, joint source channel coding schemes (JSCC) which recognize the priorities in layered coding and assigns higher protection to higher priority layers are often used [51]. We propose a JSCC scheme where the base layer is protected by using retransmission (ARQ) while the enhancement layers are sent without any protection. Given the time constraints associated with multimedia data, there is a finite time for transmission of a "frame". In our scheme, the base layer of the frame is transmitted first. If the base layer is received error free and there is time remaining for transmission of the frame, then enhancement layers are transmitted; thus a form of selective retransmission [34] is used.

In this chapter, we compare the performance of this JSCC scheme with a MDC scheme for transmission over lossy packet networks. We consider a point-to-point communication link with feedback. Parameters such as the Round trip time (RTT) of the link and the latency of the application are varied and performance of the proposed MDC and LC schemes are compared. As expected the performance of LC, where base layer is protected by ARQ, is more dependent upon the above parameters than MDC. On the other hand, MDC's performance is independent of these parameters and thus more suitable for heterogeneous environments. We have also compared a LC (without ARQ) based RLM scheme to a MDC based RLM, and established the viability of using MDC for multicast scenarios.

Reibman *et al.* [52] have also compared MDC with a JSCC scheme for a layered source encoder. In their JSCC scheme the the layers are unequally protected through FEC codes. In comparison with MDC, they have shown that their system is worse than MDC for erasure rate greater than $10^{-1.4}$ and for lower erasure rates the performance is very similar. In contrast to Reibman *et al.*'s work, we have used ARQ to protect the base layer because if a back channel is available, ARQ offers full reliability (given sufficient transmission delay), makes efficient use of the limited bandwidth and, unlike the FEC codes, is easily scalable to various channel conditions. ARQ is often ruled out for realtime data because of the delay associated with retransmission. However, this may not be true for all situations, especially if selective retransmission [34] techniques are used.

Section 3.2 presents an extensive review of the available techniques for MDC. In Section 3.3 we review the MDC scheme proposed by Jiang and Ortega [4] and extend it to the JPEG domain. In Section 3.4 the simulation architecture is presented and Section 3.5 gives the results of our simulations. Finally we conclude with comments on future work in Section 3.6.

3.2 Review of Multiple Description Coding

MDC is inherently suitable for realtime applications, it provides local recovery at the decoder which is preferable to Go-back-N type retransmission. In addition it provides graceful degradation unlike popular FEC codes. MDC is particularly useful for applications where physically separate channels are used, e.g., multi-channel, packet radio networks [53], or disparity routing for wired networks [54]. However, virtual channels having independent loss probabilities could also be created over a single physical channel by appropriate interleaving of the descriptions. The degree of interleaving would depend upon the loss properties of the underlying physical channel. Thus, MDC has been proposed as a viable means for robust transmission of realtime media over the Internet, with extensive research going on presently in this area [4, 49, 55, 56, 57]. The paper by Goyal [24] provides a good overview of MDC schemes.

The multiple description coding problem, first posed by Wolf *et al.* [58], for binary data, is that given the set of distortion (D_0, D_1, D_2) find the set of (R_1, R_2) necessary and sufficient to achieve these distortions (Fig. 3.2). It was subsequently solved for Gaussian data by Ozarow [59]. El Gamal and Cover [23] generalized the results for all kinds of data, characterizing achievable rates for multiple description in terms of the mutual information in the descriptions.

One of the first practical method for MDC, multiple description scalar quantizer (MDSQ), was developed in 1990 by Vaishampayan [49]. He designed quantizers for the side encoders such that the quantizer for the central decoder is a refinement of (either of) the side quantizers. The side and central decoders' performance can be traded off by adjusting the parameters of the side quantizer.

The pairwise correlating transform (PCT) was introduced by Wang *et al.* [60]

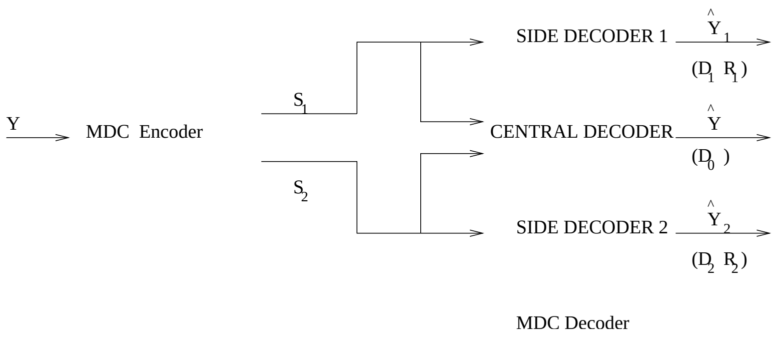


Figure 3.2: S_1 and S_2 are two descriptions generated of the source Y and transmitted over independent channels. If both are received, the decoder decodes to $\hat{Y}$ with central distortion D_0. If only one description is received, say S_1, the decoder decodes to $\hat{Y_1}$ with side distortion D_1. R_1 and R_2 is the rate associated with the respective description.

as a way of achieving MDC within the transform domain. In this scheme, MDC-PCT, the idea is to introduce correlation between a pair of source symbols by using an invertible correlating transform. The correlated symbols are sent over different channels. If both symbols are received, the inverse of the correlating transform is applied to recover the original signal. If only one symbol is received, it is used to estimate the correlated lost symbol and inverse correlation transform is applied to recover original symbols. The authors have developed an optimal transform based on rate redundancy distortion theory. Similar work was done by Goyal *et al.* [55], who extended the scheme to vectors of length greater than two.

Goyal *et al.* [61] have also used over-complete expansions for generating multiple descriptions, MDC-OC. The source, which belongs to the space R^k, is decomposed into an over-complete basis of dimensions $n > k$, thus generating n descriptions of a k dimensional source. If any $m > k$ descriptions are received, the signal can be recovered to its original quality. If less than k descriptions are received, estimators (linear or nonlinear [62]), can be used for approximately recovering the source, thus providing graceful degradation unlike a (n, k) channel code [24].

Mohr *et al.* [63] and Puri and Ramchandran [56] have independently introduced a novel scheme for MDC which utilizes an RS erasure code and overcomes the above mentioned shortcoming of FEC. They split an embedded stream across packets (descriptions) and provide unequal protection based on the importance level of the stream. If a minimum number of descriptions is received, the high priority symbols can be recovered. With each additional description more low priority symbols will be received (and/or) recovered at the decoder. This scheme is based on an unequal error protection scheme which was first developed in [33] and will be referred to as MDC-FEC.

3.3 Multiple Description Coding based on Polyphase Transform

A simple MDC scheme which gives excellent results is the polyphase transform and selective quantization scheme developed by Jiang and Ortega [4]. A block diagram for a two description system is shown in Fig. 3.3. The source is coded using two quantizers, e.g., a high resolution and a low resolution quantizer. The odd polyphase of the high-resolution (HR) coded source and the even polyphase of the low-resolution (LR) coded source are put together in one packet and sent as a description. The other packet contains the odd polyphase of HR and the even polyphase of LR. If all packets are received, the source vector is reconstructed to HR, else the reconstruction is to a quality that is the average between HR and LR.

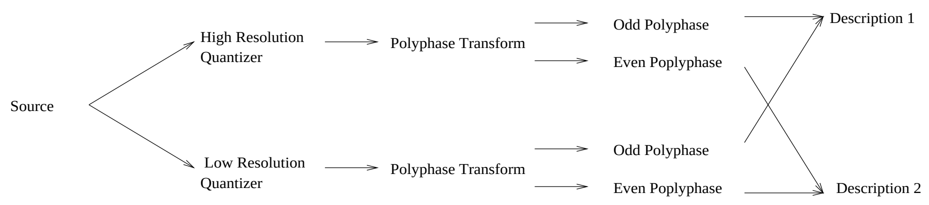


Figure 3.3: MDC based on the polyphase transform and selective quantization work of Jiang and Ortega [4]. This MDC technique will be used for simulations in this chapter.

The advantages of MDC-PT are, firstly it requires minimum modification to standard source encoders and decoders to generate or decode the descriptions.

Secondly it can easily adapt to changing channel conditions [64] by appropriately adjusting the quantizers' step size. Thus in comparison with the schemes mentioned in the previous section MDC-PT is simpler in implementation and yet gives comparable results [4].

A reliable layered coding scheme, which does not use retransmission, and is related to our MDC approach was proposed by Turletti *et al.* [65]. In their work a polyphase transform is used for producing independent layers for audio coding. Each layer contains one polyphase decomposition of the original sequence. If only one layer is subscribed to, then the other polyphase components are estimated by interpolation. Each additional layer improves the signal quality. The polyphase components are transmitted using the real time protocol (RTP), and robustness to packet loss is achieved by sending two polyphase components in the first layer. The scheme is similar to ours in that each polyphase component can be thought of as a description, with the main difference being that we incorporate redundant information. Also the emphasis of our work is the performance of a layered coding scheme when there are losses in the network. Turletti *et al.*, on other hand, use their scheme to establish the effectiveness of congestion control using a layered approach.

3.3.1 MDC-PT for JPEG coded images

Previous work on MDC-PT considered decorrelated synthetic data generated from sampling a probability distribution [4, 64]. In this thesis we have extended the MDC-PT scheme to transmission of images. Images are usually compressed before transmission and one popular compression scheme is the discrete cosine transform (DCT) based JPEG [8]. In this section we present an algorithm for generating multiple descriptions of an image, each of which is encoded using the standard JPEG algorithm.

The MDC-PT system in the JPEG domain is shown in Fig. 3.4. The image is DCT transformed and the transformed coefficients are quantized using two quantizers, high resolution (HR) and low resolution (LR). The quantizers differ in the step size, Qp_{hr}, Qp_{lr}, used in the standard JPEG quantizer. The quantized coefficients are ordered using the standard zig-zag indexing of JPEG [8]. These ordered coefficients are then polyphase transformed, e.g., in Fig. 3.4, the underlined coefficients are the odd polyphase components of a transformed block. The odd polyphase coefficients of HR and the even polyphase of LR form description 1 (D1). Description 2 (D2) contains even polyphase HR coefficients and odd polyphase LR coefficients. Each description is then entropy coded using the standard JPEG entropy coder. In practice we do not need to perform the polyphase transform, instead we make two copies of the transformed coefficients. In one copy we quantize the even (in the zig-zag order) coefficients with Qp_{hr} and the odd coefficients by Qp_{lr}. In the second copy, the odd coefficients are quantized by Qp_{hr} and the even coefficients by Qp_{lr}. These copy are entropy coded and thus we generate the same descriptions as above without the use of an explicit polyphase transform.

If both the descriptions are received correctly at the decoder, odd HR and even HR coefficients are collected from their respective descriptions and the resulting data is decoded to the high resolution quality. If only one description is received, then additional preprocessing is required before it can be decoded. For example if D1 is received a preprocessor would multiply, the entropy decoded, LR even coefficients by Qp_{hr}/Qp_{lr}. A standard decoder using the quantization step size Qp_{hr} will then decode the data.

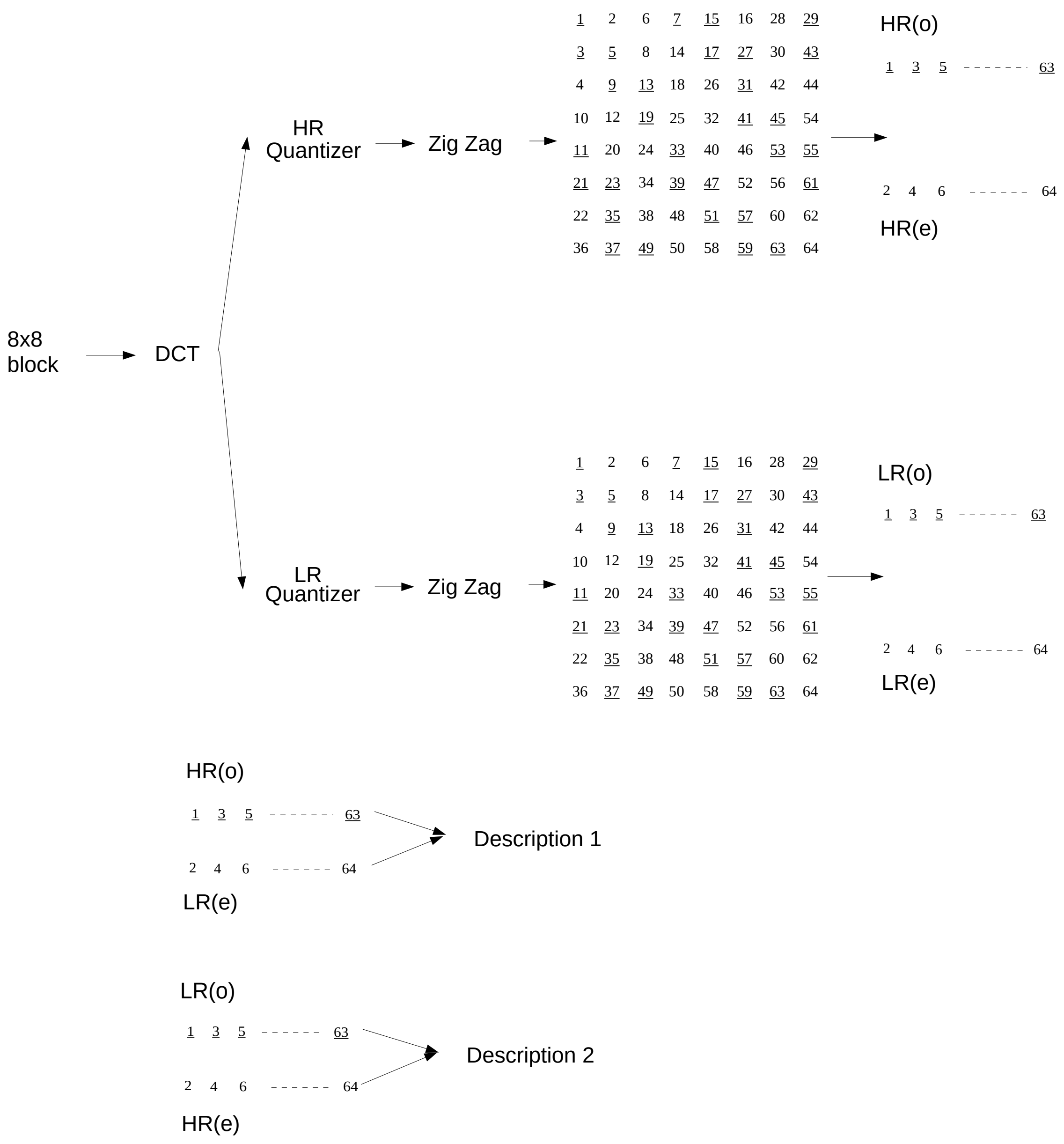


Figure 3.4: The proposed MDC system based on DCT transform coding. An 8x8 blocks of image data is DCT transformed. It is quantized by the high resolution (HR) and low resolution (LR) quantizers and zig-zag indexed for entropy coding. Polyphase transform along the zig-zag index separates the blocks into even and odd (underlined) coefficients for each of the quantizer blocks. The even coefficients of HR are packetized with odd coefficients of LR to form Description 1 (D1). The odd coefficients of HR are packetized with even coefficients of LR to form Description 2 (D2).

An example of the Lenna image coded with the MDC-PT scheme is shown in Fig. 3.5. The descriptions are generated in the DCT domain, thus ensuring that the polyphase transform does not remove the correlation of the source and hamper the compression performance of the JPEG coder.

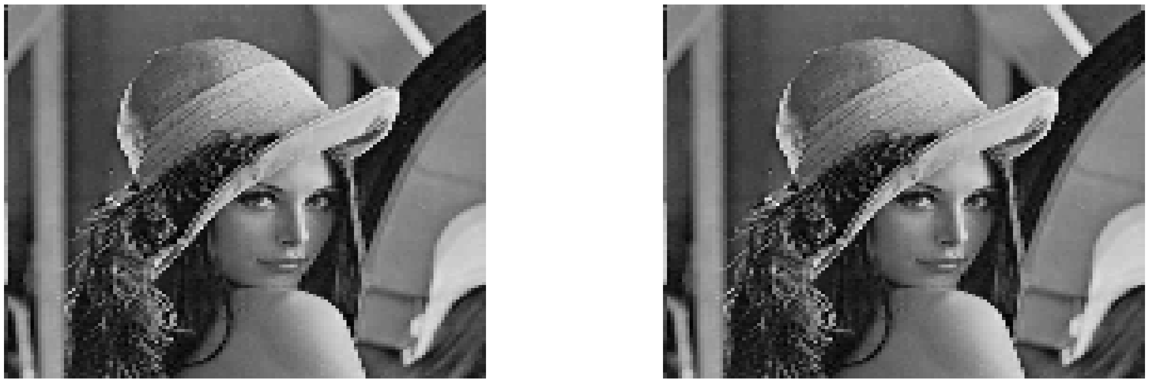


Figure 3.5: Lenna image coded using MDC-PT. Description 1 MSE = 22.16, Description 2 MSE= 22.01, Total Bpp= 1.79, HR MSE= 14.39, MSE is w.r.t. original image.

In practice we do not need to perform the polyphase transform, instead we make two copies of the transformed coefficients. In one copy we quantize the even (in the zig-zag order) coefficients with Qp_{hr} and the odd coefficients by Qp_{lr}. In the second copy, the odd coefficients are quantized by Qp_{hr} and the even coefficients by Qp_{lr}. These copies are then entropy coded and thus we generate the same descriptions as Fig 3.4 without the use of an explicit polyphase transform.

3.3.2 Generation of multiple layers/descriptions

For congestion control strategies as used in RLM or RAP, two layers/descriptions will not give sufficient granularity. In this section we propose three different schemes for generating higher number of descriptions in the JPEG environment. Also a comparison of the proposed schemes in terms of their coding efficiency, i.e., their compression performance, is presented below. For each of these schemes we are describing the method for generating four layers, however it can be easily extended to higher number of layers.

In Fig. 3.6 the MDC-PT scheme described in Fig. 3.4 is extended to multiple layers. An important feature of the JPEG entropy coder is the run-length of zeros, which it codes very efficiently reducing the bit-rate of the compressed image. However, because of the polyphase transform over individual coefficients, the run length of zeros is often broken in MDC-PT thus adversely affecting the compression performance. This problem can be alleviated by grouping the coefficients together, along the zig-zag index, and performing the polyphase transform over the set of groups. One such scheme, MDC-BPT, is shown in Fig. 3.7, where the coefficients are grouped together into four groups, G_1, G_2, G_3, G_4 and each group is quantized with the four different quantizers. In the first description coefficients in group G_1 are quantized by Qp_1, in G_2 with Qp_2, in G_3 with Qp_3 and in G_4 with Qp_4. While, in the second description coefficients in group G_1 are quantized by Qp_2, in G_2 by Qp_3, in G_3 by Qp_4, and G_4 by Qp_1. This cycle is repeated to generate the other descriptions.

An alternate scheme for reliable transmission of the image could be to encode N (N is the number of descriptions) copies of the image, each with a different quantizer: $Qp_1 ... Qp_N$. The multiple copies could then be transmitted over the channel. The disadvantage of this scheme is that each additional copy will not improve the decoding quality of the decoded image; the lower resolution copies will have to be discarded entirely if a higher resolution copy is available. The advantage of this scheme is that coding will be very efficient and hence the total bit-rate required will be lower than MDC-PT or MDC-BPT. In Fig. 3.8, MDC-PT and MDC-BPT are compared with the the multiple copies scheme in terms of coding efficiency. Coding efficiency is defined as the ratio of total bit-rate of sending the descriptions in MDC, to the total bit-rate of sending multiple copies. The figure shows show that MDC-PT is very inefficient when the redundancy is very low. MDC-BPT on the other hand performs very well for all levels of redundancy.

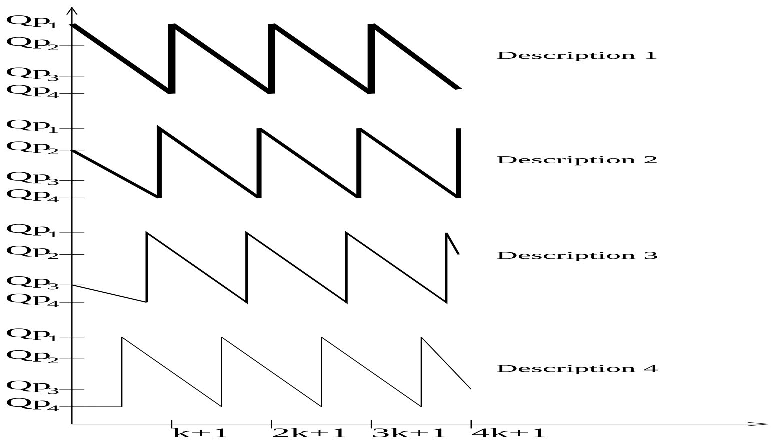


Figure 3.6: The x-axis represents the DCT frequency coefficients, along the zig zag index, in a 8x8 block. The y-axis represents the quantization parameter being used for a particular coefficient in a description. This is an extension of MDC-PT; in the first description the first coefficient is quantized with Qp_1, the second with Qp_2, so on and then the fifth coefficient is again quantized with Qp_1 (here k=1). For the second description, the first coefficient is quantized with Qp_2 and the cycle is repeated.

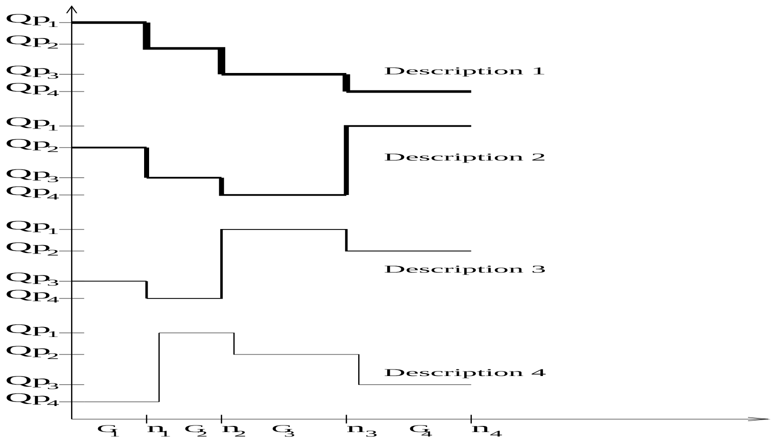


Figure 3.7: The x-axis represents the DCT frequency coefficients, along the zig-zag index, in a 8x8 block. The y-axis represents the quantization parameter being used for a particular coefficient in a description. This MDC scheme will be referred to as MDC-BPT, here the coefficients are grouped together $G_1 = \{C_0..C_{n_1}\}, G_2 = \{C_{n_1+1}..C_{n_2}\}, G_3 = \{C_{n_2+1}..C_{n_3}\}, G_4 = \{C_{n_3}....C_{63}\}$, where C_i is the ith transform coefficient. In the first description the first group is coded with Qp_1, the second with Qp_2 and so on. In the second description, the first group is quantized with Qp_2, the second group with Qp_3 and the cycle is repeated.

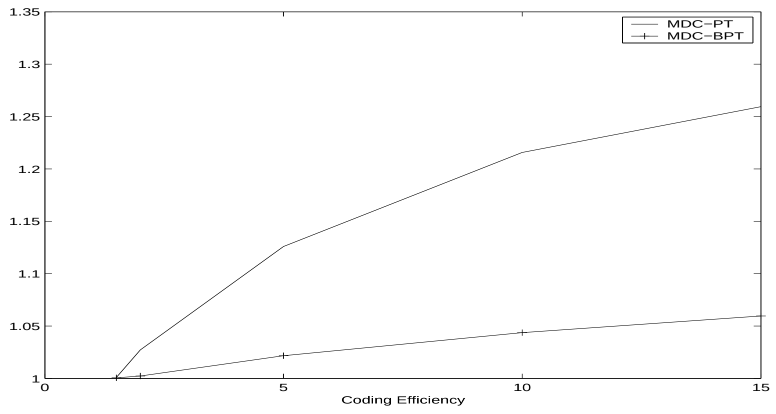


Figure 3.8: Coding efficiency is defined as the ratio of the total bit-rate for coding descriptions to the total bit-rate of sending two copies of the image. Each copy is quantized with a different quantizer,Qp_1,or, Qp_2, and encoded using the standard JPEG encoder. The results are shown for two descriptions, Qp_1 is the high resolution quantization parameter, while Qp_2 is the low resolution (redundancy) parameter. The results show the disadvantage of MDC-PT, the coding efficiency is very low.

An even more efficient method of generating multiple descriptions is shown in Fig. 3.9. This scheme, MDC-H, is a hybrid of LC and MDC. Layers are generated using a standard layered coder and the base layer is sent as the first description. For the second description, the low frequency coefficients in the base layer are coarsely quantized and added to the first enhancement layer. In the case shown in the figure, the second and third enhancement layer have no redundancy, but redundancy can be added by coarsely coding the low frequency coefficients. As each description is received the highest resolution frequency coefficients are extracted and decoded. In the next sections we compare these MDC schemes with a proposed LC scheme for transmission of realtime data over a lossy network.

3.4 Simulation Architecture

We simulate a real time video transmission over a packet switched network. In real-time streams, data has to be played back continuously at fixed time intervals, e.g., for jitter free television broadcast, 30 frames must be displayed in a second. This implies that there are strict timing constraints on the delivery of data, i.e.,

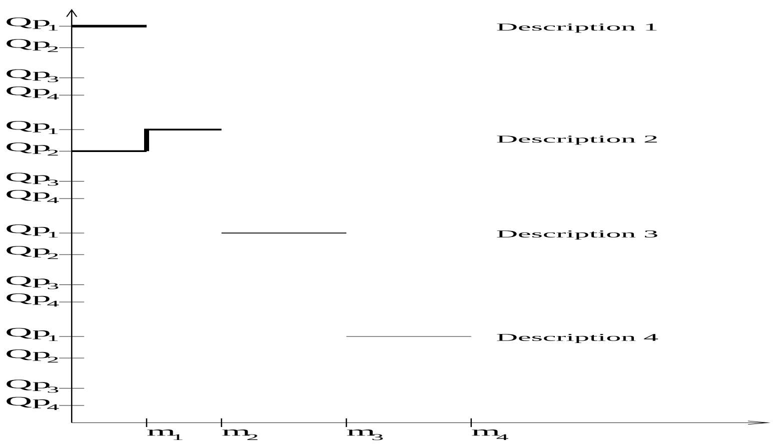
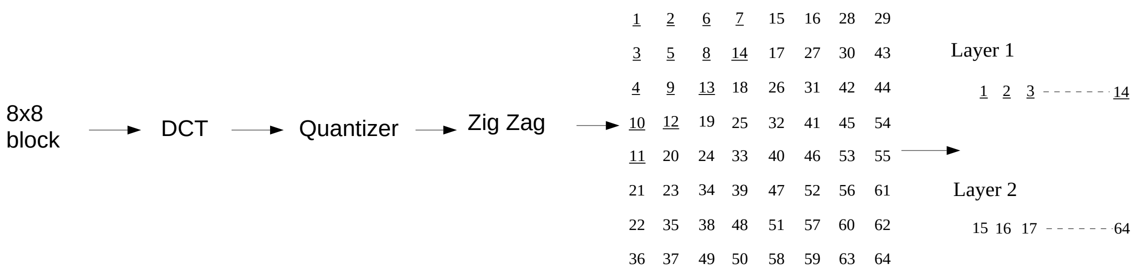


Figure 3.9: The x-axis represents the DCT frequency coefficients, along the zig-zag index, in a 8x8 block. The y-axis represents the quantization parameter being used for a particular coefficient in a description. This MDC scheme, (MDC-H), is a hybrid of layered coding and MDC; the image is split into layers, the base layer is sent as it is. For the first enhancement layer, redundancy is added by adding low frequency coefficients $\{0..m_1\}$ at the coarse resolution Qp_2. If more redundancy is needed the higher enhancement layers can also have coarsely quantized low frequency coefficients.

each frame should be received before its playback time. To simulate these constraints we define a timeout T at the receiver. This is the time that the data, of a frame, has for transmission. The assumption we are making is that there is a fixed time for transmission of a frame due to the fixed time for playback of the frame. This is not necessarily true, however it is a simplifying assumption and as both the LC and the MDC schemes are being simulated under the same conditions the comparisons are valid. T is dependent upon the receiver playback rate and the receiver buffer size and reflects the needs of the application (e.g., in a video conferencing application T will be smaller than for video playback from a server). In our simulations, we ensure, by choosing the channel bandwidth, that if there is no congestion all data of a frame is received in T time. Congestion is then added to the network using random background traffic and the performance of each scheme is measured as parameters such as RTT and T are varied.

Before getting into details of the simulation, we need to describe the layered coder used for our experiments. The layers are generated by using a progressive JPEG coder, shown in Fig.3.10. An example of the Lenna coded with LC has been shown in Fig. 3.1. The extension of the layered coding scheme to higher number of layers is shown in Fig. 3.11; all frequency coefficients in a block are quantized with the same quantizer and divided into layers.



1	2	6	7	15	16	28	29
3	5	8	14	17	27	30	43
4	9	13	18	26	31	42	44
10	12	19	25	32	41	45	54
11	20	24	33	40	46	53	55
21	23	34	39	47	52	56	61
22	35	38	48	51	57	60	62
36	37	49	50	58	59	63	64

Figure 3.10: Proposed layered coding scheme based on Progressive JPEG. The underlined coefficients are the low-frequency coefficients of the base layer.

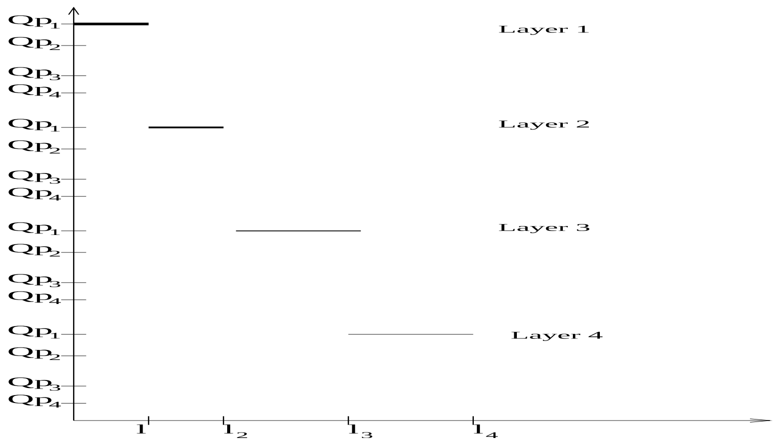


Figure 3.11: The x-axis represents the DCT frequency coefficients, along the zig zag index, in a 8x8 block. The y-axis represents the quantization parameter being used for a particular coefficient in a layer. The frequency coefficients are coded with the same quantization parameter and divided into layers. No redundancy is added to the layered coder.

3.4.1 Network Topology

We simulate the network by using *ns* [66], with the topology shown in Fig. 3.12. Transmitter T_0 and receiver R_0 are used to send and receive data, while the other transmitters and receivers are used for background traffic. Ten sources are initialized to generate background traffic in our simulations. Eight of these sources are TCP sources, five of which carry a variable amount of data, while three carry a fixed amount of data. Two sources are CBR, all the sources start and stop randomly and the side delay for these sources are also randomly generated. In the table 3.1, the parameters for the bottleneck link and the background sources are listed.

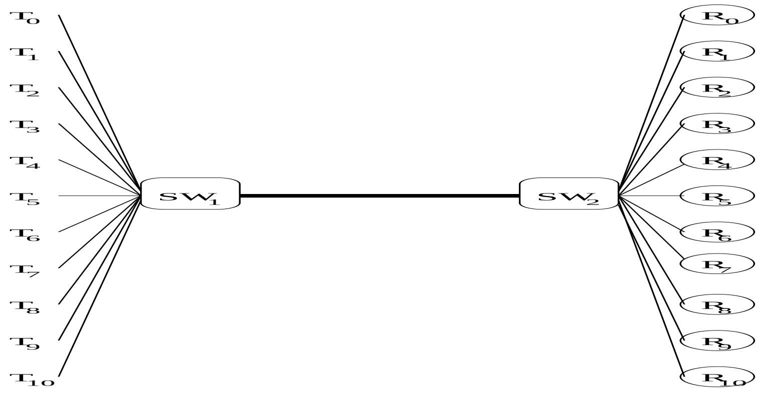


Figure 3.12: Network Topology, T_i are the transmitter and R_i are the receivers. The link between switches $SW1$ and $SW2$ is the bottleneck link and $SW1$ is the bottleneck point.

Parameter	Value
Bottleneck Link Delay	20ms
Side Bandwidth	100 Mbps
Bottleneck B/W	5 Mbps
Bottleneck Queue	50 packets
Background Packet Size	500 bytes
Background CBR Rate	0.6 Mbps
Background TCP Window Size	40 packets

Table 3.1: Simulation Parameters

Each description is divided into a fixed number of packets and sent over the network shown in Fig. 3.12. The packets are interleaved so as to lower the probability of losing both the descriptions in case losses are correlated. Similarly the base layer and the enhancement later are sent in different packets, if a base layer packet is lost, the low frequency coefficients are assumed to be zero and the high frequency coefficients are decoded from the enhancement layer packet. If both packets are lost, all coefficients are assumed to be zero; this is also true of the MDC case.

3.5 Simulation Results

Each experiment has been performed 200 times and the results reported are averaged over all the iterations. We have not used a real video sequence to generate the data; instead we have used a set of images, (available in the SIPI database), each coded with JPEG. Thus each image is considered as a frame and it is coded and packetized separately. Packets of an image have T seconds to be transmitted to the receiver, if packets of a image are received after its timeout they are discarded.

We have considered an image sequence because the problem of generating good descriptions of a motion compensated video sequence is still unsolved, with extensive work going on in this area in recent years [67]. Also motion compensation though giving a higher compression performance can have a severe degradation in quality, in presence of channel noise, due to the feedback loop in the encoder/decoder. The extension of MDC-PT to MPEG [9] coding of video is proposed as part of future work.

3.5.1 Unicast transmission using UDP

User Datagram Protocol (UDP) [68] is a transport layer protocol which offers no guarantee on quality of service. The packets are sent over the network and if they are lost the protocol does not attempt to recover them. The parameters that can be controlled are the packet size and the packet transmission rate. In Fig. 3.13, we compare the various MDC schemes with the LC scheme when the layers/descriptions are transmitted using UDP. The total bit-rate of all the layers (or descriptions) is 0.8 bps, with each layer being coded at 0.2 bps. The redundancy is set to 1/3 of the total bit-rate in MDC-PT and MDC-BPT and to 1/4 of the total bit-rate in MDC-H. A packet size of 100 bytes is used and the transmission rate is set such that all the packets of a frame would arrive in timeout T, if there were no losses. T is set to 0.1 sec for this experiment. MDC-H shows the best performance over a wide range of packet loss.

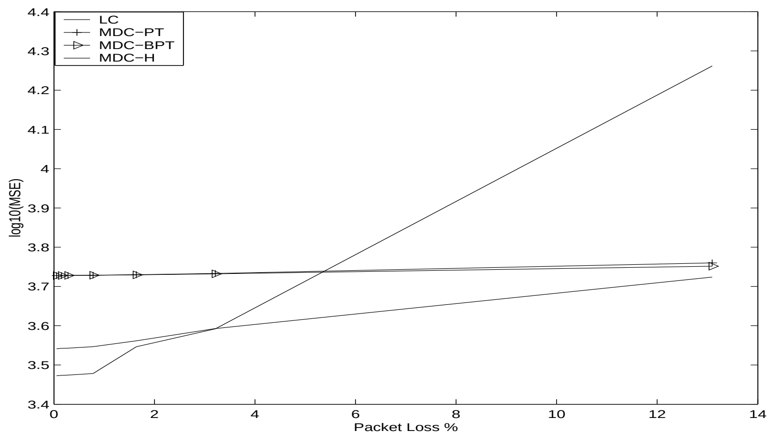


Figure 3.13: 4 layers/descriptions are transmitted using UDP. The total bit-rate is 0.8 bps, which is equally divided between the layers/descriptions. Redundancy for MDC-PT and MDC-BPT is 1/3 of the total bit-rate and for MDC-H is 1/4 of the total bit-rate. 64 packets of 100 bytes are transmitted for each layer/description. LC's performance degrades sharply with increased packet loss rate, MDC-PT and MDC-BPT have a constant performance, while MDC-H scales well between low packet loss rate and high packet loss rate.

In Fig. 3.14, MDC-H and LC are compared under the condition that the bit-rate of each additional layer is twice the bit-rate of previous layer. Comparing with Fig. 3.13, we see that MDC-H performs better than LC at a lower packet loss rate.

From these experiments it is clear that MDC will outperform LC when there are losses on the network, especially if base layer packets are lost. The need for reliable transmission of the base layer (BL) is immediately obvious.

3.5.2 Unicast transmission using TCP and UDP

TCP [13] is a transport layer protocol that offers a guarantee on quality of service, given sufficient transmission time. It uses retransmission to ensure that all packets are received at the receiver. However, this reliability comes at the cost of a variable

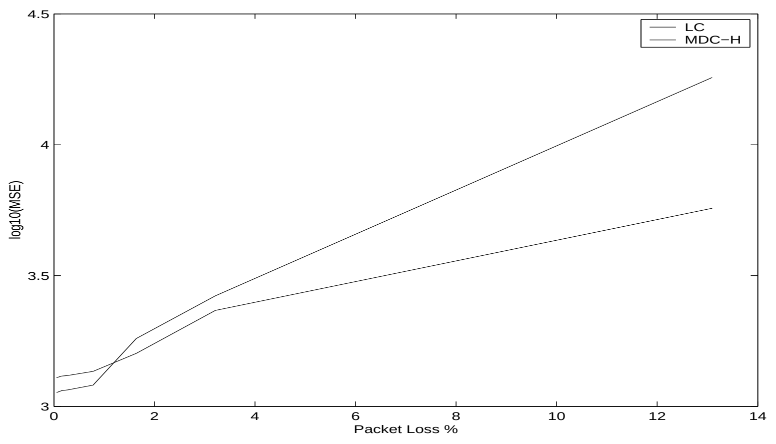


Figure 3.14: 4 layers/descriptions are sent over network using UDP. As the packet loss rate increases, LC has a sharp degradation in performance while MDC-H performance is quite constant. Here each additional layer is coded at twice the bit-rate of the previous layer.

transmission time which may or may not be suitable for an application. In this experiment, we send the base layer reliably using TCP and explore the effect of the RTT of the link and the latency of the application on the performance of MDC and LC.

In our experiment BL is transmitted first, if packets are lost in BL, the sender retransmits them. The sender has until $T_{start} + T$ to get the BL across. If BL gets across before $T_{start} + T$, the remaining time is used to send enhancement layer (EL) using UDP. Using the same background sources (we record the start/stop times of the background sources used in LC simulation) we also send MDC using UDP. At end of the T sec duration, we play back both LC and MDC with the packets that have been received and compare the MSE of the reconstructed image. In the experiment in this subsection, MDC and LC are coded at 0.8 bps with 1/3 redundancy for MDC. MDC-PT is used as the scheme for generating the multiple descriptions.

Given WindowSize and the BufferSize parameters in TCP, we have set a side delay for our network, such that if there is no congestion all packets of an image would arrive in 1sec. Then we add congestion and vary T. The results are shown in Fig 3.15. The results show that ARQ needs an additional latency of 0.6 sec. to achieve the same performance as MDC. if the application could afford this latency, the JSCC scheme of LC with ARQ would be a better alternative.

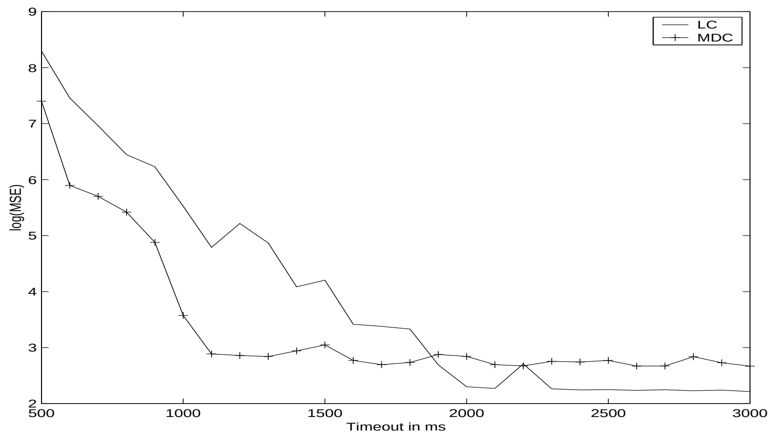


Figure 3.15: BL sent using TCP, after all BL packets received, if time is remaining from timeout T, EL is sent using UDP. MDC is sent using UDP with the same network conditions. The RTT used is such that all packets are received in 1sec. if there is no congestion. After adding congestion, the results show that MDC can achieve good performance close to 1sec. but LC requires 1.6 sec to achieve similar performance.

In the experiment above, given the bottleneck link and TCP parameters, the RTT of the network was set to ensure that if there was no congestion, then a frame (both layers) can be received in 1 sec. In realistic scenarios RTT will vary over a wide range, in our next experiment we vary the RTT of the network by varying the delay between T_0 R_0 in Fig. 3.12. The timeout is set to 1 sec. and if there is no congestion, then if for the RTTs that have been used for the experiment, both MDC and LC can get all packets of a frame across in the timeout. Congestion is added and the results in Fig. 3.16, show that for longer RTT, LC-ARQ would do worse than MDC. This experiment establishes that MDC can operate over a wide range of network conditions.

3.5.3 Multicast transmission using RLM

In the above section we have considered unicast transmission, in this section we are transmitting over a multicast network using the RLM protocol. The topology of the RLM network is shown in figure 3.17. In RLM the transport layer protocol used is UDP. A receiver subscribes to layers (descriptions) one at a time. If

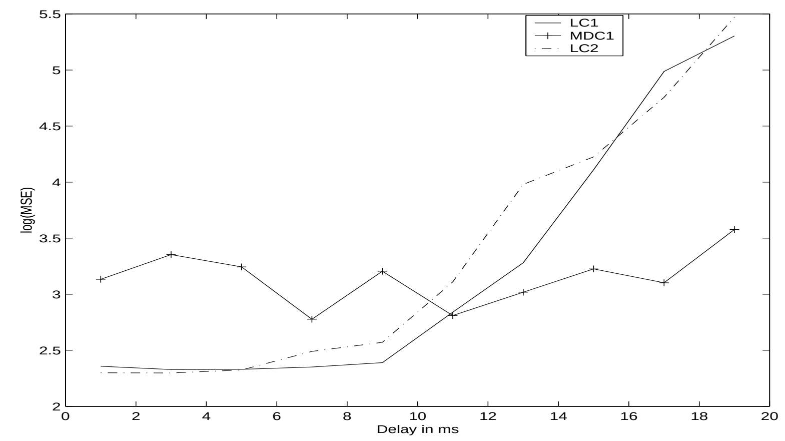


Figure 3.16: RTT between sender and receiver is varied. LC1, MDC1: the total bps is 0.8 and packet size is 500 bytes for both the packets. In LC2 the packet size for base layer is 600 bytes and for enhancement layer is 400 bytes. If there was no congestion LC would perform better than MDC for all RTT, if there is congestion than MDC performs better for long RTTs.

a receiver experiences losses it backs off by unsubscribing to the highest layer. Hence, it is form of congestion control where the receivers choose the best possible data rate.

In table 3.2 results for RLM are shown, the number are the $\log_{10}(MSE)$. At Node 6 the bandwidth to the end user is low, and every time the user subscribes to an additional layer, there are losses. In LC these losses could imply that parts of base layer are lost and this is the reason why MDC performs better than LC for all scenarios.

For Node 4 the bandwidth is capable of supporting multiple layers, however when loss is added to the link (node 1 to node 3), by the time Node 4 backs off there are losses in the base layer. If the latency of back off is high there will be more losses in the base layer, hence performance of Node 4 varies with the delay of the link 4 under these lossy conditions. Node 2 has medium bandwidth and no losses, hence MDC and LC performance are comparable.

	Node 2		Node 4		Node 6	
	LC	MDC	LC	MDC	LC	MDC
No Loss, D=200ms	2.1635	2.1821	2.129	2.1602	2.2951	2.2420
No Loss, D=2000ms	2.1635	2.1821	2.2843	2.3143	2.2993	2.2460
Loss, D=200ms	2.1707	2.1811	2.518	2.5107	2.5313	2.5107
Loss, D=2000ms	2.1707	2.1638	2.8913	2.5781	2.5325	2.5155

Table 3.2: Results for RLM

3.6 Discussion of Results and Conclusion

Table 3.3 lists the scenarios under which MDC does better than a LC with ARQ. The first unicast experiment established that if layered coding is sent without protection, its performance will deteriorate rapidly in case of packet loss, especially if base layer packets are lost. We protected layered coding with ARQ and showed that if the latency requirement was very low or the RTT was very long, MDC would be better than ARQ.

Parameter	Winner
No Protection for Layered Coding	MDC
Unequal Error Protection	MDC
Long RTT	MDC
Short Latency	MDC

Table 3.3: Summarization of Results

In case of multicast, retransmission is expensive. We have compared the case of RLM using LC with RLM using MDC. Our results show that MDC performance is comparable to LC or better in all scenarios.

The ARQ results are to be expected, retransmission is expensive if the RTT of network is long and also if the application has short latency requirements. There are two drawback of our current simulations, (a) a video encoder has not been used. (b) The base layer uses TCP as the retransmission protocol. More suitable transmission protocols like those in [34] and [69] will give better results for the ARQ scheme. This is left as part of future research.

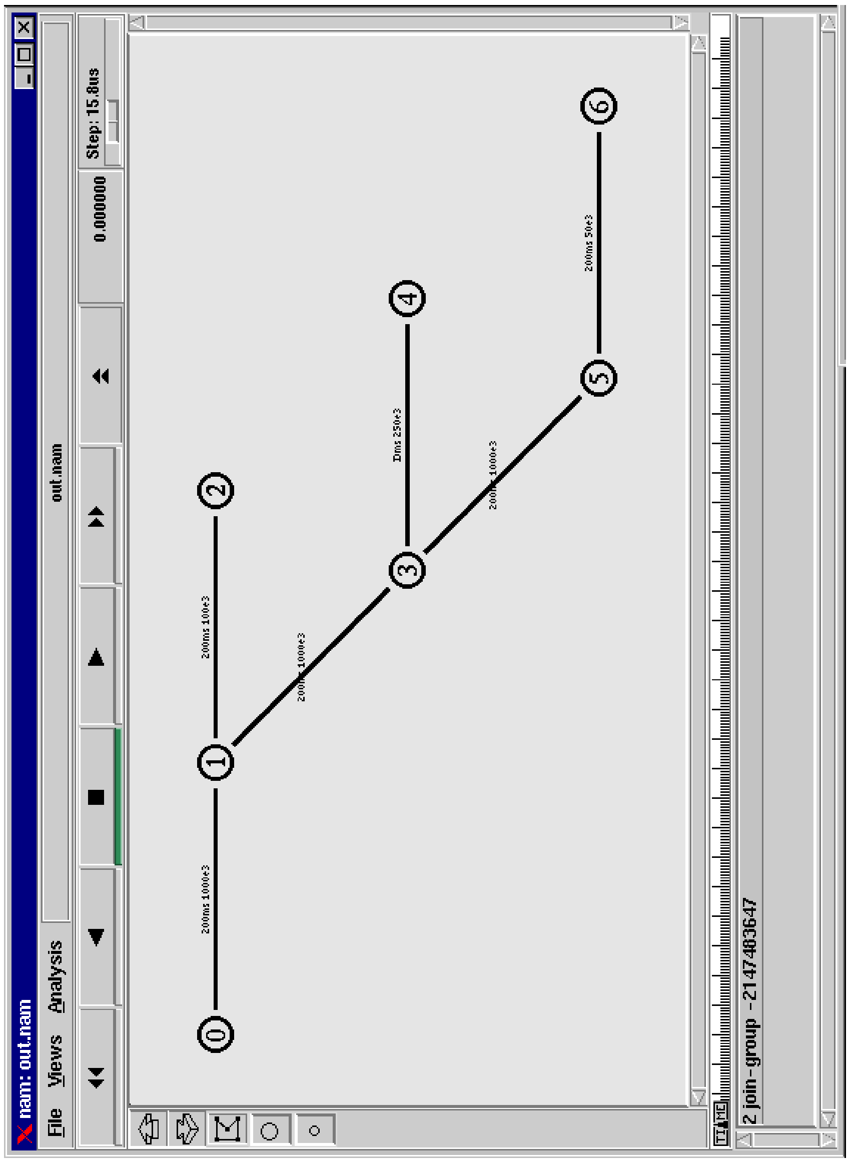


Figure 3.17: Topology of RLM, the bandwidth and the delay for each link is listed. The delay *D* of link node 3 to node 5 is varied, loss is added to link node 1 to node 3 by using a Loss Monitor. Queue size is 35.

Chapter 4

A Robust DPCM scheme based on Multiple Description Coding

4.1 Introduction

In the previous chapter we have discussed the performance of Multiple Description Coding (MDC) for transmission of images (or a sequence of frames) over a best effort packet networks, such as the Internet. In this chapter we focus on transmission of multimedia data compressed using a predictive encoder. Predictive encoders, e.g., differential pulse coded modulation (DPCM) in audio coding, motion prediction in video coding, take advantage of correlations in the source to achieve better performance than other approaches, such as pulse coded modulation (PCM) [14]. However, the main drawback of the predictive schemes is that a single erasure causes decoding errors to propagate through all the samples following the erasure. In contrast, PCM schemes treat the source as a set of independent samples and are thus are more robust, as losses do not propagate.

To make matters worse, most practical encoders tend to use entropy codes to represent the quantized data. Since the entropy coders use variable length codes (VLC), single bit error can lead to desynchronization of the decoder and cause catastrophic error propagation. In the case of entropy coded DPCM, an error in the bitstream can cause desynchronization in both the entropy decoder and the DPCM decoder.

A common approach to prevent error propagation in DPCM coders it to restart the prediction loop by periodically inserting PCM coded samples. One drawback of this approach is that though it limits the length of error propagation, there is a loss in coding efficiency due to the frequent restarting of the loop. Moreover, in this scheme it is difficult to recover lost samples. A similar approach is followed for VLC coders where self-synchronizing markers [70] are added to allow the encoder and decoder to resynchronize periodically.

In this chapter, we propose a novel robust entropy coded DPCM scheme, based on MDC, Fig. 4.1. For a detailed description of MDC refer to Chapter 2. The significant contribution of our work is the development of a sequence based

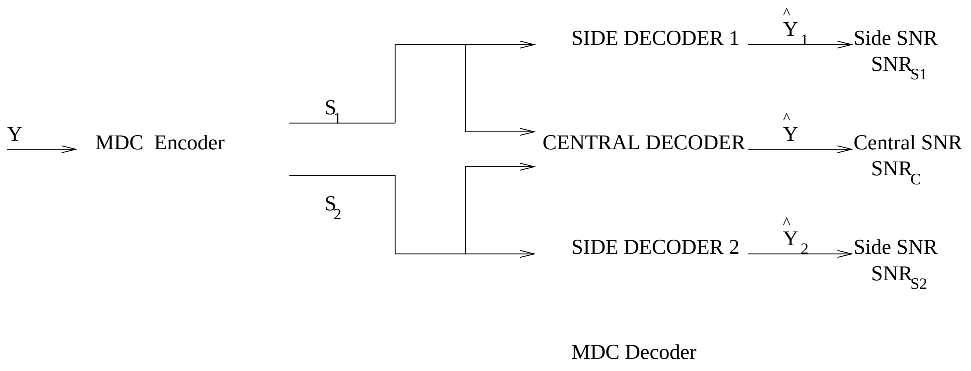


Figure 4.1: Multiple Description coding and decoding. The source Y is coded in two descriptions, S_1 and S_2 and each description is transmitted over an independent channel. If both descriptions are received, then the central decoder decodes to $\hat{Y}$ with SNR_c, else if one description is received, the corresponding side decoders decode to $\hat{Y}_1$ or $\hat{Y}_2$ with corresponding SNR_{s_1} or SNR_{s_2}. $SNR_c \geq SNR_{s_1}$ with $SNR_c \geq SNR_{s_2}$.

decoder to estimate data lost due to channel noise in a description. An example of the environment we are considering is shown in Fig. 4.2. If there was error propagation across packets, either due to the DPCM coder or the VLC coder or both, packet P_{23} will also have to be discarded [5].

Clearly the descriptions are correlated and the lost samples can be estimated from corresponding samples of the other description. In addition, we propose that these estimates have to be *consistent* with the received data. In simple terms the estimates are consistent, if and only if, when source decoded the output lies within

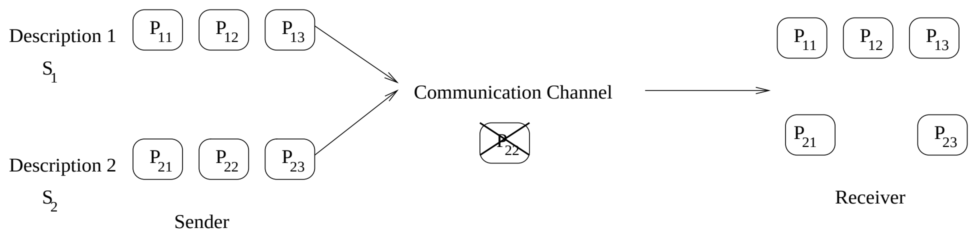


Figure 4.2: Two descriptions are sent over the channel and packet P_{22} of description 2 is lost. If the packets are not independent, the error propagates, the decoder discards packet P_{21} and P_{23} and decodes only description 1.

some bounds; these bounds are set up by the second description which is coding the same source though with a different quantizer. Also, for the predictive coding scenario consistency needs to be checked in a lookahead manner. As discussed in the introduction of this thesis, in memory based systems decisions that are taken at the present instant are going to affect the future. This implies that a estimate which is consistent at the present instant may lead to inconsistency in the future samples. An example is shown in Fig. 4.3. Thus our algorithm is based on maximum likelihood estimation of the erased samples P_{22}, where likelihood is defined in terms of a distance measure between the estimated $\hat{Y}_1$ and $\hat{Y}_2$ with the added constraint that the estimated P_{22} samples be consistent with all the error-free data that has been received, i.e., description 2, P_{21} and P_{23}. Our estimation algorithm exploits both the redundancy between descriptions and the encoder memory. As consistency is a key part of our algorithm we will call it consistent sequence estimation algorithm (CSE). This idea of consistent estimation is taken from **??** though have applied it in the totally different field of oversampled A/D converters. Note that we have taken the example of a packet network but our algorithm applies to all erasure channels. Vaishampayan and John have developed a MDC based robust DPCM scheme (BMDC-VJ) [5] using the multiple description scalar quantizers [49]. In their scheme their assumption is that if there are erasures (single/multiple) in a description, the description is discarded and the decoder decodes to the side SNR. Using our CSE algorithm we can estimate the lost data and thus decode close to the central SNR, this implies an improvement in performance of about 10-12dB over BMDC-VJ. In addition we have developed our own MDC scheme for DPCM coding. Two quantizers with

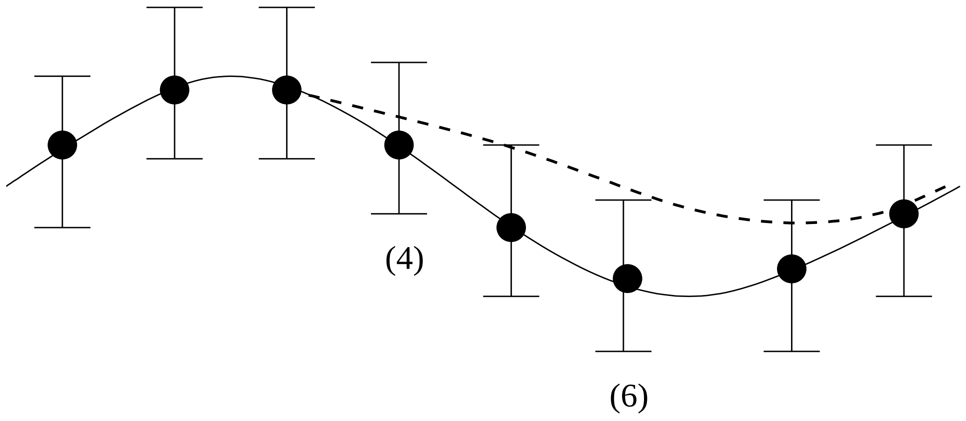


Figure 4.3: In the figure the dots are reconstructed samples of a DPCM coded signal. The vertical lines represents the bounds within which the output samples have to lie. The erased sample, (4) is estimated, however due to error in the estimation, which propagates, the estimation becomes inconsistent, i.e., violates the bound, at sample (6). This motivates the need for lookahead in consistency check for memory based encoders/decoders.

different step sizes are used in the side DPCM loops. Unlike those proposed in [5], the quantizers in our scheme do not require any special indexing. Thus standard quantizers like the Lloyd Max quantizers (for fixed length coding) and uniform threshold quantizer (for entropy coding) can be used. An important advantage of our scheme is that both balanced and unbalanced descriptions can be generated by simply changing the relative step size of the quantizers. A simple central decoding algorithm ensures that the central SNR is higher than the maximum side SNR for both the balanced and unbalanced schemes. Moreover, we propose a novel optimization algorithm which given the quantizers, can be used to generate descriptions where we can tradeoff the central and side SNR.

Forward error correction (FEC) codes could also be used for local recovery at the decoder. The main disadvantage of FEC codes is that they experience the *cliff effect* [24]; the performance is constant for up to some e erasures and then drops very sharply for more than e losses. Our scheme, on the other hand, provides *graceful degradation*; as the length of erasure grows, the performance of the CSE degrades gracefully.

The chapter starts with some details on DPCM coding which are relevant to our work. Section 4.3 presents a review of robust coding schemes, in particular

MDC based DPCM and VLC scheme and Section 4.4 gives the overview of our system. Section 4.5 gives the encoding algorithm for the central encoder and in Section 4.6 we introduce the erasure estimation algorithm. The results in Section 4.7 compare various schemes for MDC and show the advantage of using our estimation algorithm. We conclude with some directions for future work in Section 4.8.

4.2 Differential Pulse Coded Modulation

A typical DPCM encoder and decoder, Fig. 4.4, is an important example of a predictive coding system. Let $Y(i)$ be the original signal, $\hat{Y}(i)$ be the reconstructed signal, and $\tilde{Y}(i) = f(\hat{Y}(i-1), \hat{Y}(i-2)...)$ be a prediction of $Y(i)$ based on the past reconstructed samples. The DPCM equations can be written as

$$X(i) = Y(i) - \tilde{Y}(i) \tag{4.1}$$

$$\hat{X}(i) = X(i) + q(i) \tag{4.2}$$

$$\hat{Y}(i) = \hat{X}(i) + \tilde{Y}(i), \tag{4.3}$$

where $X(i)$ is the prediction error, $\hat{X}(i)$ the corresponding quantized value and $q(i)$ the quantization error. Equations (4.1) and (4.2) lead to the interpretation that DPCM is a generalized quantizer for which the zero, or center-point, keeps getting shifted to the latest value of the predictor, $\tilde{Y}(i)$. This shifting aligns the quantizer with the amplitude range most likely to be occupied by Y and enables the encoder to use a finer quantizer (than in PCM) for a given number of quantization levels. This interpretation will be used in deriving and explaining the CSE algorithm. See Jayant and Noll [14] for more details on DPCM systems.

In this chapter we consider a first order linear predictor, i.e.,

$$\tilde{Y}(i) = \alpha \hat{Y}(i-1),$$

where $\alpha < 1$ is the predictor coefficient. Our approach can be easily generalized to higher order predictors.

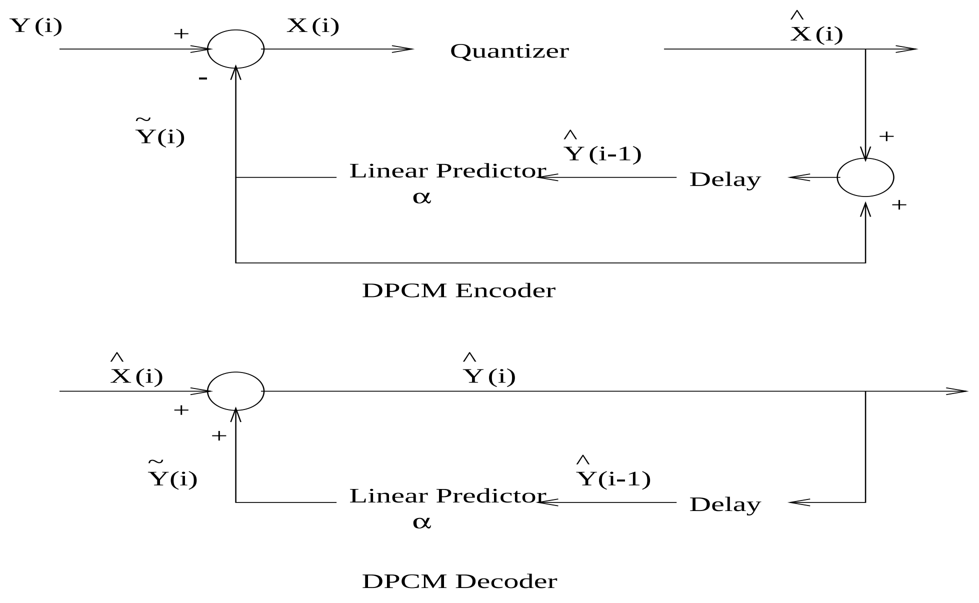


Figure 4.4: A first order DPCM encoder and decoder. Y is the source, $\hat{X}$ is the quantized prediction error sent over the network, $\hat{Y}$ is the reconstructed output.

4.3 Review: Robust DPCM schemes

A significant amount of work has been done on erasure recovery algorithms for predictive coding schemes; an excellent review can be found in [51]. In MDC based robust schemes, explicit redundancy is sent in the form of the multiple independent descriptions. Vaishampayan and John [5] developed one of the first MDC scheme (BMDC-VJ) for predictive coding environment. They use multiple description scalar quantizers (MDSQ) [49] in each of the side loops. Hence the descriptions are mutually refining, i.e., each additional description improves the decoder performance, and are balanced, i.e., each description has approximately the same quality and bitrate. An example of the two side quantizers is shown in Fig. 4.5. To maintain the mutual refining property for DPCM coders it is necessary that the shifted quantizers in the two loops have aligned thresholds. The scheme ensures this by quantizing the predicted values to multiples of the quantizer step size, so that both quantizers will always be shifted by multiples of their step size. This quantization of prediction leads to a loss in performance in the side decoders but this loss is a trade off against an increase in the central SNR. As mentioned in the introduction of this chapter, the CSE algorithm can be used to estimate data lost in BMDC-VJ thus improving its performance in case of erasures in a description. We also compare our proposed MDC scheme with BMDC-VJ.

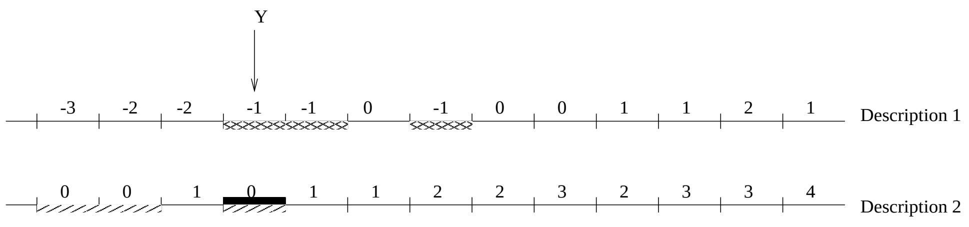


Figure 4.5: The two side quantizers of BMDC-VJ [5] with their index assignment. Input Y is quantized to the bin in black, and indexes (-1,0) are sent as the description to the decoder. If both are received, the decoder uses reconstruction values of the black bin. If one description, e.g. description 2, is received the centroid of the reconstruction levels of the three bins (shaded with a slant) with this index is used.

BMDC-VJ designs balanced descriptions, unbalanced descriptions could also be designed such that one of the descriptions has a high quality and the other has a low, but acceptable, quality. The low resolution description is primarily used as redundancy, to be decoded in case of losses in the high resolution description. An example of an unbalanced scheme is the robust audio tool (RAT) [71]. The advantage of an unbalanced MDC system is that it is easier to optimize the level of redundancy given channel conditions, mainly because no special quantizers are needed. Note that the descriptions can be unbalanced, however, using appropriate packetization strategy, e.g., the one used in RAT, packets of equal importance can be designed. In our MDC scheme for DPCM we present a method for scaling from unbalanced to balanced descriptions.

Another MDC based predictive coder is the video coder developed by Reibman *et al.* [72]. The descriptions are generated by using the pairwise correlating transform [60]. In case of erasures in one description, a linear estimator derived from the correlating transform is used to estimate the lost data. Our erasure recovery scheme, on the other hand, uses the inherent memory of the source to perform a maximum likelihood sequence estimation. Reibman *et al.*'s work has been specifically designed for video data. We have not yet applied our technique to video data, we propose it as part of future work.

Sequence estimation algorithms are used in many channel decoders. Two which deserve mention, because they use the source memory, are [73] and [21] where the authors have used the residual redundancy in a DPCM loop and the catastrophic nature of the VLC codes to correct errors. In principle this is similar to our work, except that we are exploiting the correlation between the description and the memory of the source to estimate erased data. Similar work has been presented by Yao and Vaishampayan [74] for a MDC based PCM scheme, where each description is VLC coded and a Viterbi based sequence decoding algorithm has been proposed.

One of the problems associated with MDC based prediction systems is that of choosing the best predictors. If both descriptions are received, the best predictor would be formed from past information transmitted on both the descriptions. However, if only a single description is received, prediction should be based on information corresponding to the one description that has been received, otherwise, if the prediction is based on information unavailable at the decoder, there will be a prediction loop mismatch leading to a poor MSE performance. Regunathan and Rose [57] have developed efficient prediction for MDC based on an estimation theoretic approach. In their work, information available at the decoder is used to calculate regions in which the original signal should lie. This is very similar to our definition of a consistent region (Section 4.4), though we developed our work independently in [75]. In addition, we introduce a sequence based algorithm for estimating data lost due to channel noise, an issue which none of the previously proposed algorithms addressed.

4.4 Overview of the proposed scheme

Fig. 4.7 shows the block diagram of our proposed MDC based scheme. The source is encoded using two parallel DPCM loops, where each DPCM coder uses a quantizer with a different step size (for uniform threshold quantizers) or different number of bins (for a Lloyd Max quantizer) thus generating two descriptions. Note that these quantizers will be replaced by MDSQ for the BMDC-VJ scheme, the rest of the system remains the same.

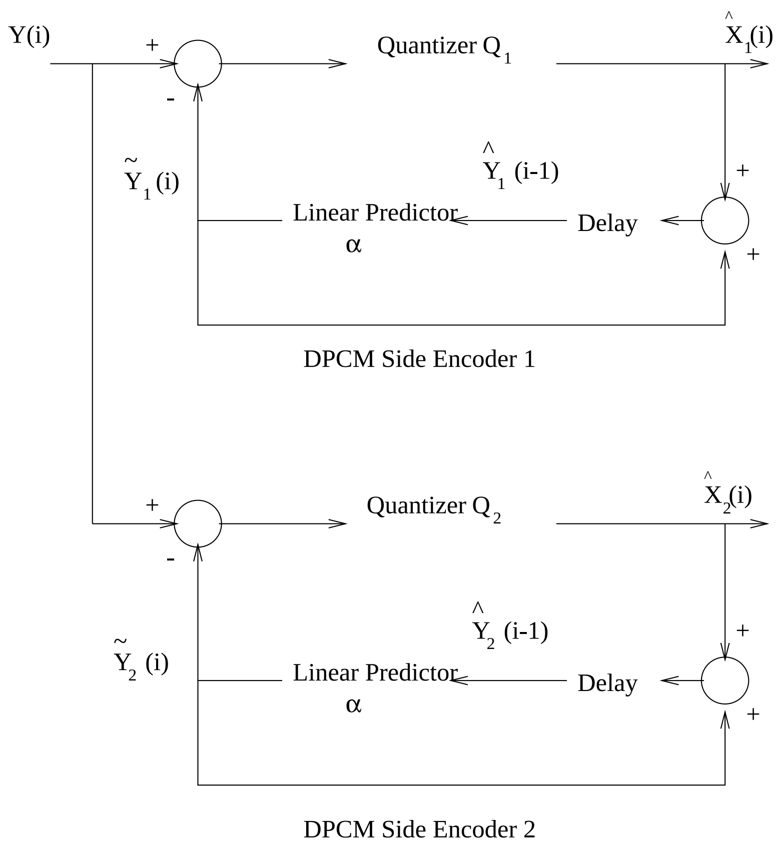


Figure 4.6: A MDC DPCM encoder.

The encoder with its parallel DPCM loops is shown in Fig. 4.6. The input Y is coded using quantizers Q_1 and Q_2, so that $\hat{Y}_1$ and $\hat{Y}_2$ are the outputs. $\hat{X}_1$ and $\hat{X}_2$ are the corresponding quantized prediction error sequences which correspond to the descriptions S_1 and S_2 in Fig. 4.7. Note that, unless otherwise stated, quantized variables are denoted with a hat and a variable without an index denotes a sequence, i.e., X is a sequence while $X(i)$ is the ith sample of the sequence. In addition to the side DPCM loops, an optimization algorithm is used to find the description S_1, for a given set of quantizers and description S_2, the algorithm will maximize the central SNR if both descriptions are received at the decoder. The descriptions are packetized before being transmitted over the network. Packets of equal size can be created by packetizing each description separately for the balanced MDC. For unbalanced MDC the RAT packetization strategy can be followed; each packet contains high resolution samples of the present frame and low resolution samples of a previous frame (a frame is a segment of data in time). Thus from the packet network viewpoint, each packet, in either case (balanced or unbalanced) has equal priority and the same size.

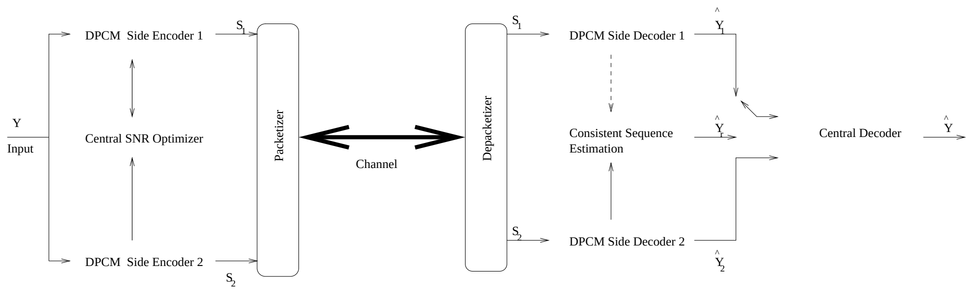


Figure 4.7: Block diagram of the proposed system. Two side DPCM loops generate the prediction errors, corresponding to the descriptions S_1 and S_2. The central SNR optimization block searches for the optimal S_1, given the quantizers and the S_2, which would maximize the central decoder's SNR if both descriptions were received at the decoder. If both descriptions are received then they are decoded to $\hat{Y}_1$ and $\hat{Y}_2$ and used as inputs to the central decoder. If one of the descriptions, say S_1, has a loss, then the consistent sequence estimation algorithm estimates the erased data. The inputs to this estimation algorithm are $\hat{Y}_2$ and the received data of description one. The estimated signal, denoted by $\hat{Y}_r$, is used as input to the central decoding to ensure that the performance is as close to central SNR as possible.

If both descriptions are available at the decoder then the central decoder decodes the descriptions. In the case of loss, for example in S_1, the sequence estimation algorithm uses the received S_1 and S_2 data to estimate $\hat{Y}_1$. The estimated signal $\hat{Y}_r$ can then be used as an input to the central decoder in order to achieve an outcome close to the central SNR performance.

The assumption underlying the estimation algorithm is that there are no correlated losses, i.e., erasures do not occur at the same location in both the descriptions. This can be ensured by using sufficient interleaving among packets containing different descriptions for a bursty channel. If the erasures lie in the same segment of the data for both descriptions then the data cannot be decoded.

There are three important observations to be made about this MDC-DPCM system. We are quantizing the same source using two different quantizer. Hence the bin that the source is quantized to by one quantizer should overlap with the bin that it is quantized to by the second quantizer. In the case of DPCM loops, there will be an overlap only after the appropriate shifting of each quantizer to account for the different predictors used in each loop. Fig. 4.8 shows an example where each quantizer has been shifted by its prediction. The sample X is then quantized to bin 2 of Q_2 and the overlapping bin 7 of Q_1.

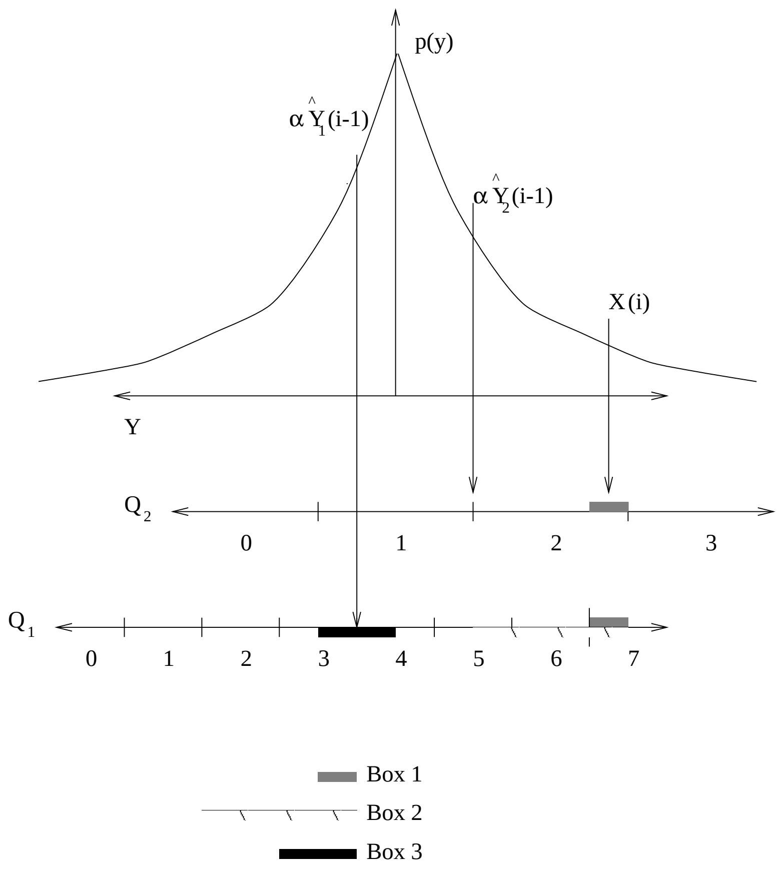


Figure 4.8: Quantization in DPCM is equivalent to using a scalar quantizer with its center shifted to the value of predictor. In this example Q_1 & Q_1 are shifted by $\alpha\hat{Y}_1(i-1)$ & $\alpha\hat{Y}_2(i-1)$ respectively. The prediction error by the two quantizers is quantized to $\hat{X}_1(i) = 2$ and $\hat{X}_2(i) = 7$ respectively. It is clear that the source will always be quantized to bins which overlap. Note that Y is the source being quantized while X is the prediction error being generated by the DPCM loop.

The observations are closely tied with our definition of *consistency* and consistent region:

Definition 1 *Consistency: Given quantizers Q_1 and Q_2, a sample, $\hat{X}_1(i)$ is consistent with $\hat{X}_2(i)$ if there exists an input $Y(i)$ such that $Q_1(Y(i)) = \hat{X}_1(i)$ and $Q_2(Y(i)) = \hat{X}_2(i)$.*

Definition 2 *Consistent Region R^c The consistent region given quantizers Q_1 and Q_2 and output sequences $\hat{X}_1$ and $\hat{X}_2$ is the set of inputs that can generate these outputs:*

$$R^c = \{Y : \forall Y,\ \hat{X}_1 \textit{ is consistent with } \hat{X}_2\}$$

For a DPCM loop the quantizers are going to be shifted to their respective predictions. Hence the consistent region will be defined by the prediction errors, $\hat{X}_1(i)$ or $\hat{X}_2(i)$ and the respective predictions, $\alpha\hat{Y}_1(i-1)$ or $\alpha\hat{Y}_2(i-1)$.

The three observations are,

- **Observation 1** In Fig. 4.8, if $\alpha\hat{Y}_1(i-1)$, $\alpha\hat{Y}_2(i-1)$, $\hat{X}_1(i)$ and $\hat{X}_2(i)$ are available at the decoder, i.e., both the descriptions are received error free, then it can be inferred that $X(i)$ belongs to Box 1. As Box 1 is a subset of bin 7, using the centroid of the box for the reconstruction will give a lower distortion than if using the centroid of bin 7. In general, when both descriptions are received at the decoder the consistent region R^c is a subset of the bins of the quantizer and its centroid can be used for reconstruction to get a lower distortion. This result will be used in the central decoding algorithm developed in section 4.5.

- **Observation 2** From the figure, if $\alpha\hat{Y}_1(i-1)$, $\alpha\hat{Y}_2(i-1)$, and $\hat{X}_2(i)$ are available at the decoder, then it can be inferred that $X(i)$ belongs to Box 2. As Box 2 overlaps with bins 5,6,7 of Q_1, if sample $\hat{X}_1(i)$ has been lost, then the lost sample has three possible *candidates*, namely 5,6,7. In general, if there is an erasure on a single description, at the point of erasure the second description can be used to define the consistent region R^c which would give the possible candidates for the lost data. This will be used in the estimation algorithm to generate candidates in section 4.6.

- **Observation 3** Suppose there is a single erasure in description one but at some point beyond the erasure, $\hat{X}_1(i) \in$ bin 7 and $\hat{X}_2(i) \in$ bin 2, $\alpha\hat{Y}_2(i-1)$ are available at the decoder. For $\hat{X}_1(i)$ and $\hat{X}_2(i)$ to be consistent the bins have to overlap, i.e., the prediction $\alpha\hat{Y}_1(i-1)$ has to belong to Box 3. If a candidate (observation 2) has been selected for the erased sample $\hat{X}_1(e)$, $(e < i)$, such that $\alpha\hat{Y}_1(i-1)$ is not in Box 3, it can be inferred that the candidate for $\hat{X}_1(e)$ is inconsistent. Box 3 corresponds to the bounds discussed in Fig. **??**. This observation can be used to check inconsistencies that may occur in the future if an incorrect candidate is selected for an erased data. It will be further discussed in the path consistency check step of the estimation algorithm in section 4.6.

Before ending the section, let us introduce some notation which will be used in the central optimization algorithm and the estimation algorithm introduced in the next two sections. Let C_1 and C_2 represent the codebook partitions of Q_1 and Q_2 respectively. Note that when the quantizers are shifted in a DPCM loop, the partitions will be shifted accordingly. In the DPCM encoder with the predictor coefficient α, for any sample i

$$\begin{aligned} X_1(i) &= Y(i) - \alpha\hat{Y}_1(i-1) && (4.4)\\ X_2(i) &= Y(i) - \alpha\hat{Y}_2(i-1) && (4.5)\\ \Rightarrow X_1(i) &= X_2(i) + \alpha(\hat{Y}_2(i-1) - \hat{Y}_1(i-1)). && (4.6) \end{aligned}$$

Let $\epsilon = \hat{Y}_2(i-1) - \hat{Y}_1(i-1)$, then we have

$$X_1(i) = X_2(i) + \alpha\epsilon.$$

Finally, if $\hat{X}_2(i) = j$, then, $X_2(i) \in C_2(j) \triangleq [a_j, b_j]$, and if $\hat{X}_1(i) = k$, then, $X_1(i) \in C_1(k) \triangleq [A_k, B_k]$.

4.5 The central SNR optimization algorithm

At the central decoder both descriptions are received and decoded; from observation 1 the consistent region R^c is a subset of the bins of either of the quantizers because of the overlap. Using notation from section 4.4, R^c is defined as,

$$R^c \triangleq [a_j + \alpha\epsilon, b_j + \alpha\epsilon] \cap [A_k, B_k] \tag{4.7}$$

Let $\bar{X}(i)$ be the centroid of R^c. As R^c is a subset of $C_1(k)$ and $C_2(j)$, reconstruction with $\bar{X}(i)$ is going to give a lower distortion than $\hat{X}_1(i)$ or $\hat{X}_2(i)$. If there is a model for the source then the centroid could be calculated based on the model, or else the center of R^c could be used.

The best central SNR performance can be achieved with the following central decoder [57],

$$\hat{Y}(i) = \alpha\hat{Y}(i-1) + \bar{X}(i) \tag{4.8}$$

iff there is a corresponding central encoder such that,

$$\bar{X}(i) = Y(i) - \alpha\hat{Y}(i-1) \tag{4.9}$$

In order to achieve this central decoder performance and maintain independence of the descriptions, three independent encoders/decoders are needed and $\bar{X}$ has to be transmitted along with $\hat{X}_1$ and $\hat{X}_2$. This scheme has been used by Reibman *et al.* [72]. Though the scheme gives optimal central SNR performance its disadvantage is that there is an associated overhead of transmitting $\bar{X}$.

We on the other hand run two DPCM loops at the encoder and decoder, and transmit $\hat{X}_1$ and $\hat{X}_2$. $\bar{X}$ is derived at the decoder and is used to refine the prediction error of the S_1 decoder loop, i.e., our central decoder equation is,

$$\hat{Y}(i) = \alpha\hat{Y}_1(i-1) + \bar{X}(i) \tag{4.10}$$

Thus we avoid overhead cost of transmitting $\bar{X}$, but we do have a lower central SNR performance than that of the central encoder/decoder defined by equations (4.8 &4.9). Equation (4.10) gives us a gain in central SNR of about 1 dB over the side SNR, SNR_{S_1}. The question that arises is that can we do some processing at the encoder to increase the central SNR_c ? In Fig. 4.8, given the quantizers and $\hat{X}_2$, $\bar{X}(i)$ is dependent on $\alpha\hat{Y}_1(i-1)$, i.e., on $\hat{X}_1(i-1)$. Thus, if there was a different $\hat{X}_1(i-1)$ there would be a different $\bar{X}(i)$, i.e., each possible sequence $\hat{X}_1$ will give a different R_c and thus a different central SNR. Hence a central optimizing algorithm can be designed which consists of choosing among all possible $\hat{X}_1$ sequences that sequence which gives the highest central SNR given Q_1, Q_2 and $\hat{X}_2$. The sequence that gives the highest central SNR will obviously not give the highest side SNR. Hence in this search there is a tradeoff between the central and side distortions.

Given a Q_1, a trellis describing all possible prediction error sequences can be designed. Each state corresponds to a reconstruction value (centroid) of a quantization bin of Q_1 (the possible values that a prediction error $\hat{X}_1(j)$ can take) and every state is connected to every other state. Given the sequence $\hat{X}_2$, Viterbi algorithm [19] is used to find the minimum $D_c + \lambda D_{s_1}$. λ is the parameter which controls tradeoffs between the side distortion and the central distortion. There are many sequences which may not be possible and can be removed at run time.

Fig. 4.9 shows the tradeoff between the side S_1 and central SNR for the balanced and unbalanced cases. The points on the SNR axis are results when no lookahead is used. In the balanced case, our search algorithm gives a gain of nearly 4dB over the no lookahead case, while for the unbalanced case a more modest gain of about 0.7 dB is achieved. For either case, note that the highest side SNR is obtained by using the lookahead with $\lambda = 0$ and that the decrease in side SNR is not more than 0.5 dB.

4.6 Consistent Sequence Estimation Algorithm

In the previous section we proposed an algorithm for generating multiple descriptions in a DPCM coding environment. In this section we develop an algorithm for estimating the data lost in a description due to channel losses. The first subsection develops the algorithm for fixed length coded descriptions. The variable length coded case and the associated errors in the bitstream are covered in the

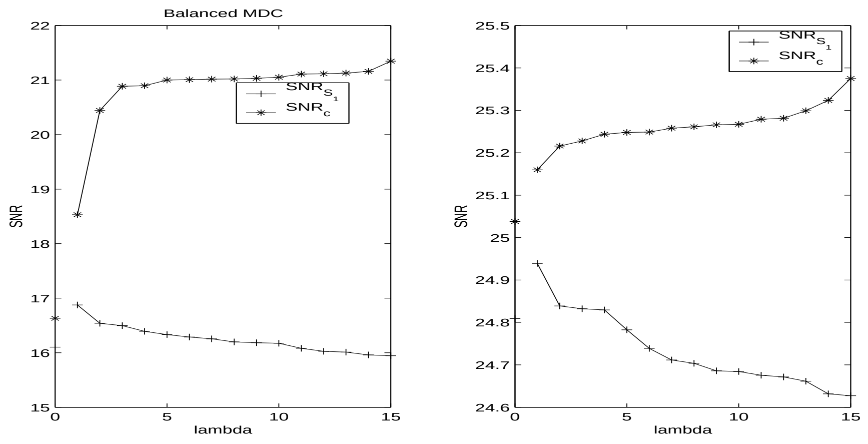


Figure 4.9: Tradeoff between the central SNR_c and side SNR_{S_1}. Results are for Gaussian Markov source with correlation 0.9. Uniform threshold quantizers are used. In the balanced case the total bit rate (after Huffman coding) is 1.99 & 1.92 for the two descriptions. For the unbalanced case, the bit rate is 3.23 for S_1 and 0.75 for S_2.

second subsection.

4.6.1 Fixed length coded DPCM

We start with the outline of our algorithm, assuming that only sample $\hat{X}_1(e)$ of the description one is lost at the decoder while description two is received error free. Our goal is to estimate the lost sample by taking into account the information that was received. The algorithm works through a 3 step process:

- *Candidate selection*: Of all possible quantized values for $\hat{X}_1(e)$, only those that are consistent with $\hat{X}_2(e)$ are considered as candidates. In mathematical terms the candidates can be found as follows: if $\hat{X}_2(e) = j$, i.e., $X_2(e) \in C_2(j) \triangleq [a_j, b_j]$, then $X_1(e) \in R^c$ where R^c is defined as

$$R^c \triangleq [a_j + \alpha\epsilon, b_j + \alpha\epsilon] \tag{4.11}$$

  All the bins of Q_1 that intersect with R^c are candidates for $\hat{X}_1(e)$. This is related to observation 2 of section 4.4.

- *Path consistency check*: For each of the above candidates, description one is decoded N samples into the future. Observation 3 is then used to check consistency of each of these decoded sequences. For an incorrect candidate, the shift of quantizer Q_1 would be such that bins of the received $\hat{X}_1(i)$ and $\hat{X}_2(i)$ for some $i > e$ will not overlap.

  Mathematically, the path consistency check can be derived as follows. If $\hat{X}_1(i) = j$ is given, i.e., $X_1(i) \in C_1(j) \triangleq [A_j, B_j]$ then $X_2(i) \in r$ where r is defined as

$$r \triangleq [A_j - \alpha\epsilon, B_j - \alpha\epsilon] \tag{4.12}$$

  Given a sequence of $\hat{X}_1(i)$, $i > e$, the reconstructed sequence $\hat{Y}_1(i)$ is consistent with $\hat{Y}_2$ if, at each sample, the interval r from (4.12) overlaps with the the quantization bin of $\hat{X}_2(i)$.

- *Likelihood test*: Finally, the consistent sequence closest to $\hat{Y}_2$, in Euclidean distance is chosen as the recovered $\hat{Y}_1$.

The example shown in Fig. 4.10 can be used to explain the algorithm. Here $\hat{X}_1(0) = 1$ is lost but we assume that $\hat{X}_2(0)$, $\hat{Y}_1(-1)$, $\hat{Y}_2(-1)$ and $\hat{X}_1(i), \hat{X}_2(i)$ $\forall i > 0$ have been received correctly. Candidate selection gives three possible candidates for the erased data, namely $\{0, 1, 2\}$. Each of these three choices are decoded, their paths are shown in the figure, and checked for consistency. In the figure the top path, corresponding to $\hat{X}_,(0) = 2$, becomes inconsistent; after shifting Q_1 by the prediction corresponding to this path, the bin of $\hat{Y}_1(2)$ does not overlap with the bin of $\hat{X}_2(2)$. The path consistency is tested for N samples in the future and if there are more than one consistent path, then the one closest to $\hat{Y}_2$ is chosen.

The above algorithm has been explained for a single erasure and it can easily be extended to multiple erasures. At each erasure, the candidate selection algorithm is run to find the candidates and after the end of erasures, the path consistency check is run to remove all the inconsistent paths. It can be easily seen that for burst erasures a tree of candidates will grow: for each candidate for the previous erased sample, there will be a set of candidates for the current erased sample. This increases the complexity of our algorithm and is its main limitation. To limit the complexity, pruning is used where only M candidates are kept at any

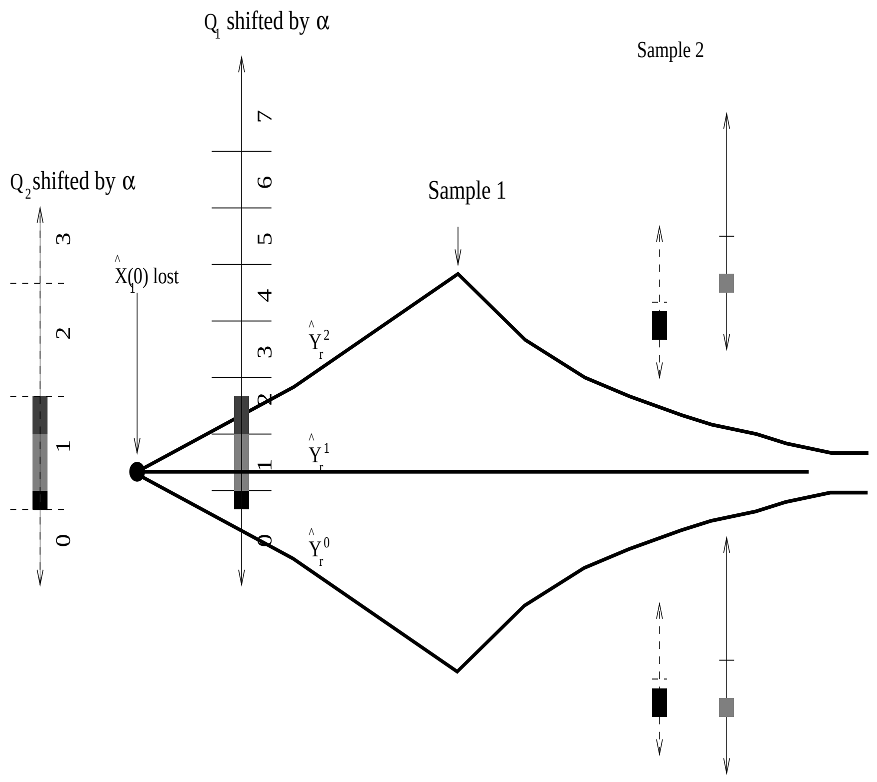


Figure 4.10: Erasure Recovery Algorithm. $\hat{X}_1(0)$ is erased. $\hat{Y}_1(-1), \hat{Y}_2(-1), \hat{X}_2(0)$ give 3 possible candidate for lost data: $[0, 1, 2]$. Each of the candidates is decoded, $\hat{Y}_r^k$ represents the estimated path decoded with $\hat{X}_1(0) = k$. Consistency check is done for the decoded paths, at sample 2 the top path is inconsistent: after shifting by $\alpha\hat{Y}_r^2(1)$, the bin of $\hat{X}_1(2)$ does not overlap with the bin of $\hat{X}_2(2)$, shifted to $\alpha\hat{Y}_2(1)$. The bins are shown by the shaded boxes on each quantizer. In case of multiple consistent streams, the one that is most likely i.e. closest (in Euclidean distance) to $\hat{Y}_2$, is chosen.

erasure. As pruning can remove all consistent paths, it affects the performance of our algorithm. We are looking at ways to solve this problem.

Fig. 4.11 shows the result of our algorithm for erasure burst lengths of up to 7 samples. Gaussian Markov data with a correlation coefficient of 0.9 is encoded with two DPCM loops using Lloyd Max quantizers. The high resolution uses a 3 bit quantizer while the low resolution uses a 1 bit quantizer. Hence, the total bit-rate is 4 bps. SNR_{Y_1} is the SNR of description one, SNR_{Y_r} is the SNR of the estimation algorithm while $SNR_{Y_{rnla}}$ is that of the recovery without using any lookahead. Note that this result does not take the central decoder into account. Only the side decoder performance is given and compared with that of the estimation algorithm. The results show that for small, but catastrophic, erasures the estimation algorithm is able to produce estimates which are very close to original signal. Lookahead performs better than no lookahead because a "good" local estimate may prove to be inconsistent in the future. N, the amount of lookahead, depends upon the predictor coefficient and the relative bit-rates of the two descriptions. We do not have an explicit formula for N but for all our experiments $N = 20$ has sufficed.

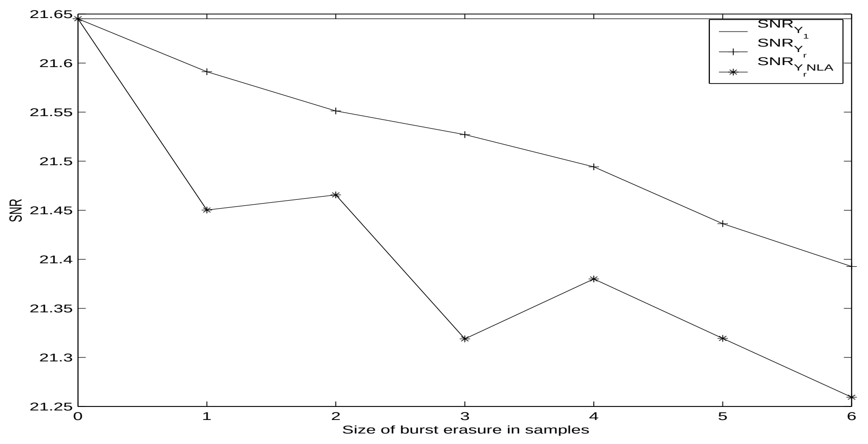


Figure 4.11: Results for erasure recovery algorithm when fixed length codes are used. Gaussian Markov data with correlation coefficient 0.9 is encoded with two DPCM loops using Lloyd Max quantizers. The high resolution uses a 3 bit quantizer, while the low resolution uses a 1 bit quantizer resulting in a total bit-rate of 4 bps. SNR_{Y_r} is the output of the recovery algorithm, $SNR_{Y_{rnla}}$ is for recovery without using any lookahead, i.e., no path consistency check is used and the likelihood test is performed on the sample not on the sequence.

4.6.2 Entropy Coded DPCM

In the above section we assumed that the DPCM descriptions were fixed length coded. However in most practical scenarios, including BMDC-VJ, the prediction errors will be entropy coded. In this section we develop an algorithm for recovery of erasures when the bitstream is Huffman codes encoded.

Fig. 4.12 shows a variable coded bit stream between two resynchronization markers [70]. The number of symbols, N, between the resynchronization markers is known. Consider that a segment of the bit stream is lost (the shaded box). With reference to the bit stream, the starting point of the erasure s and its end point e are known. The bitstream packet can be decoded at most to the s^{th} bit.

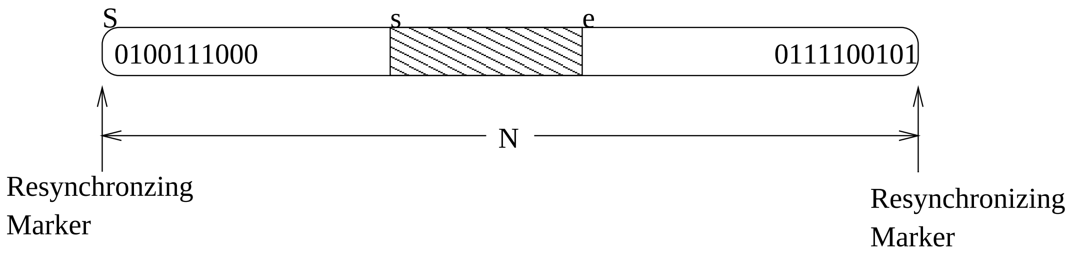


Figure 4.12: Example of a packet loss in a Huffman coded environment. A VLC bit stream between two resynchronization markers is shown. The number of samples coded between the markers is known. The shaded box represent erasures in the bit stream. Thus the start of loss s and the end of loss e is known. The bit stream can be decoded at most to bit s and also after the right resynchronization marker.

To estimate the erased samples, the candidate selection and path consistency check introduced in the previous subsection are used along with additional constraints. The additional constraints are that if all the candidates for an erasure are VLC encoded, the resulting bitstream should (i) be consistent with bits that have been received, (ii) consistent with the number of bits erased and (iii) consistent with the number of symbols N between the resynchronization marker.

The algorithm is best explained by using the example shown in Fig. 4.13. The prediction error $\hat{X}_1$ can be represented by any of the three symbols $\{a, b, c\}$, and each symbol has its associated Huffman code. Sequence $\hat{X}_1 = acac$ is Huffman encoded and sent over the channel. Two bits (third and fourth) are erased. At the decoder only the first bit can be decoded, to a; the second bit (1) does not correspond to a symbol. Let *candidate selection* give two candidates for the second prediction error: a, c. The Huffman code of the string aa is 00 which does not match the first two error free bits (01), hence aa path is discarded. For the ac path *candidate selection* is run again and gives three possible prediction error paths acc, acb, aca. Of these, acb can be discarded because the fifth bit of this path, when VLC coded, does not match the fifth bit in the received string. Further, for path acc the sixth bit of the received sequence cannot be decoded, thus this path is also discarded. The only valid path in this example is $acac$ which was the originally transmitted path. If more than one paths are left at this stage, *path consistency* tests will be run for it and the consistent path closest to $\hat{Y}_2$ will be the estimated $\hat{Y}_r$.

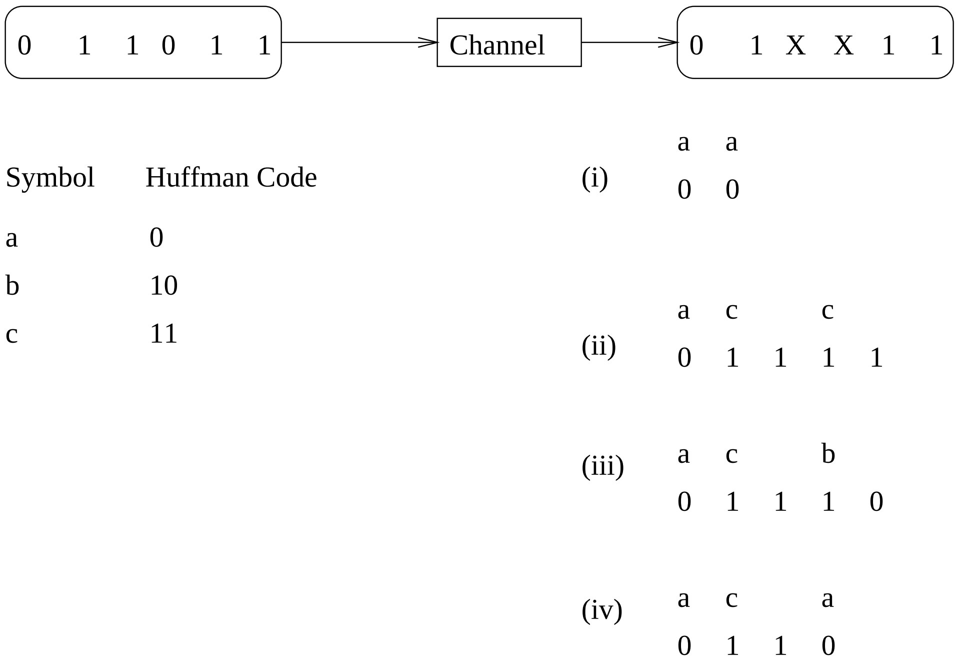


Figure 4.13: Sequence $\hat{X}_1 = acac$ is transmitted over a noisy channel. The third and fourth bits are erased in the transmission. Candidate Selection gives four possible paths for the erased samples. Of these *aa* is not valid as its 2nd bit does not match the 2nd bit of the received string. *acb* is not valid as the 5th bits do not match and *acc* is not valid as the sixth bit cannot be decoded. Only valid path in this example is *acac*.

The results of this algorithm for UMDC are shown in Fig. 4.14. Note that the size of burst erasure has been kept small to avoid the exponential growth in number of candidates associated with long burst erasures. In the results section we experiment with much longer bursts, in any case even if a single bit is erased, the decoder cannot decode till the next resynchronization marker. In view of this our results are significant as the estimated signal is very close to the original signal for the short burst erasures considered here. Long bursts can always be avoided by using some form of interleaving.

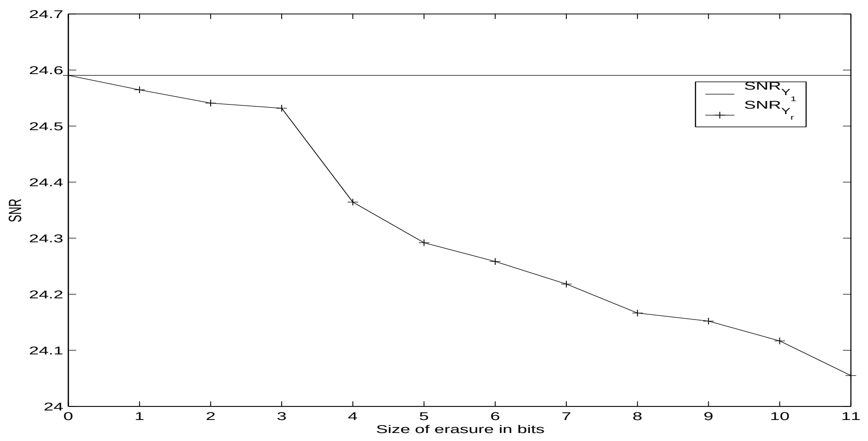


Figure 4.14: Results for erasure recovery algorithm when variable length codes are used. Gauss-Markov data with correlation coefficient 0.9 is encoded with two DPCM loops using uniform threshold quantizer. The high resolution is coded at 3.23 bps, while the low resolution is coded at 0.78 bps, resulting in a total bit-rate of 4 bps. SNR_{Y_1} is the decoded description S_1 without errors, SNR_{Y_r} is the estimated signal when S_1 has errors.

4.7 Results

We first extend our sequence estimation algorithm to the adaptive DPCM encoder and compare it to DPCM for a speech signal. Then, we compare the various MDC schemes, BMDC, UMDC and BMDC-VJ along with a comparison to the traditional method to avoid error propagation, namely restarting loop at the beginning of each packet.

4.7.1 ADPCM

Speech/Video signals are non-stationary in nature, i.e., they have time varying source statistics. An adaptive quantizer and/or predictor learns the source statistics as it codes the source and thus performs better than a fixed coder. In their book, Jayant and Noll [14] have shown the advantage of using adaptive quantizers and predictors, e.g., for 24kbps speech coding adaptivity can give a gain of over 4dB. The gain in performance comes at the additional cost of sensitivity of errors. In DPCM the error decays, with lookahead n, at rate α^n. In ADPCM there is an additional error due to the wrong step size which does not decay with n.

In our implementation of ADPCM we have used only adaptive quantizers and a first order fixed predictor. The adaptive quantizers are based on the step size multipliers design for uniform quantizers and speech inputs developed by Jayant [76]. In Fig. 4.15 results with ADPCM and its comparison with DPCM are shown for speech data. Note that for DPCM we use uniform quantizers with range equal to three times the estimated variance of the signal. The estimation is done for every 1000 samples, when the coding loop is restarted. Results show that for low erasures ADPCM uses its memory to recover better than DPCM, but if the bursts becomes very large ADPCM's performance is worse. This is to be expected as ADPCM will have more constraints so it will recover more often but if it does not recover the error will propagate further.

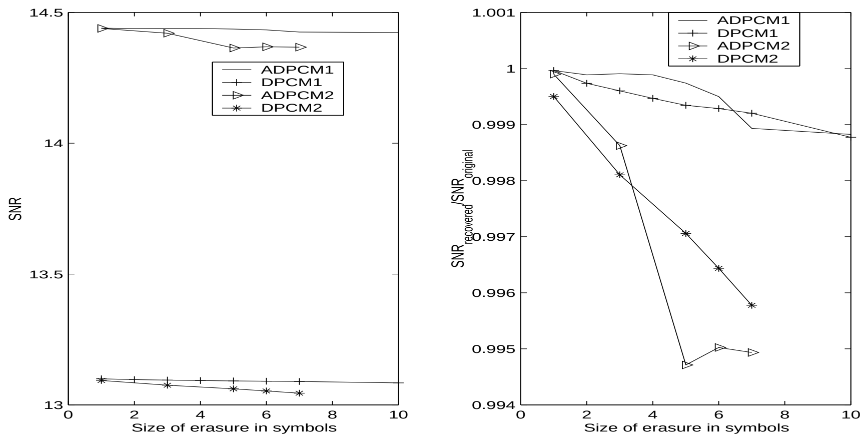


Figure 4.15: Results for ADPCM and comparison with DPCM, both using uniform quantizer. The left figure gives the SNR of the recovered signal while the right figure gives the ratio of recovered SNR to the original SNR. In ADPCM1/DPCM1, $\hat{Y}_1$=4 bps, $\hat{Y}_2$ =3 bps, while ADPCM2/DPCM2 $\hat{Y}_1$ =4 bpp and $\hat{Y}_2$=2 bps.

4.7.2 Comparisons between various schemes

We start with a comparison of our scheme with the MDC-DPCM scheme of Vaishampayan and John (BMDC-VJ). Note that all the results for BMDC-VJ are based on our own simulations of their scheme. For a given total bit rate there is one balanced MDC (BMDC), but there can be many unbalanced MDC (UMDC) schemes. We design a UMDC scheme based on the principle that the central SNR of unbalanced is close to the central SNR of BMDC-VJ. The optimal performance along with the step size of the uniform threshold quantizer are given in Table 4.1. The results are for a Gauss-Markov source with correlation coefficient 0.9.

Table 4.1: Results of UMDC,BMDC, BMDC-VJ

Scheme		S_1	S_2	$Central$
Balanced	SNR	16.00	16.31	21.25
	Bit Rate	1.99	1.92	3.91
	Step size	1.3	1.31	
Unbalanced	SNR	24.62	8.31	25.25
	Bit Rate	3.23	0.75	3.98
	Step size	0.46	3.7	
BMDC-VJ	SNR	13.18	13.25	25.12
	Bit Rate	2.03	2.03	4.06

In Fig. 4.16, we compare the expected SNR at the decoder, given a probability of loss p of losing a description, for the three different cases. The expected distortion $E(D)$ is calculated as,

$$E(D) = (1-p)^2 D_c + p(1-p) D_{S_1} + p(1-p) D_{S_2} + p^2 D_l$$

where D_l is the distortion when both the descriptions are lost. The figure shows that the UMDC has better results than BMDC-VJ for very low p and BMDC is better than BMDC-VJ for $p > 0.1$. These results are obtained when no CSE algorithm is used.

In Fig. 4.17, we present results for our estimation algorithm when applied to BMDC-VJ and compare it to the UMDC case. Again small burst erasures are considered, BMDC-VJ is able to recover more erasures because it has more balanced information, hence less candidates.

In Fig. 4.18 we compare BMDC and UMDC for different burst erasure sizes. The step like function in the result is due to the pruning. It is clear from the results that, if there is a large burst of erasures, BMDC will do better than UMDC as the minimum side SNR for BMDC is higher than the minimum side SNR of UMDC.

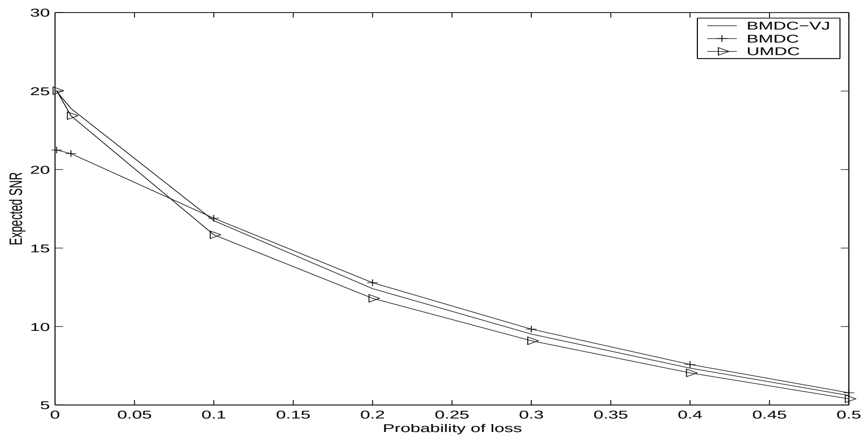


Figure 4.16: Comparison of expected SNR with given probability of loss for three different schemes. UMDC is the unbalanced MDC scheme described above, BMDC is the balanced scheme. BMDC-VJ is the scheme from [5]

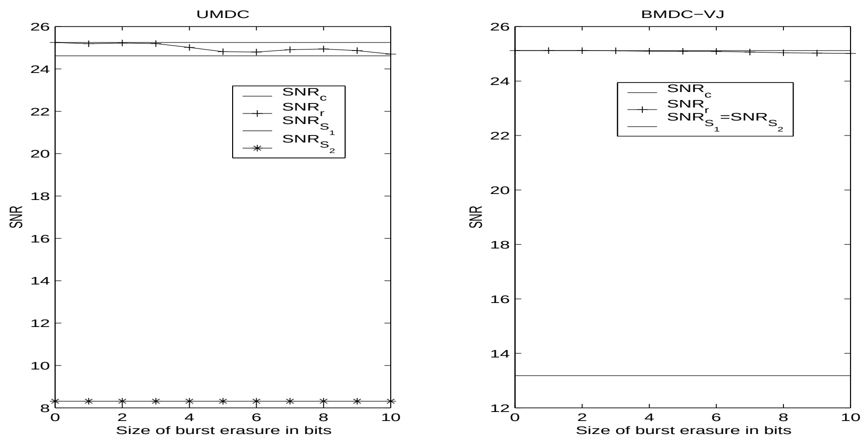


Figure 4.17: Results for the channel estimation algorithm for UMDC and BMDC-VJ. In the BMDC-VJ[5] there is no estimation of lost data, if there are erasures in a description it is discarded and the other description is used for decoding. This would mean a loss of about 12 dB (central to side SNR), with our algorithm close to central SNR performance can be obtained.

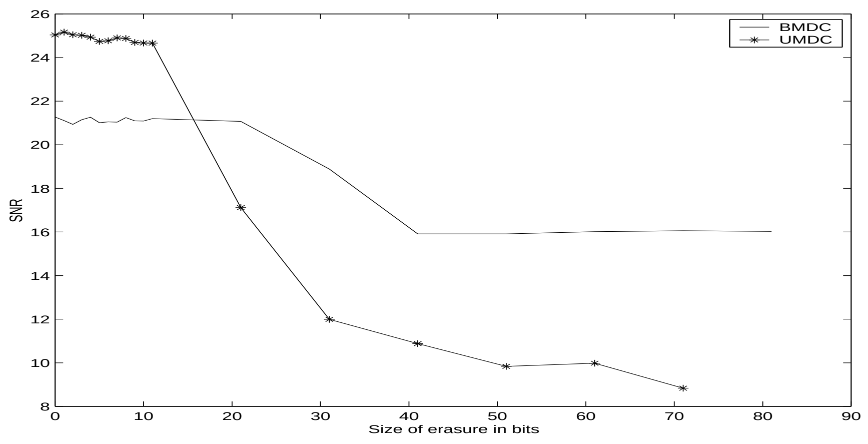


Figure 4.18: The plot compares the UMDC and BMDC scheme of Table 1. A Gaussian Markov source with correlation 0.9 is used. The descriptions are Huffman coded and the VLC erasure recovery algorithm is used.

We have mentioned pruning as the main reason for the degradation in performance as the size of burst erasure grows. A way of avoiding large bursts is to interleave samples across packets. One such packet with interleaved samples is shown below.

$\boxed{X(i+1), X(i+2), ...X(i+5), X(i+21), X(i+22), ..X(i+25), X(i+41)...}$.

A 1000 sample stream was packetized into 4 such packets, a packet loss implies that 250 or 1/4 of the samples were lost. The erasure recovery algorithm estimated the lost data to within 0.4 dB of the central SNR. As the longest burst erasure with this interleaving is 5 samples pruning is not needed in the CSE algorithm.

A simple approach to prevent error propagation in predictive coders is to restart the prediction loop by periodically inserting PCM-coded samples. In Fig. 4.19 SNR_s is the SNR of the 2000 samples Gaussian Markov source, coded with 4bps, when there are no restarts in the loop. SNR_p is the SNR when the loop is restarted with every packet. If a packet is lost, then SNR_{pr} is the performance at the decoder, where the previous packet samples are used in place of the lost samples. We compare this scheme to our UMDC scheme, where the high resolution description is coded at 3bps and the low resolution at 1 bps. MDC based schemes do well for small burst erasures, and even when a long burst occurs in a description, the second description can be used for decoding.

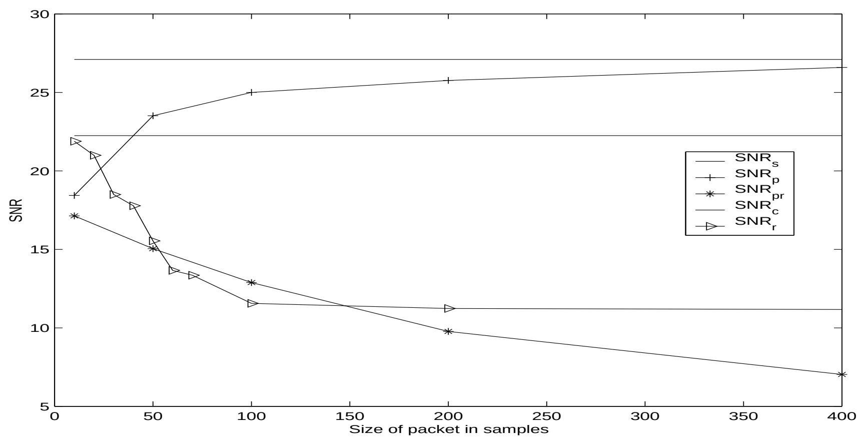


Figure 4.19: The plot compares the restart with each packet scheme to the MDC scheme. A 1000 symbol source is coded with 4 bits and packetized with different packet size. If the loop is not restarted the performance is SNR_s and if it is restarted the performance is SNR_p. If there is a packet loss, the performance is SNR_{pr}. MDC with 4 bits, 3 for the description 1 and 1 for the description 2 is designed. SNR_c is the quality if everything is received error free, else SNR_r is the performance of the recovery algorithm.

4.8 Conclusion

In this chapter we have designed a robust DPCM scheme based on multiple description coding. Our scheme compares favorably with other MDC based DPCM schemes. More importantly, we have addressed the key issue of estimating data erased due to packet loss in a predictive coding environment. Our results show that small burst erasures can be recovered by using a sequence based estimator. By using interleaving of samples we can also achieve very good results when a packet is lost. Our algorithm does not guarantee that the erased data will be recovered or that there will not be any error propagation. However, it provides a method for erasure recovery given all the information that has been received at the decoder. Restart points will still have to be used to limit error propagation, but they can be positioned further apart thanks to the added robustness provided by our erasure recovery techniques.

Chapter 5

Modeling of temporal dependence in packet losses using Information Theory concepts

In this chapter we present the last contribution of this thesis, channel modeling. With the increasing congestion on the Internet, packet losses and delay have become commonplace. These losses and delay can lead to severe degradation in the quality of delay constrained multimedia applications such as audio/video conferencing and Internet telephony. To ensure a suitable end-user quality and fair bandwidth sharing with the other connections, these applications have to adapt their transmission rate to the perceived congestion level in the network [1]. They can adjust the transmission rate either by adapting their encoding rate (bits per sample) [77], or by changing the number of layers they will send [78], or by adjusting the relative allocation between the source coding and channel coding bits [79, 4]. To enable applications to be adaptive, simple and efficient network loss models are needed so that the application can adopt the optimal strategy based on predicted future network behavior.

A number of studies have shown the network traffic is correlated, i.e., the packet losses exhibit a finite dependence in time [1]. Thus, the channel, i.e., the packet loss trace, can be modeled as a correlated random process, e.g. a Markov process, where the conditional probabilities of a symbol depend upon a function of a finite number of contiguous past observations. This function of the past observations will be referred to as the "context" of the symbol in this paper. Specifically, if the packet loss trace is represented as a binary message, with the symbol one representing a lost packet and the symbol zero a received packet, then the channel can be modeled as a binary random process. The problem of modeling the random process, generating this binary message, thus amounts to finding a set of contexts and their associated conditional probabilities.

A popular model for the packet loss is the Markov chain model [1]. In a Markov chain model of order k, the context of a symbol is the string of past k observations. Thus there are $N_C = 2^k$ contexts in this model. An advantage of the Markov chain is that a finite state machine (FSM) can be associated with it, where each state corresponds to a context and its associated conditional probabilities.

However, when fitting Markov chain models to the data by estimating the conditional probabilities of the contexts, a number of difficulties arise [36]. First, there is an explosive increase in the number of states (2^k) if the order is increased to find the the best fit. In the paper by Yajnik *et al.* [1], analysis show that models of up to order 40 may be required to best fit the trace, clearly the computational cost of such a large model will be very high.

Another problem with the Markov chain model is that some contexts occur very rarely in the data and when they are used for modeling they may not provide enough information about the process, this is referred to as the "context dilution" problem. Let us take a concrete example, the Markov chain model of order 3 for trace-27 (section 5.1) is shown in table 5.1. The counts in the table, one for each symbol, 0 and 1, are the number of times a symbol occurs for the given context and can be used to calculate the conditional probabilities for the context. Contexts $(110), (101), (011), (111)$, do not occur with any regularity, and therefore their counts are very low. Thus it may be more efficient to either delete these set of contexts from the model or maybe "lump" them together to form a new context. Similarly, contexts (100), (010) and (001) have very similar counts and thus a single context representing this set of contexts may give a better modeling performance. However, if the redundant parameters are removed by arbitrarily lumping together equivalent states, the result may not a finite state machine implementation of the process, let alone one of a Markov type [36].

Context	Counts	Context	Counts
000	360639, 3700	100	3702, 155
001	2989, 864	101	158, 46
010	2984, 163	110	872, 40
011	805, 106	111	106, 41

Table 5.1: Counts for Markov chain model of order 3.

In this paper, we use the universal modeling concept [36] to alleviate the problems of Markov chain models. A universal model is one which represents an entire class of probability distributions, such that it is able to capture the behavior of any model in the class. From this set of models, the "best" model class for the observed data is found. To define the "best" model we need a measure of the performance of a model. We use the concept of the "shortest description length" [37], where description length is defined as the negative logarithm of the probability distribution estimated for the given data, a concept similar to Shannon's entropy [30]. Thus to evaluate a model, the entropy of the observed data is calculated with respect to the model, the best model is the one which which gives the lowest entropy.

An example of universal model is the tree model proposed by Rissanen [25]. This model will be referred to as the Markov tree model, and a simplistic view of the model is that it is a collection of all possible Markov models, from which the most appropriate model is chosen for prediction of a symbol of the source [38]. In a kth order Markov chain model, memory of all the contexts was of the order k. On the other hand, in a kth order Markov tree model, all possible contexts with memory of order $1...k$ will be arranged in a tree, with the nodes of the tree representing a context and its associated probability distribution. The best set of contexts, i.e., a set of connected nodes, for the observed data can be efficiently found using the Context algorithm developed by Rissanen [25]. Thus rather than deleting or lumping contexts of the Markov chain, all possible contexts are grown and the best set is chosen, i.e., from an "over-complete" model, a model which captures essential information of the source is chosen. We propose to use this concept for modeling of network traces, an area where to the best of our knowledge it has not been used before. Our results show the improvement in modeling performance of the tree models over Markov chain models.

The Markov tree model not only performs well but is also truly adaptive: if the source changes not only the probabilities but also the set of contexts used to model the source can change. This is in contrast to a Markov chain model, where given a order, the set of contexts is fixed and only the probability distributions can be adapted. The performance of the tree model does come at the cost of increased complexity, there are $N_T = \sum_{i=0}^{k} 2^i$ contexts in the tree model of order (depth) k. The number of contexts and hence the complexity could be reduced by explicitly pruning the best set of nodes of a Markov tree model after the tree has been trained on some data. Given this set of contexts (nodes), the probability distributions can be adapted as in the Markov chain model. Our results show that the pruned tree model can perform better than a Markov chain model with comparable number of nodes.

The chapter starts with a review of related work in Section 5.1 followed by a brief introduction to some mathematical concepts in Section 5.2. In Section 5.3 we introduce the universal source coding algorithm and motivate the use of a tree structure over a Markov chain with some results. Finally, in Section 5.4 we give the design rules for pruning the Markov tree model and compare the pruned model with the Markov chain model.

5.1 Related Work

There has been a lot of work on measuring, analyzing and modeling network traffic [80, 81]. We are primarily concerned with Yajnik *et al.*'s work because they have analyzed real network traces and modeled the temporal dependence of packet losses in these traces as Markov processes. They have measured and analyzed 128 hours of unicast and multicast traffic. Of these 128 hours, 76 hours were found to be stationary and were used for further analysis. Two methods of analysis were used, the first being based on the auto correlation present in the binary trace data. The lag was varied in the experiment and if the corresponding correlation was greater than a statistical measure, the data was considered correlated for that lag. In the results it was shown that the network data can be correlated up to lags of 1 sec. The second analysis method is based on runlength coding the binary data and analyzing the histogram of the runs of various length and analyzing the correlation between the good and bad runs.

Three models were proposed, the Bernoulli model, the 2-state Markov-chain model and the kth order Markov chain model. Out of the 38 trace segments considered, the Bernoulli model was found to be accurate for 7 segments, the 2-state model was found to be accurate for 10 segments and for the rest of the traces higher models, up to the order of 40, were required. The order of the Markov chain process is estimated by calculating the minimum lag beyond which the process is independent, this is similar to the estimation of the correlation time scale. Once the order has been chosen, the probability models can be adapted on the fly for the set of contexts associated with the model.

In this paper we are using traces measured by Yajnik *et al.* Instead of using the correlation measure, we use entropy of the trace for a given model, in order to find the best model. We have simulated both the Markov chain and the tree models and the results show the advantage of using a tree model. The motivation for this work is based on results obtained by Perret in his Master's thesis [82].

We have analyzed two different sets of traces, each of which is about 2 hour long and has been found stationary by the method proposed in [1]. The descriptions of the traces are given in table 5.2.

	Date	Type	Sampling Interval	Destination	Duration	Loss Rate
Trace-25	20Dec97	unicast	20ms	Seattle	2hrs	1.7%
Trace-27	21Dec97	unicast	20ms	Los Angeles	2hrs	3.4%

Table 5.2: Trace Descriptions from [1]

5.2 Preliminary mathematical concepts

We will use this section to define some preliminary concepts which will be used later in the paper. The information source we are considering is a string of binary random variables $\{x_1..x_n\}$. At each time instant i, after having scanned past data $x^i = x_1x_2..x_i$ the modeling problem is to make inferences on the next symbol x_{i+1} by assigning a conditional probability distribution $p(\cdot|c(x^i))$ to it, where context $c(x^i) = f(x_i...x_{i-k_i})$ is a function of the past symbols. In the long run, the modeling goal is to maximize the probability assigned to the entire sequence [41],

$$P(x^n) = \prod_{i=0}^{n-1} p(x_{i+1}|c(x^i)). \tag{5.1}$$

Taking the negative of the logarithm (base 2) of the above equation we see that the goal is to minimize the code length $h(x)$,

$$h(x) = -\log_2(P(x^n)) = -\sum_{i=0}^{n-1} \log_2 p(x_{i+1}|c(x^i)), \tag{5.2}$$

where $h(x)$ is Shannon's entropy [30]. Entropy will be small when there is order in the sequence, i.e., some patterns occur regularly, and large when there is disorder. It is related to data compression in that the number of bits required to store compressed data should ideally be equal to the entropy of the data.

Clearly there are two steps to modeling the source; the first is to find the "best" context $c(x^i)$ for a given x^i and x_{i+1}. The second is to estimate the conditional probability $p(x_{i+1} = a|c(x^i) = y_j)$, where $y_j \in \mathcal{S}$ and $a \in \mathcal{A}$. ($\mathcal{S}$ is the context space and $\mathcal{A}$ is the alphabet set). The probabilities can be estimated by storing the count, $n(a|y_j)$, i.e., the number of times symbol a occurs with context y_j. Each time x_{i+1} follows a given context $c(x^i)$, $n(x_{i+1}|c(x^i))$ is increased by one. The probability is estimated by

$$p(x_{i+1} = a|c(x^i) = y_j) = \frac{n(a|y_j)}{\sum_{b\in\mathcal{A}} n(b|y_j)}. \tag{5.3}$$

Other methods including those developed by Yajnik et al [1] could be used for online probability estimation.

The more difficult task is to find the best set of contexts $\{c(x^i)\}$ for the given source. In the kth order Markov chain model developed by Yajnik *et al.*, the context is defined as $c(x^i) = (x_i...x_{i-k})$, i.e., the memory is constant for all i and independent of x_{i+1}. Thus, once the order k has been defined, based on a correlation measure, the problem reduces to estimating the probability models.

In the Markov tree model proposed here, we build a tree of all possible contexts $c(x^i) = (x_i..x_{i-k_i})$, $k_i = 0...k$. For each of these contexts the probabilities are estimated using equation (5.3) and for each i and x_{i+1} the best context $c(x^i)$ is found such that the chosen set of contexts, minimize the code length in equation (5.2). In effect at each instant i the context $c(x^i)$ is

$$c(x^i) \triangleq Best\ \{\Phi, (x_i), (x_i x_{i-1}) .... (x_i ... x_{i-k})\} \tag{5.4}$$

where *Best* is defined in terms of code length. The algorithm to do so is presented in the next section.

In the pruned tree model proposed in this chapter, the operation in equation (5.4) is performed over a training set and set of best contexts extracted. These are then used to model the observed data. As in the Markov chain model, the probabilities associated with the contexts can be learned from the data.

5.3 Introduction to the universal coding algorithm

The Context Algorithm, introduced in [25], provides a practical means to search for the best set of contexts for a Markovian process. The algorithm has two integrated steps, the first for growing a tree, where each node of the tree represents a context $c(x^i)$ and the second for selecting from the tree the best context to encode x_{i+1} given x^i. This is a truly adaptive modeling structure: it learns and models the source at the same time. The algorithm as modified by Furlan [38] is presented below.

1. *Initialization*:

   Start with a tree consisting of a single root node, which has a counter for each of the symbols (0,1), set to zero. Read the first symbol x_1, encode this symbol as described in step 3 and increment the count the count of this symbol by 1. Denote the resulting tree by T(1) which is still a 1 node tree.

2. *Choice of the coding node:*
   Each node of the tree has a list of 2 symbol counts. Recursively, let $T(i)$ be constructed after i symbols have been processed. Climb the tree starting at the root according to the path defined by $x_i, x_{i-1}....$ The node which has the lowest entropy is selected, this can be done by using an efficiency counter introduced in [38]. Let the node be referred to as y_j where j denotes the jth level of the tree.

3. *Probability Estimation:*
   Estimate the probability of x_{i+1} using the counts stored at the y_j node. If instead of modeling, we were doing prediction, then this node could be used to predict x_{i+1}.

4. *Update of the tree:*
   Climb the tree, starting at the root, according to the path defined by $x_i, x_{i-1}....$ Then for each node, y_j, visited increment by 1 the count corresponding to the symbol x_{i+1}.

5. *Growing the tree:*
   Continue traversing the tree till a node is reached whose count for symbol x_{i+1} is one before the update. If the node is internal, update the count of its children by one. If its not internal, split the node, initialize its symbol counts to zero except for the symbol x_{i+1} whose count is set to 1.

Fig. 5.1 provides an example where we run the algorithm for sequence 10001. At the first step, the symbol 1 is coded using the memoryless context, then the root node is split and the counters updated for $x_1 = 1$. The second symbol $x_2 = 0$ can be coded using conditional probability $p(0|\Phi)$ or using probability $p(0|1)$, i.e., the comparison is between the coding efficiency of root node and its child, the node corresponding to context (1). This comparison can be efficiently handled by the relative efficiency counter [38] using step 2. After selecting the node (context) to be used to encode symbol the tree is updated and grown using steps 4 and 5. The internal node condition of step 5 is used to update the counts in $T(1)$ which leads to $T(2)$.

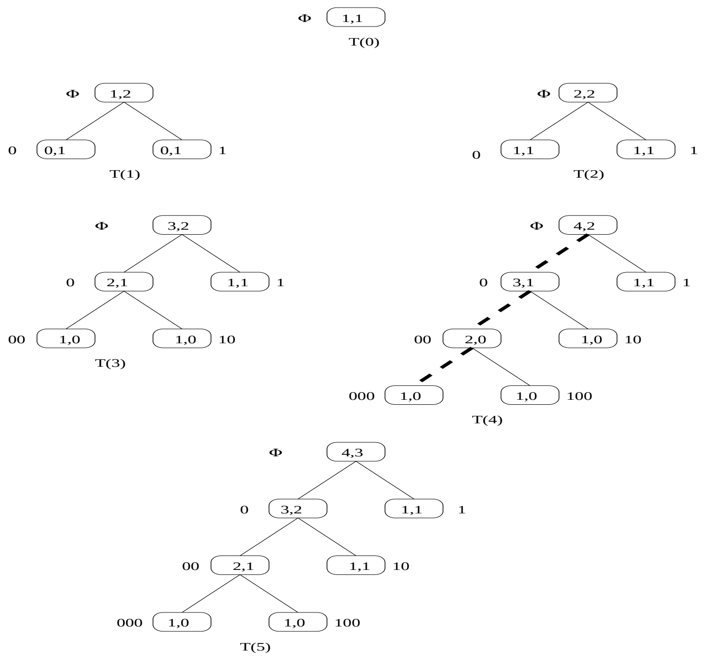


Figure 5.1: The tree coding algorithm for the sequence 10001. The numbers within each node are counts for the number of times a node occurs with the associated symbol. Each node is labeled with the context $(x_i..x_{i-k_i})$ associated with it, the root node is the memoryless context. The first step of the algorithm is to find the best contexts, in terms of coding efficiency, for the given past symbols and the present symbol to be encoded. For example at tree $T(4)$ the fifth symbol 1 has to be encoded and the past symbols are 1000. The algorithm compares the conditional probabilities $p(1|\Phi)$, $p(1|0)$, $p(1|00)$ and $p(1|000)$, i.e., it compares the coding efficiency of the nodes along the dotted line. Father child comparisons are made for all nodes lying on the path and as soon as the father node wins the best node (context) is found. After finding this best context the tree is grown with the rules in steps 4 and 5. The count for symbol 1 is updated along all the nodes of the path till a node which has a count of 1 is reached. This is the (00) node, as it is internal node, from step 5 the counts of children are updated. If it was not an internal node, it would have been split to form new children nodes.

A fully grown tree for the network trace file trace-27 is shown in Fig. 5.2. The leaves of this tree form a order 3 Markov chain model. The disadvantages of using this model have already been discussed in the introduction. Let us take another example, if the sequence ...1100 has occurred, then to code the last 0, the Markov chain will use the node corresponding to context 110, i.e., node $n(10)$. The universal coding algorithm on the other hand will compare costs of coding the last 0, using three different contexts, 0, 10, and 110. Of these three contexts, whichever has the lowest cost will be selected. From the counts in the figure, the most efficient context is 0. This implies that the pattern 00 occurs much more regularly than the pattern 1100. This is the cause of inefficiency of a Markov chain, it is forced to use contexts which may not be occurring regularly enough in the source and thus not giving enough information about the process. The Markov tree model chooses the most efficient contexts and in effect deletes (or lumps) contexts of the Markov chain model.

In Fig. 5.2 the number of times a node (context) is used to code a symbol is shown alongside the node. From the usage count itself, it is clear that the contexts $\Phi, 0, 00, 000, 1, 01$ will be more efficient to code this trace file than the contexts $000, 001, ...111$ which are associated with the Markov chain model.

In the next subsection we give results which compare the modeling performance of a Markov chain model and a fully grown Markov tree model. A full tree of depth k will have $\sum_{i=0}^{k} 2^i$ contexts as compared to the 2^k contexts in the Markov chain, so the comparison may not be fair. However, the Markov tree model gives the lower bound on the best achievable entropy (within the modeling constraints). The results also make clear why nodes besides the ones used in Markov chain models are useful for modeling the source.

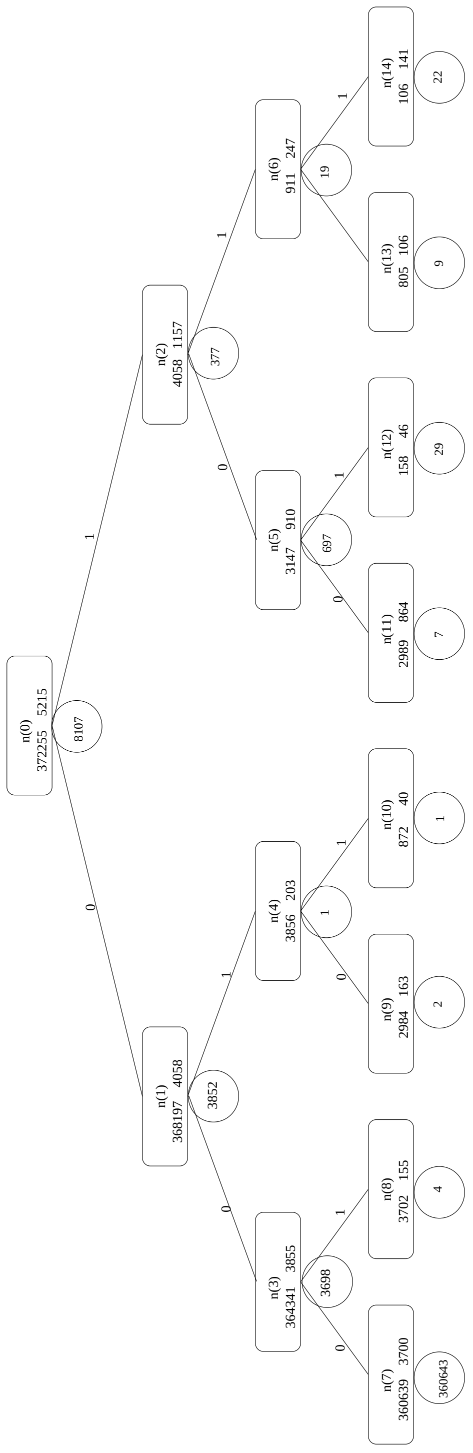


Figure 5.2: The full tree generated for the trace-27 file. The counts for each time a node (context) occurs is given within the node and can be used to calculate the conditional probability using equation (5.3). Also, the number in the circle outside each node, represents the number of times a node is used to code a symbol.

5.3.1 Results

In Fig. 5.3 we present the results for entropy coding of trace-25 and trace-27 using a Markov chain model. Beyond an order of 3 there is no change in the entropy, this is due to context dilution. This figure shows that if Markov chains were to be used to analyze or model the network traffic, high order Markov chains will not improve the modeling performance.

In Figure 5.4, we present entropy results for tree coding algorithm for trace-25 and trace-27. We see that the minimum entropy for trace-25 occurs for a tree with a depth of around 35. Further, notice that using these models the entropy of source reduces by at least 5% over the Markov chain model, which implies that the tree structure is a better model.

The number of times a node is used to code a symbol of the source can also be a good analyzing tool because it will give an indication of the popular patterns (contexts) that occur in the trace. As the number of nodes are very large in a fully grown tree, we group the nodes in a level of the tree and accumulate their counts. In Fig. 5.5, the counts are given for different levels for a tree of depth 45, modeling the trace-27. From the figure, there is a node on level 32 and another on level 42 of the tree which is used very frequently to code the symbol 0. However, for coding symbol 1, only nodes at level 2,3 and 4 are used. In other words, when coding symbol 1 a context of smaller memory than when coding symbol 0 is required. Clearly, this is not possible in a Markov chain model.

From the above discussion it is clear that the tree is unbalanced in the choice of nodes for coding symbols 0 and 1. But, because the counts have been accumulated for a level, Fig. 5.5 does not reveal whether the tree is unbalanced within a level for a symbol, e.g., it is clear that nodes on level 32 are efficient for coding symbol 0, but are all the nodes on level 32 used or is there one node which is used more frequently than others?

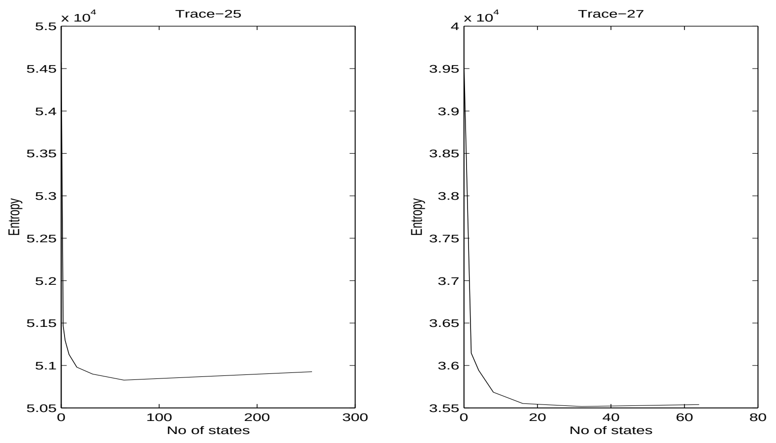


Figure 5.3: Entropy results for Markov-chain modeling of the two traces. It can be seen that the minimum entropy is for chains of order 3, and there is no advantage in increasing the order.

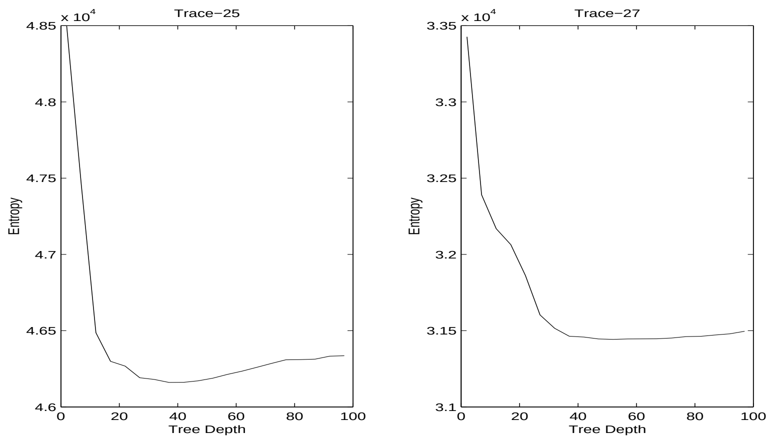


Figure 5.4: Entropy results for the Markov tree modeling. The minimum entropy occurs for a tree depth of around 35 for trace 25. This observation is corroborated by the fact that in [1] the authors have found that this trace has a correlation lag of around 42. For trace 27 the minimum occurs for a tree depth of around 25. The number of contexts in the model are $\sum_{i=0}^{k} 2^i$ where k is the tree depth.

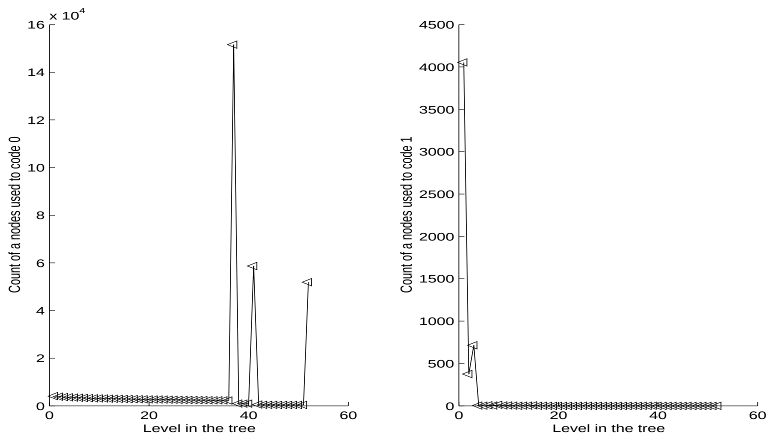


Figure 5.5: The figure analyses the number of times a node is used to code a symbol. We group together nodes on a level of the tree and accumulate their counts. From this figure, there are nodes on level 32 and level 42 which are used very frequently to code the symbol zero. On the other hand, nodes at level 2,3 and 4 are used to code symbol 1. Clearly, the memory of contexts needed to code symbol 0 and symbol 1 is different.

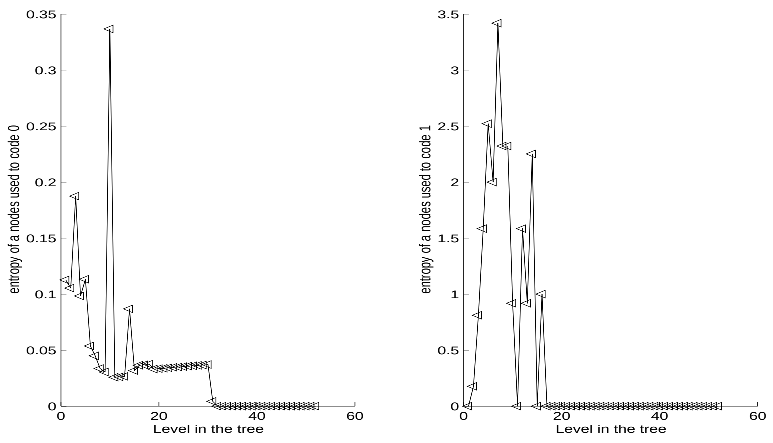


Figure 5.6: The entropy of usage of nodes within a level is being plotted, the probability of using a node to encode a symbol is used to calculate entropy. If the nodes are being equally used, the usage-entropy will be close to one, if only a very small subset of nodes in a level are being used, the usage-entropy will be very small. If the usage-entropy is high it means that nodes are used randomly, this will mainly happen when nodes are not being used regularly enough.

To answer this question, we again use the concept of entropy, the entropy of the probability of using nodes within a level is calculated. If all the nodes are used equally, the usage-entropy is going to be very close to one. The smaller the usage-entropy is it means the smaller the subset of nodes within the level are being used. If the usage-entropy is very large, nodes are being used randomly within the level. In Fig. 5.6, the usage-entropies for coding symbol 0 and 1 are plotted for various levels of a tree of depth 45, for trace-27. From the plot for symbol 0, it is clear that a very small subset of nodes is being used in each level. This is corroborated on analyzing the trace, where we find that only the leftmost node on a level, i.e., the context corresponding to string of zeros is used to code the symbol 0. For symbol 1, the usage-entropy is low till level 4, after which it is very high, this is mainly because nodes on higher level are not being used for coding symbol 1. From the above discussion, the conclusion is that the trees are unbalanced.

5.4 Pruned tree model

In the universal coding algorithm, there is an implicit selection of contexts integrated with the growing (learning of model) of tree. As pointed out, there are some contexts which are used more often than others because of their coding efficiency. A tree pruning algorithm can be used to explicitly choose these contexts and remove the redundant contexts, with pruning based on the efficiency of a node. A training set is used to generate the full tree, the child node which is less efficient than its parent node is pruned. An example of a pruned tree model for the Markov tree of Fig. 5.2 is shown in Fig. 5.7. A modified Context Algorithm can then be used to estimate the probabilities for the observed data, the modifications are that REC is not updated and the pruned tree is not grown. If the number of nodes in the pruned tree are N_{PT}, essentially this model amounts keeping a list of N_{PT} contexts and associated probability models. Based on the past k observations one of the contexts is selected, the probability estimated and then updated.

In Fig. 5.8, the results of coding the trace-27 with the pruned tree model are

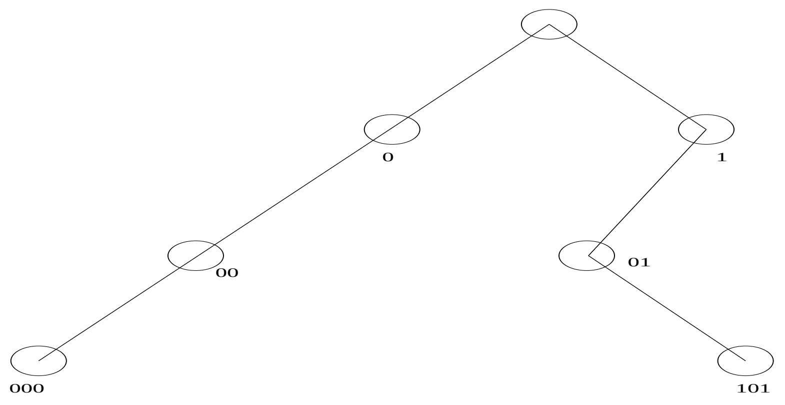


Figure 5.7: The tree in Fig. 5.2 is pruned using the REC counter introduced in section 5.3. Only the most efficient nodes are kept. These connected nodes, and the corresponding conditional probabilities form the best model for the given trace.

given along with the results of coding with a Markov chain model. Markov chain in the figure is adaptive, the probability models are learned while encoding the trace. For Pruned Model-1, the Context Algorithm is run on trace-27. From the full tree, the best set of contexts is pruned and the corresponding counts set to zero. Then, this pruned tree is used to code the trace-27 with probability models being learned on the fly. On the other hand, in Pruned Model-2, Context Algorithm is run on trace-25 and the best set of nodes is pruned from the tree. This best of contexts (nodes) is used to adaptively (probability models are learned on fly) encode trace-27. We can see from the results, that though the pruned tree structure has been derived from a different trace in Pruned Model-2, it still gives better results than an adaptive Markov chain model.

5.5 Conclusion

This chapter shows the limitations of Markov chain models and proposes alternated models based on the universal modeling concepts. The Markov chain model though being very efficient to implement, due to the associated FSM, gives the worst performance among the three models. The Markov tree model on the other hand gives very good results but is difficult to implement due to its large number of contexts. An intermediate solution is the pruned tree model which gives better

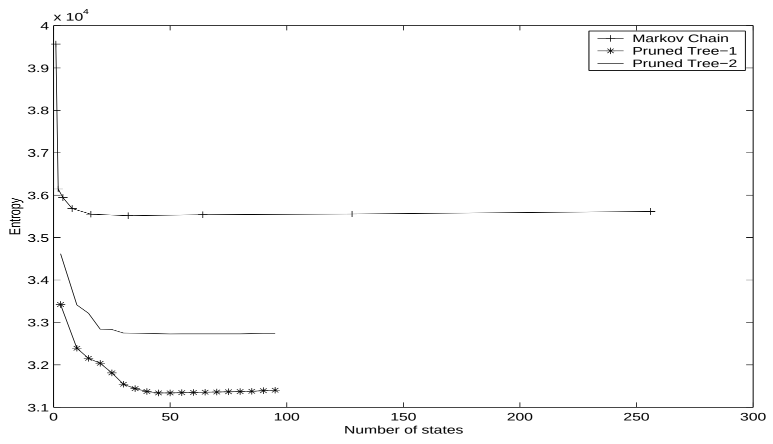


Figure 5.8: Results comparing the pruned tree model with the chain model. Results are for trace-27. All results are adaptive, in that the probability models are learned on the fly. In Pruned Model-1 the set of pruned nodes (best set of contexts) has been derived by running Context Algorithm on trace-27 and pruning the resultant tree. In Pruned Model-2, the set of contexts has been derived from trace-25, i.e., Context Algorithm is run on trace-25 and the best set of contexts are extracted. Pruned Tree-1's performance is very close to the Markov tree model's performance.

result than the chain model using similar number of contexts. However, there may not be a FSM associated with the tree model, to find the next context, the past k (at most) observations have to be scanned. This makes it a little more complex than the Markov chain model. This is summarized in table 5.3.

Model	Complexity	Performance
Markov Chain	Low	Bad
Markov Tree	High	Good
Pruned Tree	Intermediate	Intermediate

Table 5.3: Summary of the comparison between different models

We have developed rules for associating a FSM with the pruned tree model, i.e. given a set of contexts, states are designed so as to uniquely define the transition rule. However, this process is right now offline, as noted by Yajnik *et al.* online adaption of the model can help for multimedia applications. We propose this as part of future research in this area.

References

[1] M. Yajnik, S. Moon, J. Kurose, and D. Towsley, "Measurement and modelling of the temporal dependence in packet loss," in *Proc. of INFOCOM*, 1999.

[2] L. Ke and M.W. Marcellin, "Near lossless image compression: Minimum entropy, constrained error DPCM," *IEEE Trans. on Image Processing*, vol. 7, no. 2, pp. 225–228, Feb. 1998.

[3] C. Chrysafis and A. Ortega, "Line based, reduced memory, wavelet image compression," *IEEE Trans. on Image Processing*, vol. 9, no. 3, pp. 378–389, March 2000.

[4] W. Jiang and A. Ortega, "Multiple description coding via polyphase transform and selective quantization," in *Proc. of VCIP (SPIE)*, 1999.

[5] V.A. Vaishampayan and S. John, "Interframe balanced multiple description video compression," in *Proc. of PVW*, 1999.

[6] "Cyber atlas," http:cyberatlas.intenet.com.

[7] ISO/IEC International Standard 11544, "Progressive bi-level image compression," 1993.

[8] ISO/IEC JTC1/SC2/WG8 CCITT SGVIII, "Joint photgraphic experts group (jpeg)," 1990.

[9] ISO/IEC JTC1/SC2/WG11, "Motion pictures experts group (mpeg)," 1992.

[10] A. Glavieux, C. Berrou, and P. Thitimajishima, "Near shannon limit error-correcting coding and decoding: Turbo-codes," in *Proc. of ICC*, 1993.

[11] J. Hagenauer, "Rate compatible punctured convolutional codes (rcpc codes) and their applications," *IEEE Trans. on Information Theory*, vol. 36, pp. 389–400, April 1988.

[12] L. Rizzo and L. Vicisano, "Effective erasure codes for reliable computer communication protocols," *ACM Computer Communication Review*, vol. 27, pp. 24–36, April 1997.

[13] "Transport control protocol," RFC 793, 1981.

[14] N.S. Jayant and P. Noll, *Digital Coding of Waveforms*, Prentice Hall, 1984.

[15] S.G. Chang, B. Yu, and M. Vetterli, "Spatially adaptive wavelet thresholding with context modelling for image denoising," *IEEE Trans. on Image Processing*, vol. 9, pp. 1522–1531, Sept. 2000.

[16] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer Academic Pub., 1992.

[17] W-W. Chang and J.D. Gibson, "Smoothed DPCM codes," *IEEE Trans. on Communications*, vol. 39, pp. 1351 –1359, September 1991.

[18] P.W. Wong, "Entropy-constrained halftoning using multipath tree coding," *IEEE Trans. on Image Processing*, vol. 6, no. 11, pp. 1567–1579, Nov. 1997.

[19] G.D. Forney, "Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference," *IEEE Trans. Inform. Theory*, vol. 18, pp. 363–378, May 1972.

[20] S. Lin and J.D.J Costello, *Error control coding: Fundamentals and applications*, Prentice-Hall, 1983.

[21] K. Sayood and J.C. Borkenhagen, "Use of residual redundancy in the design of joint source/channel coders," *IEEE Trans. on Communications*, vol. 39, pp. 838–846, June 1991.

[22] D. Miller and M. Park, "A sequence based approximate mmse decoder for source coding over noisy channels using discrete hidden markov models," *IEEE Trans. on Communications*, vol. 46, pp. 222–231, Feb. 1998.

[23] A. A. El Gamal and T. M. Cover, "Achievable rates for multiple descriptions," *IEEE Trans. on Information Theory*, vol. 28, pp. 851–857, Nov. 1982.

[24] V.K. Goyal, "Multiple description coding: Compression meets ths network," *IEEE Signal Processing Magazine*, vol. 18, pp. 74–93, Sept. 2001.

[25] J. Rissanen, "A universal data compression system," *IEEE Trans. on Information theory*, pp. 656–664, 1983.

[26] R. Raheli, A. Polydoros, and C. Tzou, "Per-survivor processing: A general approach to MLSE in uncertain environments," *IEEE Trans. on Communication*, vol. 43, pp. 354–364, 1991.

[27] M.W. Marcellin, T. Fischer, and J.D. Gibson, "Predictive trellis coded quantization of speech," *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. 38, no. 1, pp. 46–55, Jan. 1990.

[28] K.Ramchandran, A.Ortega, and M.Vetterli, "Bit allocation for dependent quantization with applications to multiresolution and MPEG video coders," *IEEE Trans. on Image Processing*, vol. 3, no. 5, pp. 533–545, September 1994.

[29] B.K. Natarajan, "Filtering random noise from deterministic signals via data compression," *IEEE Trans. on Signal Processing*, vol. 43, pp. 2595–2605, Nov. 1995.

[30] C.E. Shannon, "A mathematical theory of communication," *Bell System Tech. Journal*, vol. 27, pp. 379–423, 1948.

[31] N. Farvardin, "A study of vector quantization for noisy channels," *IEEE Trans. on Information Theory*, vol. 26, pp. 799–809, July 1990.

[32] K. Ramchandran and M. Vetterli, *Multiresolution Joint Source Channel Coding*, chapter Wireless Communications, Signal Processing Perspectives, Prentice Hall Signal Processing Series, 1997.

[33] J. Albanese, A. Blomer, J. Edmonds, M. Luby, and M. Sudan, "Priority encoded transmission," *IEEE Trans. on Information Theory*, vol. 42, pp. 1737–1744, Nov. 1996.

[34] C. Papadopoulos and G.M. Parulkar, "Retransmission based error control for continuous media applications," in *Proc. of NOSSDAV*, 1996.

[35] S. McCanne, M. Vetterli, and V. Jacobson, "Low complexity video coding for receiver driven layered multicast," *IEEE Journals on Selected Areas in Communications*, pp. 983–1001, Aug. 1997.

[36] M.J. Weinberger, J.J. Rissanen, and M. Feder, "A universal finite memory source," *IEEE Trans. on Information theory*, pp. 643–652, 1995.

[37] A. Barron, J. Rissanen, and B. Yu, "The minimum description length principle in coding and modeling," *IEEE Trans. on Information Theory*, pp. 2743–2760, Oct. 1998.

[38] G. Furlan, "An enhancement to universal modeling algorithm context for real time applications to image compression," in *Proc. of ICAASP*, 1991.

[39] G.D. Forney, "The Viterbi Algorithm," *Proceedings of the IEEE*, vol. 61, pp. 268–278, Mar. 1973.

[40] M. W. Marcellin and T. R. Fischer, "Trellis coded quantization of memoryless and Gauss-Markov sources," *IEEE Trans. on Communications*, vol. 38, no. 1, pp. 82–93, Jan. 1990.

[41] M.J. Weinberger, J.J. Rissanen, and R. B. Arps, "Applications of universal context modeling to lossless compression of gray-scale images," *IEEE Trans. on Image Processing*, vol. 5, Apr. 1996.

[42] K.M. Chugg, X. Chen, A.Ortega, and C. Chang, "An iterative algorithm for two dimensional digital least metric problems with applications to digital image compression," in *Proceedings of ICIP*, 1998.

[43] P. Thiennviboon, A. Ortega, and K.M. Chugg, "Simplified grid message-passing algorithm with application digital image halftoning," in *Proc. of ICIP*, 2001.

[44] R. Singh and A. Ortega, "Lookahead search for lossy context-based adaptive entropy coding," in *Proc. of ICIP*, 2000.

[45] B. Martins and S. Forchhamer, "Lossless, near lossless, and refinement coding of bi-level images," *IEEE Trans. on Image Processing*, vol. 8, pp. 601–613, May 1999.

[46] K. Ramchandran and M. Vetterli, "Rate-distortion optimal fast thresholding with complete JPEG/MPEG decoder compatibility," *IEEE Trans. on Information Processing*, vol. 3, Sept. 1994.

[47] S.G. Chang, B. Yu, and M. Vetterli, "Image denoising via lossy compression and wavelet thresholding," *IEEE Trans. on Image Processing*, vol. 9, pp. 1532–1546, Sept. 2000.

[48] J. Liu and P. Moulin, "Complexity-regularized image denoising," *IEEE Trans. on Image Processing*, vol. 10, no. 6, pp. 841–851, June 2001.

[49] V.A. Vaishampayan, "Design of multiple description scalar quantizers," *IEEE Trans. on Information Theory*, vol. 39, pp. 821–834, Nov. 1993.

[50] R. Rejaie, M. Handley, and D. Estrin, "Quality adaptation for congestion controlled video playback over the internet," in *Proc. of ACM SIGCOMM*, 1999.

[51] Y. Wang and Q.F. Zhu, "Error control and concealment in video communications- a review," *Proceedings of IEEE*, vol. 86, pp. 974–997, May 1998.

[52] A.R. Reibman, H. Jafarkhani, M.T. Orchard, and Y. Wang, "Performance of multiple description coders on a real channel," in *Proc. of ICIP*, 1998.

[53] N. Shacham and P. King, "Architectures and performance of multichannel multihop packet radio networks," *IEEE Journal Selected Areas of Communication*, vol. 5, July 1987.

[54] N.F. Maxemchuk, *Disparity Routing in Store and Forward Networks*, Ph.D. thesis, University of Pennsylvania, 1975.

[55] V.K. Goyal, J. Kovacevic, R. Arean, and M. Vetterli, "Multiple description transform coding of images," in *Proc. of ICIP*, 1998.

[56] R. Puri and K. Ramchandran, "Multiple description source coding using forward error correction codes," in *Proc. of 33rd Asilomar Conference on Signals and Systems*, 1999.

[57] S. L. Regunathan and K. Rose, "Efficient prediction in multiple description video coding," in *Proc. of ICIP*, 2000.

[58] J.K. Wolf, A.D. Wyner, and J. Ziv, "Source coding with multiple descriptions," *Bell System Technical Journal*, vol. 59, pp. 1417–1427, Oct. 1980.

[59] L. Ozarow, "On a source-coding problem with two channels and three receivers," *Bell System Technical Journal*, vol. 59, pp. 1909–1921, Dec. 1980.

[60] Y. Wang, M.T. Orchard, and A.R. Reibman, "Optimal pairwise correlating transforms for multiple description coding," in *Proc. of ICIP*, 1998.

[61] V.K. Goyal, J. Kovacevic, and M. Vetterli, "Multiple description transform coding: Robustness to erasures using tight frame expansions," in *Proc. of ISIT*, 1998.

[62] P.A. Chou, S. Mehrotra, and A. Wang, "Multiple description decoding of overcomplete expansions using projections onto convex sets," in *Proc. of DCC*, 1999.

[63] A.E. Mohr, E.A. Riskin, and R.E. Ladner, "Generalized multiple description through unequal loss protection," in *Proc. of ICIP*, 1999.

[64] P. Sagetong and A. Ortega, "Optimal bit allocation for channel adaptive multiple description coding," in *Proc. of ICVP(SPIE)*, 2000.

[65] T. Turletti, S.F. Parisis, and J-C. Bolot, "Experiments with a layered transmission scheme over the internet," Tech. Rep., INRIA Research Report No 3296, 1997.

[66] "Vinnt project," http://netweb.usc.edu/vint.

[67] D. Comas, R. Singh, and A. Ortega, "Rate-distortion optimization in a robust video transmission based on unbalanced multiple description coding," in *Proc. of MMSP*, 2001.

[68] "User datagram protocol," RFC 768, 1980.

[69] M. Podolsky, M. Vetterli, and S. McCanne, "Limited retransmission of real-time layered multimedia," in *Proc. of MMSP*, 1998.

[70] W.M. Lam and A.R. Reibman, "Self-synchronizing variable-length codes for image transmission," in *Proc. of ICASSP*, 1992.

[71] V. Hardman, M.A. Sasse, M. Handley, and A. Watson, "Reliable audio for use over the internet," in *Proc. of INET*, 1995.

[72] A.R. Reibman, H. Jafarkhani, Y. Wang, M. Orchard, and R. Puri, "Multiple description coding for video using motion compensated prediction," in *Proc. of ICIP*, 1999.

[73] N. Demir and K. Sayood, "Joint source channel decoding of variable length codes," in *Proc. of DCC*, 1999.

[74] H. Yao and V.A. Vaishampayan, "A Viterbi based decoding algorithm for multiple description variable length codes," in *Proc. of ICASSP*, 2000.

[75] R. Singh and A. Ortega, "Erasure recovery in predictive coding environments using multiple description coding," in *Proc. of MMSP*, 1999.

[76] N.S. Jayant, "Adaptive quantization with a one word memory," *Bell Systems Tech. Journal*, pp. 1119–1144, Sept. 1973.

[77] C.-Y. Hsu, A. Ortega, and M. Khansari, "Rate control for robust video transmission over burst-error wireless channels," *IEEE Journal on Selected Areas in Communications, Special Issue on Multimedia Network Radios*, vol. 17, pp. 756–773, May 1999.

[78] Z. Miao and A. Ortega, "Optimal scheduling for streaming of scalable media," in *Proc. of 34th Asilomar Conference on Signals and Systems*, 2000.

[79] J. Bolot and T. Turletti, "Adaptive error control for packet video in the internet," in *Proc. of ICIP*, 1996.

[80] V. Paxson, "End to end internet packet dynamics," *IEEE Trans. on Networking*, pp. 277–292, 1999.

[81] J. Bolot, "End-to-end packet delay and loss behaviour in the internet," in *Proc. of SIGCOMM*, 1993, pp. 289–298.

[82] L. Perret, "Simulation and modeling of internet packet losses with applications to video transmission," Travail de Diplome, Section Electricite, EPFL, Lausanne, Suisse, 1999.