PERFORMANCE ANALYSIS OF CONCATENATED CODING SCHEMES

A THESIS
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING
AND THE INSTITUTE OF ENGINEERING AND SCIENCES
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
Gün Akkor
June 1999

PERFORMANCE ANALYSIS OF CONCATENATED CODING SCHEMES

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCES

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

Gün Akkor

By

Gün Akkor

June 1999

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Dr. Erdal Arıkan(Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Melek Yücel

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assist. Prof. Dr. Murat Alanyalı

Approved for the Institute of Engineering and Sciences:

Prof. Dr. Mehmet Baray
Director of Institute of Engineering and Sciences

ABSTRACT

PERFORMANCE ANALYSIS OF CONCATENATED CODING SCHEMES

Gün Akkor
M.S. in Electrical and Electronics Engineering
Supervisor: Prof. Dr. Erdal Arıkan
June 1999

In this thesis we concentrate on finding tight upperbounds on the output error rate of concatenated coding systems with binary convolutional inner codes and Reed-Solomon outer codes. Performance of such a system can be estimated by first calculating the error rate of the inner code and then by evaluating the outer code performance. Two new methods are proposed to improve the classical union bound on convolutional codes. The methods provide better error estimates in the low signal-to-noise ratio (SNR) region where the union bound increases abruptly. An ideally-interleaved system performance is evaluated based on the convolutional code bit error rate estimates. Results show that having better estimates for the inner code performance improves the estimates on the overall system performance. For the analysis of a non-interleaved system, a new model based on a Markov Chain representation of the system is proposed. For this purpose, distribution of errors between the inner and outer decoding stages is obtained through simulation. Markov Chain parameters are determined from the error distribution and output error rate is obtained by analyzing the behavior of the model. The model estimates the actual behavior over a considerable SNR range. Extensive computer simulations are run to evaluate the accuracy of these methods.

Keywords: concatenated coding, performance analysis, union bound, non-interleaved systems.

ÖZET

Gün Akkor
Elektrik ve Elektronik Mühendisliği Bölümü Yüksek Lisans
Tez Yöneticisi: Prof. Dr. Erdal Arıkan
Haziran 1999

Bu tezde konvolusyonel iç kodlama ve Reed-Solomon dış kodlama kullanan kademeli kodlama şekillerinin sistem çıkışındaki hata olasılıkları için üst sınır bulunması üzerinde durulmuştur. Bu tür sistemlerin performansı iç kodun hata olasılığının hesaplanması ve bunun dış kodun performansının hesaplanmasında kullanılması yolu ile tahmin edilebilir. Konvolusyonel kodların hata olasılıkları üzerindeki klasik üst sınırların iyileştirilmesi için iki yeni yöntem önerilmiştir. Bu yöntemler özellikle düşük sinyal-gürültü oranlarında hata olasılıklarını daha iyi tahmin etmektedir. Konvolusyonel kodların hata olasılık tahminleri kullanılarak bloklar arasındaki sembollerin ideal olarak düzenlendiği (interleaved) kademeli kodlama sisteminin performansı incelenmiştir. Sonuçlar iç kodlama hata olasılığı üzerinde elde edilen daha iyi tahminlerin, tüm sistemin hata olasılığı üzerindeki tahminleri de iyileştirdiğini göstermiştir. Bloklar arasındaki sembollerin düzenlenmediği (non-interleaved) kademeli kodlama sistemlerinin performansının incelenmesi için bir Markov Chain modeli önerilmiştir. Simülasyon yolu ile iç ve dış kodlama blokları arasındaki hata dağılımı elde edilmiştir. Bu dağılımdan yararlanılarak Markov modelinin parametreleri elde edilmiştir. Sistem çıkışındaki hata olasılıkları modelin davranışları incelenerek elde edilmiştir. Model, geniş bir sinyal-gürültü oranı için sistem performansını doğru tahmin etmektedir. Metodların doğruluğunun kıyaslanması için tüm incelenen sistemlerin hata olasılıkları simülasyon yolu ile de elde edilmiştir.

Anahtar Kelimeler: kademeli kodlama, performans analizi, hata olasılıkları için üst sınırlar.

ACKNOWLEDGMENTS

I would like to express my deep gratitude to my supervisor Prof. Dr. Erdal Arıkan for his guidance, suggestions and valuable encouragement throughout the development of this thesis.

I would like to thank Assoc. Prof. Dr. Melek Yücel and Assist. Prof. Dr. Murat Alanyalı for reading and commenting on this thesis and for the honor they gave me by presiding the jury.

I am also grateful to my friends, Arçın Bozkurt, Ayhan Bozkurt, Deniz Gürkan, Tolga Kartaloğlu, and Güçlü Köprülü, who made life at Bilkent more enjoyable during the course of this thesis work. I would like to thank everyone who in one way or the other helped me to complete my thesis.

Sincere thanks to Assoc. Prof. Dr. Irşadi Aksun, for our friendly conversations and his motivation during the very last moments of this thesis work.

Duygu deserves the right to be mentioned separately for sharing all my time with me. Thank you for your encouragement, support, and understanding but above all for being with me.

My gratitude is far less than they deserve for my wonderful parents. Without their love, endless support, and understanding nothing would be possible.

Contents

List of Figures

List of Tables

Glossary

AWGN : Additive White Gaussian Noise

BDD : Bounded Distance Decoder

BER : Bit Error Rate

BSC : Binary Symmetric Channel

BPSK : Binary Phase Shift Keying

MC : Markov Chain

ML : Maximum-likelihood

RS : Reed-Solomon

SNR : Signal-to-noise Ratio

$\mathbf{d}$: Hamming distance

$\mathbf{k}$: Number of information symbols in a codeword

$\mathbf{K}$: Constraint length

$\mathbf{m}$: Number of bits in a RS symbol

$\mathbf{n}$: Codeword length

$\mathbf{P_b}$: Probability of bit error

$\mathbf{P_c}$: Probability of correct decoding

$\mathbf{P_e}$: Error event probability

$\mathbf{P_E}$: First error event probability

$\mathbf{P_s}$: Probability of symbol error

$\mathbf{P_w}$: Probability of word error

$\mathbf{R}$: Rate of a code

$\mathbf{T(D)}$: Path enumerator of a convolutional code

$\mathbf{t}$: Number of correctible RS symbol errors

$\mathbf{v}$: Memory of a convolutional code

To my parents ...

Chapter 1

Introduction

Error control coding is an area of increasing importance in communication. This is partly because the data integrity is becoming increasingly important. There is downward pressure on the allowable error rates for communications and mass storage systems as bandwidths and volumes of data increase. In any system which handles large amount of data, uncorrected or undetected errors can degrade performance.

Just as important as the data integrity issue is the increasing realization that error control coding is a system design technique that can fundamentally change the trade-offs in a communications system design. Even if a communications system has no problem with the data quality, designers may consider discarding or downgrading the most expensive or troublesome elements while using the error control coding techniques to overcome the resulting loss in performance.

An oversimplified structure of a communications system is shown in Fig. 1.1. Data symbols generated by the source are encoded and then transmitted through the channel. Due to presence of noise, some of the symbols are received in error at the other end. Decoder processes the received data in an attempt to minimize the number of symbols in error. A natural measure of data integrity at the system output is the *probability of symbol error*, P_s, which is the ratio of symbols in error to the overall number of transmitted symbols. Performance of a coding scheme can be evaluated based on the error rate at the output when all other system parameters (channel properties, transmitter power, etc.) are kept constant.

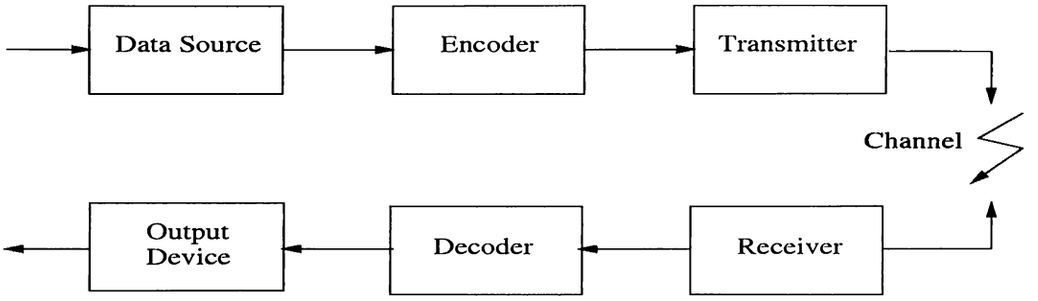

Figure 1.1: A communication system example

In system design, an estimate on the performance of a particular coding scheme is essential to decide on the system parameters. The most useful techniques for estimating the performance of a code are computer simulations and error bounds. The usefulness of computer simulation is limited by the long computational times that are required to get a good statistical sample (it may take several hours to get a single point). Bounds on the error performance, on the other hand, aim to find analytical models to the actual behavior. They require less computational effort, but may not be able to predict the behavior of the code over the full operation range, or may overestimate the actual error rate.

Problem of finding tight bounds on the error rate of different coding schemes is an active research area. The problem of interest in this thesis is to find tight bounds on the error performance of *concatenated coding* schemes. Most of today's system applications require low rates of error. This can be accomplished by imposing codes with longer block lengths and larger error-correcting capabilities. However, encoder/decoder complexity also increases placing a burden on implementation. In concatenated coding, this problem is overcame by utilizing multiple levels of coding where each stage has moderate complexity. In the next section, fundamental concepts of concatenated coding is discussed.

1.1 Fundamental Concepts of Concatenated Coding

When digital data are transmitted over a noisy channel, there is always a chance that the received data will contain errors. The user generally establishes an error rate below which the received data are usable. If the received data will not meet the error rate

requirement, error-correction can often be used to reduce errors to a level at which they can be tolerated. In recent years the use of error-correction coding for solving this type of problem has become widespread.

The utility of coding was first demonstrated by the appearance of Shannon's classic paper [1]. In 1948 he proved that as long as the data source rate is less than the *channel capacity*, communication over a noise channel with an error probability as small as desired is possible with proper encoding and decoding. Essentially, Shannon's work states that the signal power, channel noise, and available bandwidth set a limit only on the communication rate and not on the accuracy.

It soon became clear that the *real* limit on communication rate was set not by the channel capacity but by the *cost* of implementation of coding schemes. Cost constraints force communication at rates substantially below capacity. In recent years much research has been directed toward finding efficient and practical coding schemes for various types of noisy channel. Concatenated coding, which was first investigated by Forney [2], is a way of achieving low error probabilities using long block length (linear) codes at rates below capacity without requiring an impossibly complex decoder.

The basic idea behind concatenated coding is that the "encoder-channel-decoder" of a normal communication system can be viewed as a kind of *superchannel* for which an outer code can be employed (Figure 1.2). If length, dimension, and minimum Hamming distance for the inner and outer (block) codes are denoted by (n_1, k_1, d_1) and (n_2, k_2, d_2), respectively, it is then well known that the corresponding concatenated code has length $n = n_1 n_2$, dimension $k = k_1 k_2$, and minimum Hamming distance $d \geq d_1 d_2$ with equality if every nonzero codeword in the inner code has constant weight. Concatenation is then a technique for producing long codes from short codes. This property is particularly important for the receiver-side operation. Though shorter codes have weaker error correction capabilities, they have simpler decoding mechanisms. Inner decoding and outer decoding are performed separately on the receiver side allowing the overall decoding complexity to remain moderate. On the other hand, decoding errors made by the inner decoder can be corrected by the outer decoder improving the overall performance.

Though numerous configurations are possible, concatenated coding systems with Reed-Solomon (RS) outer codes and binary convolutional inner codes are popular. Binary convolutional codes work on binary (0 or 1) digits or bits. The encoder of such codes maps a binary input sequence into a binary output sequence. Binary convolutional codes are particularly effective in correcting uniformly distributed bit errors. However, closely

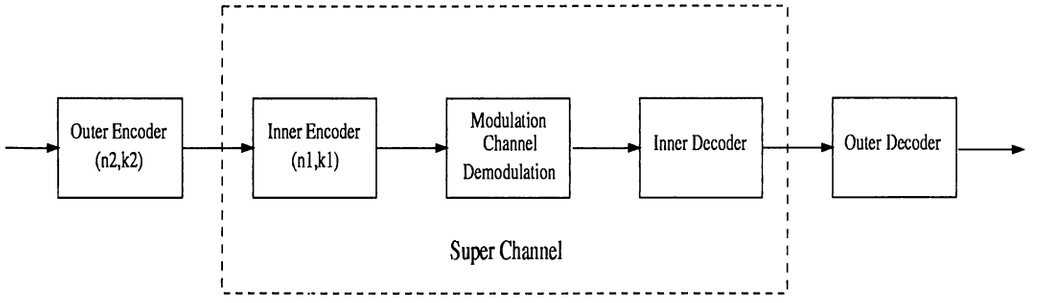

Figure 1.2: Concatenated coding system

spaced bit errors are decoded into an error burst due to presence of decoder memory. A detailed discussion on convolutional codes is presented in Chapter 2.

RS codes work on a higher alphabet of m-bit symbols. They map every k information symbols into a codeword of $n = 2^m - 1$ symbols. Such a code can correct any $t = (n-k)/2$ symbol errors in a n symbol codeword. Considering a bounded-distance-decoder (BDD) which corrects all error patterns of weight t or less and corrects no patterns of weight larger than t, probability of correct decoding for a (n, k) RS code is given by

$$P_c = \sum_{i=0}^{t} \binom{n}{i} P_s^i (1 - P_s)^{n-i} \tag{1.1}$$

where P_s is the probability of symbol error at the RS decoder input. When used in a concatenated scheme, k information symbols are coded into a n-symbol codeword by the RS encoder. Next, the convolutional encoder maps this nm-bit sequence into another binary sequence of nm/R bits where R is the rate of the convolutional code. RS codes are particularly effective in correcting the burst errors caused by the inner decoding if a binary error burst when grouped into symbols affect less than t symbols in a codeword. Properties of RS codes are further discussed in Chapter 4.

In cases where burst errors are a problem, one potential solution involves the utilization of a suitable interleaver/deinterleaver pair between the inner and outer stages. Using this approach, output sequence of outer encoder is interleaved prior to inner encoder at the transmitter side. The output sequence of the inner decoder is then deinterleaved prior to outer decoder at the receiver side. A system block diagram is shown in Fig. 1.3. An *interleaver* is a device that rearranges (or permutes) the ordering of a sequence of symbols in a deterministic manner. Associated with an interleaver is a *deinterleaver* that applies the inverse permutation to restore the sequence to its original ordering. When deinterleaving is applied on the receiver side, the burst errors caused by the inner

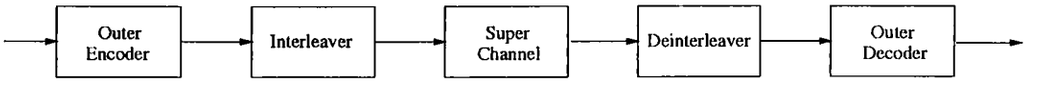

Figure 1.3: An Interleaved System

decoding are distributed more uniformly prior to outer decoder increasing the overall probability of correct decoding.

Concatenated schemes are one of the most effective coding systems. In the next section, we state two basic problems of performance analysis of concatenated systems.

1.2 Problem Statement

In this thesis, we restrict our attention to two-stage concatenated schemes with rate-1/2 binary convolutional inner codes, and RS outer codes. The performance of such a system based on a probability of symbol error measure can be calculated by first estimating the symbol error rate of the inner convolutional code and then by evaluating the error rate of the outer RS code. The probability of symbol error for a convolutional code can be upper bounded by using the well-known union bound and path enumeration techniques. This bound matches exact behavior at high signal-to-noise ratio (SNR) region, but is rather loose as we move to the low SNR region. This region is the target operational range for concatenated schemes. Therefore, if classical union bound is used to evaluate the inner code performance, the resulting bound will be loose in the range of interest. For a better estimate on the overall system performance, the classical union bound should be modified in the low SNR region.

There exists closed form expressions for calculating the probability of error at the RS decoder output based on the probability of error estimates at the input. But such expressions assume that the symbol errors at the decoder input are independent of each other. For the inner decoder operation this might be validated under the assumption that the channel is memoryless. However, convolutional codes have memory, and a single error appearing at the decoder input may propagate and form a burst of errors at the output. Consequently, the symbol errors at the RS decoder input are not independent of each other. The same expressions can be used under the assumption that interleaving/de-interleaving is applied between the stages. However, for non-interleaved cases, a model

that takes the bursty nature of errors into account should be devised.

1.3 Outline of the Work Done

In an attempt to improve the union bound on convolutional codes, we generalized the results of [3] for convolutional codes. In [3] an improved union bound for the binary input AWGN channel in terms of the weights of the codewords of a linear block code is presented. For the convolutional code case, we treated every path that diverges from the all-zero path at the origin and merges back at the jth trellis stage as a codeword of length $n_j = 2j$ for a rate-1/2 code. Paths of length n_j form a set, S_j, of codewords of equal length. The process of partitioning paths into codewords of equal length can go up to infinity, with increasing j. However, the probability that a path will stay diverged from the all-zero path for a large j decreases. Therefore, we limit j to be less than or equal to some constant L_{max}. Every set S_j is then treated as a block code of blocklength n_j for which the arguments of [3] can be applied. The overall probability figure is obtained by summing the individual results. This method is applied to both a binary symmetric channel (BSC) with hard decision decoding and an AWGN channel with soft decision decoding. The method provides promising results. The results show 2 to 10 fold improvement over the classical union bound.

The second approach to improve the union bound on convolutional codes calculates the probability of error as a sum of two terms. If a sequence of weight less than i is received, then we use the path enumerator polynomial to calculate the probability of error of this case. A sequence of weight less than i can be decoded into any path of weight $d_f \leq d \leq 2(i-1)$ by a maximum-likelihood (ML) decoder, where d_f is the free distance of the convolutional code. Therefore, only these terms of the path enumerator polynomial is used in the summation. If a sequence of weight larger than i is received, we assume that the decoder always makes an error and hence in this case the probability of error is equated to the probability of receiving such a path. The overall probability of error is given by the sum of the two cases. The value of i changes the contribution of the two cases. Therefore, we repeat the calculations for i running from $\lceil d_f/2 \rceil$, which is the minimum number of channel errors that may cause the received sequence to be decoded into a wrong path, to $\lceil d_{max}/2 \rceil$ which is the number of channel errors that may cause the received sequence to be decoded into a path that is most far away from the

correct path assuming that we use a truncated code. Then, the minimum is chosen as the error probability. This approach provides improvement in the low SNR region. In the high SNR region the bound tends to converge to the classical union bound.

To deal with the non-interleaved cases, we run extensive computer simulations to obtain burst histograms for the convolutional code. Also, histograms for the distribution of waiting times between the occurrence of successive bursts are obtained. The results are used to set up a two state Markov Chain model; the states representing the events of a symbol being correct or erroneous. The steady state distribution of the Markov Chain is then used to calculate the probability of symbol error for the outer RS code. The model provides almost exact results over a considerable range of SNR values. At high SNR region, the results show deviation from the exact behavior as the assumption of geometrically distributed waiting times does not correctly model the system.

Finally, computer simulations are run for both a BSC with hard decision decoding and an AWGN channel with soft decision decoding. These data are used to evaluate the accuracy of the above methods. The details of all methods and their results are extensively discussed in Chapter 3 and Chapter 5.

1.4 Organization of the Thesis

The organization of the thesis is as follows. In the next chapter, following an overview of convolutional codes, a survey of results on convolutional code performance is presented. Chapter 3 includes a detailed discussion on the work done to improve the union bound at low SNR region and presents the results. In Chapter 4 RS code structure and its properties are presented and an ideally interleaved system performance is evaluated based on the results of Chapter 3. In Chapter 5 work done to model the bursty behavior of symbol errors at the RS decoder input is presented. Based on these results a non-interleaved system is investigated. In the final chapter an overall conclusion is presented and possible future work is discussed.

Chapter 2

Survey of Results on Convolutional Code Performance

In this chapter, an overview of convolutional code structure is presented. Following a discussion on weight distribution of convolutional codes, the classical union bound on probability of error is introduced. Finally, a survey of results on convolutional code performance is presented.

2.1 Binary Rate-1/2 Convolutional Codes

A constraint length K convolutional encoder consists of a K-stage shift register with the outputs of selected stages being added modulo-2 to form the encoded symbols. A simple example is the $R = 1/2$ convolutional encoder shown in Fig. 2.1. Information symbols are shifted in at the left, and for each information symbol the outputs of the modulo-2 adders provide two channel symbols. The connections between the shift register stages and the modulo-2 adders are conveniently described by generator polynomials. The polynomials $g_1(D) = 1 + D + D^2$ and $g_2(D) = 1 + D^2$ represent the upper and lower connections, respectively (the lowest-order coefficients represent the connections to the leftmost stages). The input information sequence can also be represented as a power

series $I(D) = i_0 + i_1 D + i_2 D^2 + \ldots$, where i_j is the information symbol (0 or 1) at the jth symbol time. With this representation the outputs of the convolutional encoder can be described as a polynomial multiplication of the input sequence, $I(D)$, and the code generators. For example, the output of the upper encoded channel sequence of Fig. 2.1 can then be expressed as $T_1(D) = I(D)g_1(D)$, where the polynomial multiplication is carried out in $GF(2)$.

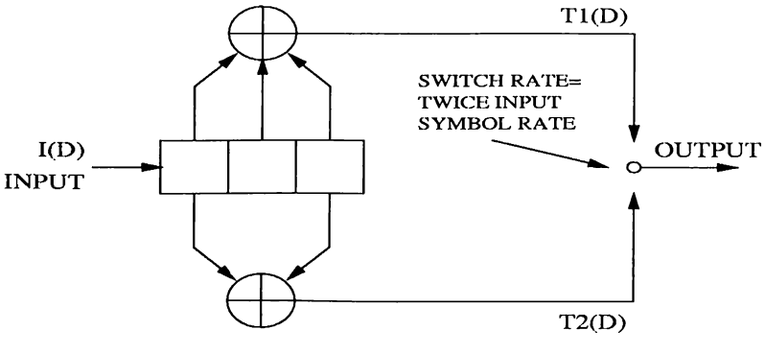

Figure 2.1: Encoder for a $R = 1/2$, $K = 3$ convolutional code

It is possible to view the above convolutional encoder structure as a finite-state machine. The state of a convolutional encoder is determined by the most recent $v = K-1$ information symbols shifted into the encoder shift register. The state is assigned a number from 0 to $2^v - 1$. The current state and the output of the encoder are always uniquely determined by the previous state and current input. A complete description of the encoder as far as the input and the resulting output are concerned can be obtained from a state diagram, as shown in Fig. 2.2 for the $R = 1/2$, $v = 2$ ($K = 3$) code. The symbol along the top of the transition line indicates the information symbol at

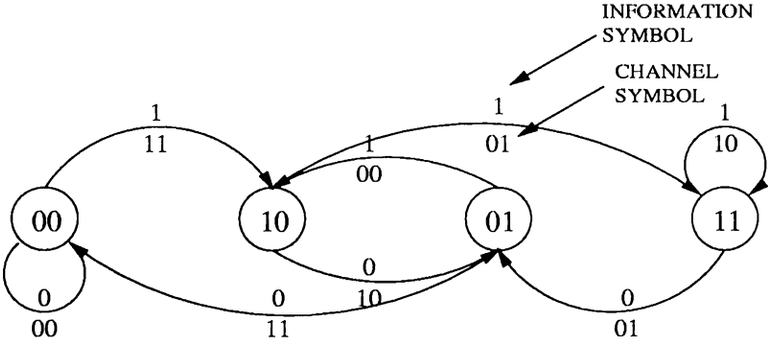

Figure 2.2: State diagram for the $R = 1/2$, $v = 2$ convolutional code

the encoder input that causes the transition and the symbols below the transition line

show the resulting channel symbols at the encoder output. Any sequence of information symbols dictates a path through the state diagram, and the channel symbols encountered along this path constitute the resulting encoded channel sequence.

Because of the topological equivalence between the state diagram of Fig. 2.2 and a signal flow graph [4], the properties and theory of signal flow graphs have been applied to the study of convolutional code structure and performance. The state diagram of a convolutional code can be alternatively viewed as in Fig. 2.3. This structure has been given the name *trellis structure*, and has proved to be useful in performance analysis as well as in the study of decoding algorithms.

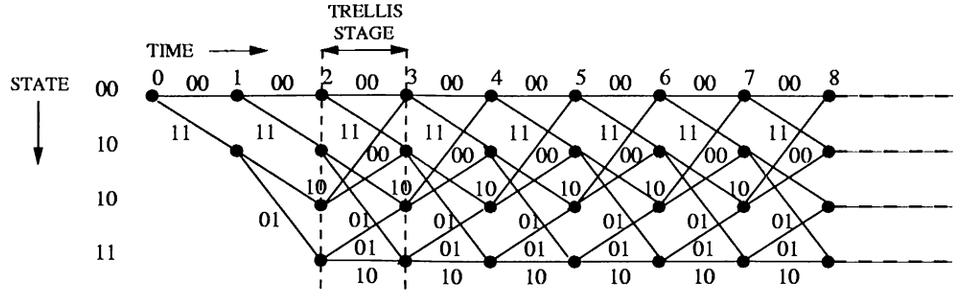

Figure 2.3: Trellis diagram for the $R = 1/2$, $K = 3$ convolutional code

Decoding of convolutional codes can be accomplished by a maximum-likelihood decoding algorithm known as Viterbi algorithm [5]. The Viterbi algorithm operates step by step, tracing through a trellis identical to that used by the encoder in an attempt to emulate the encoder's behavior. At any time the decoder does not know which node the encoder reached and thus does not try to decode this node yet. Instead, given the received sequence, the decoder determines the distance between each such path and the received sequence. This distance is called the *discrepancy* of the path. If all paths in the set of most likely paths begin in the same way, the decoder knows how the decoder began.

Then in the next trellis stage , the decoder determines the most likely path to each of the new nodes of that stage. But to get to any of the new nodes the path must pass through one of the old nodes. One can get the candidate paths to a new node by extending to this new node each of the old paths that can be thus extended. The most likely path is found by adding the incremental discrepancy of each path extension to the discrepancy of the path to the old node. There are 2^v such paths to each node, and the path with the smallest discrepancy is the most likely path to the new node. This process

is repeated for each of the new nodes. At the end of the iteration, the decoder knows the most likely path to the each of the nodes in the new trellis stage.

Consider the set of surviving paths to the set of nodes at the jth trellis stage. One or more of the nodes of the first stage will be crossed by these paths. If all the paths cross through the same node at the first trellis stage, then regardless of which node the encoder visits at the jth stage, decoder knows the most likely node it visited at the first stage. That is, the first information is known even though no decision can be made for the jth stage. To build a Viterbi decoder, one must choose a decoding-window width W, usually several times as the constraint-length. At jth stage, the decoder examines all surviving paths to see if they agree in the first branch. This branch defines a decoded information frame, which is passed out of the decoder. Next the decoder drops the first branch and takes a new frame of the received word for the next iteration. If again all surviving paths pass through the same node of the oldest surviving trellis stage, then this information frame is decoded.

The process continues in this way, decoding frames indefinitely. If W is chosen long enough, then a well-defined decision will almost always be made at each stage. In some cases, the surviving paths might not all go through a common node, or may converge into a wrong node. In such cases a decoder failure or a decoding error occurs. If the errors occur randomly, with enough spacing in between, the decoding algorithm chooses the correct path with high probability. However, if errors occur in bursts, decoder algorithm may fail to decode to the correct path causing an error burst at the output.

The Viterbi algorithm and the trellis structure provides a framework for the calculation of weight distributions and performance bounds using path enumeration techniques.

2.2 Weight Distribution of Convolutional Codes

Weight distribution of a code has particular importance for evaluating the performance of the code. Calculation of the weight distribution of convolutional codes requires the examination of the state diagram structure of Section 2.1. The weight of a convolutional codeword, infinitely long, is the number of nonzero components it has. If the code is simple enough for the trellis to be drawn, all of these parameters can be read from the trellis. For example, consider the $R = 1/2$, $K = 3$ convolutional code which has the

trellis structure shown in Fig. 2.3. Tracing the trellis it is found that there is a path of weight 5, that occupies three stages. This is also the *free distance* of the code. The free distance of a convolutional code is defined as the distance of the shortest path that diverges from the all-zero path at the origin and merges back at a later trellis stage. Looking further, there are two paths of weight 6, four paths of weight 7 and so on. In this way the number of paths of each weight can be enumerated, but this quickly becomes tedious. A more powerful approach uses the state diagram of the code and the theory of signal flow diagrams to determine the complete weight distribution [6].

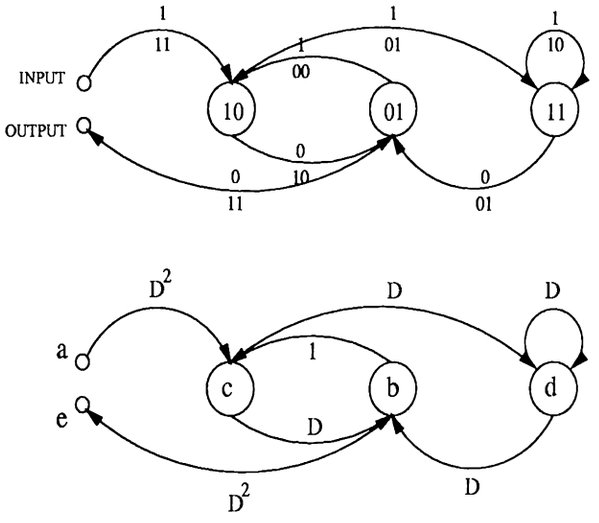

Figure 2.4: State and signal flow diagrams for the $R = 1/2$, $K = 3$ convolutional code

In Fig. 2.4, the state diagram of Section 2.1 is redrawn. Note that the state labeled 00 is drawn twice, once as input and one as output, because we are only interested in paths starting at 00 and ending at 00. Also each branch is attached a gain as power of a dummy variable D. The weight of a branch now appears as the power of D. The weight of a path is obtained by multiplying all the gains along the path. It is possible to compute the gain $T(D)$ of the whole network between the input and the output by simultaneous solution of the following equations obtained from Fig. 2.4:

$$\begin{aligned} b &= Dc + Dd \\ c &= D^2a + b \\ d &= Dc + Dd \\ e &= D^2b. \end{aligned} \tag{2.1}$$

Solution of these equations gives

$$e = \frac{D^5}{1-2D}a. \tag{2.2}$$

Thus the path enumerator is

$$\begin{aligned} T(D) &= \frac{D^5}{1-2D} \\ &= D^5 + 2D^6 + 4D^7 + \ldots + 2^i D^{i+5} + \ldots. \end{aligned} \tag{2.3}$$

Comparing the expression in Eq. 2.3 with the trellis diagram of Fig. 2.3 one can verify that there are actually 2^i paths of weight $i+5$. The same technique can be used to count other properties of the code. Introducing two new dummy variables, L to count the number of trellis stages, and I to count input ones, and solving as before,

$$\begin{aligned} T(D,L,I) &= \frac{D^5L^3I}{1-DL(1+L)I} \\ &= D^5L^3I + D^6L^4(1+L)I^2 + D^7L^5(1+L)^2I^3 \\ &\quad + \ldots + D^{5+i}L^{3+i}(1+L)^iI^{1+i} + \ldots \end{aligned} \tag{2.4}$$

Thus, the path of weight 5 has a length of 3 and is caused by a single input bit equal to one. There are two paths of weight 6, each caused by two input ones; one path has length 5 and one has a length of 6. In this way the distance structure which is needed for bounding the probability of error can be fully determined.

2.3 Union Bound on Probability of Decoding Error

This technique can be used for any block or convolutional code with maximum-likelihood decoding. It is based on the following idea. If an event can be expressed as the union of several subevents, then the probability of that event is less than or equal to the sum of the probabilities of all subevents.

$$P(A) = P(A_0 \bigcup A_1 \ldots \bigcup A_{n-1}) \leq \sum_{i=0}^{n-1} P(A_i) \tag{2.5}$$

This sum is obviously an over-bound since it counts the contribution due to overlapping events more than once.

In the case of a linear block code, the probability of error can be computed by considering the effect of transmitting the all-zero codeword. An error will be made if the received sequence (the term 'sequence' is introduced here because in general a received sequence may not be equal to any of the valid codewords due to errors) is closer to one of the other codewords than it is to the all-zero word. Thus, the probability of error can be over-bounded as the sum of the probabilities of each of these individual error events. Define A_0 as the event of transmitting the all-zero codeword and B_j as the event that distance between the received sequence and some codeword of weight j is smaller than the distance between the received sequence and the all-zero codeword. Using this approach the probability of codeword error for a maximum-likelihood decoder is upper bounded by [7]

$$P_w \leq \sum_{j=1}^{n} c_j P(B_j/A_0), \tag{2.6}$$

where c_j is the number of codewords of weight j. In a similar fashion the average bit error probability is upper bounded by

$$P_b \leq \sum_{j=1}^{n} \frac{\eta_{0j}}{k} c_j P(B_j/A_0), \tag{2.7}$$

where η_{0j} is the average number of nonzero information bits associated with a codeword of weight j and k is the number of information bits in a n bit codeword. The term η_{0j}/k can be very closely approximated by j/n [7]. $P(B_j/A_0)$ for the case of a BSC with hard decision and cross-over probability of p, is given by

$$P_j = P(B_j/A_0) = \begin{cases} \sum_{i=\frac{(j+1)}{2}}^{j} \binom{j}{i} p^i (1-p)^{j-i}, & j \text{ odd} \\ \frac{1}{2}\binom{j}{j/2} p^{j/2}(1-p)^{j/2} + \sum_{i=\frac{j}{2}+1}^{j} \binom{j}{i} p^i (1-p)^{j-i}, & j \text{ even} \end{cases} \tag{2.8}$$

Eq. 2.8 can be upper-bounded by the Chernoff bound [8]

$$P_j < [2\sqrt{p(1-p)}]^j \tag{2.9}$$

For an AWGN channel, a similar expression can be given as below assuming white Gaussian noise with (two-sided) power spectral density $N_0/2$ and an energy of $\sqrt{E_s}$ per symbol with BPSK modulation [9].

$$P_j = P(B_j/A_0) = \frac{1}{2}\text{erfc}(\sqrt{\frac{jE_s}{N_o}}) \tag{2.10}$$

Eq. 2.10 can be approximated as [6]

$$P_j < e^{-\frac{jE_s}{N_o}} \tag{2.11}$$

Ideas presented in this section can be directly applied to the computation of error probability for binary convolutional codes. There are, however, several possible definitions of error probability, and each must be handled slightly differently [10]. Without loss of generality, assume that the all-zero path is transmitted. This means that the path followed by the encoder is the horizontal path at the top of the trellis diagram (Fig. 2.3). The decoder does not know which path the encoder has followed, but will make a guess based on the received noisy version of the transmitted path. The path actually taken by the encoder is called the *correct path* and the path postulated by the decoder is called the *decoder's path.* The decoder's path can clearly be partitioned into a (possibly empty) set of correct path segments separated by a set of paths which lie entirely below the correct path expect for their end points (Fig. 2.5). These incorrect path segments are called *error events.*

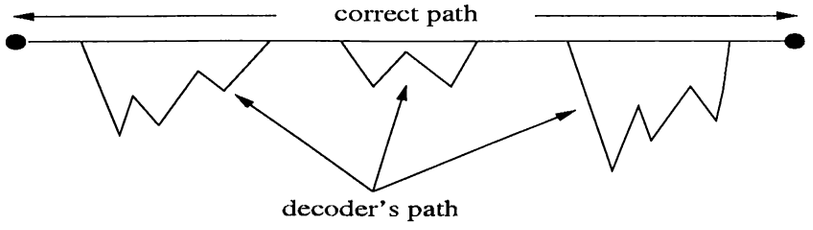

Figure 2.5: Some trellis paths

The *first error event probability* is defined as the probability that the decoder will be off the correct path at the origin. For a maximum-likelihood decoding algorithm, such as Viterbi algorithm, this is possible if a path of weight j, that diverges from the all-zero path at the origin, is closer to the received sequence than the all-zero path. If P_j denotes the probability of this event, then

$$P_j < \gamma^j \tag{2.12}$$

where γ is a channel dependent parameter (see Eq.'s 2.9 and 2.11). Using union bound, probability of first error event is bounded by

$$P_E < \sum_j c_j \gamma^j \tag{2.13}$$

where c_j is the number of paths of weight j that diverge from the all-zero path at the origin. Since c_j are given by the coefficients of the path enumerator $T(D)$ of the convolutional code, Eq. 2.13 can be written by [11]

$$P_E < T(\gamma) \tag{2.14}$$

Another possible case of interest is the *error event probability*, P_e, which is the probability that the decoder is off the correct path at the jth trellis stage. This is just the probability that there is an error event hanging below the correct path at jth stage (Fig. 2.6). By a reasoning similar to the derivation of Eq. 2.14, an upperbound on the

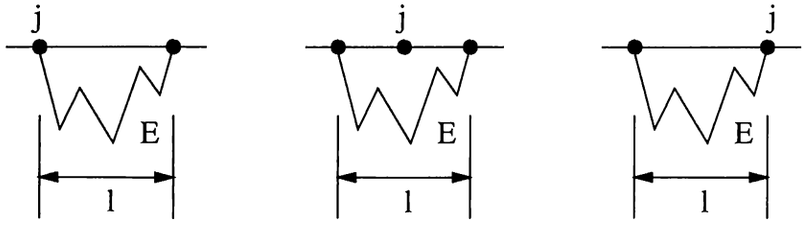

Figure 2.6: Some error events at jth trellis stage

probability of error event is given by [11]

$$P_E < \left.\frac{dT(D,L,I)}{dL}\right|_{D=\gamma,L=1,I=1} \tag{2.15}$$

The decoder outputs information bits corresponding to its postulated path, and that even if it is on the wrong path, some of the individual bits will be "accidentally" right. Thus the *bit error probability*, P_b, will in general be less than the error event probability. Using a reasoning similar to that leading to Eq. 2.7, the following bound on P_b is obtained [11]:

$$P_b < \left.\frac{1}{k}\frac{dT(D,L,I)}{dI}\right|_{D=\gamma,L=1,I=1} \tag{2.16}$$

for a rate-k/n binary convolutional code. In both cases, γ equals either Eq. 2.9 or 2.11 depending on the channel.

2.4 Other Bounds on Probability of Error of Convolutional Codes

The ideas presented in the previous sections have been extended to tighter bounds on the error performance of various coding schemes. As we stated earlier, the performance of a concatenated code can be evaluated by first estimating the symbol error rate of the inner code and then calculating outer code performance. Therefore, the problem of computing

the overall performance of a concatenated system is equivalent to finding good estimates for the inner convolutional code performance. In this respect, this section presents other research on finding improved bounds on error performance of convolutional codes.

In [12], tighter upper bounds on the error-event and bit error probabilities are derived from Viterbi's upper bounds for the case of a BSC with maximum-likelihood decoding. From Eq. 2.8, it is observed that $P_{2n} = P_{2n-1}$, $n = 1, 2, \ldots$. Then,

$$P_{2n} \leq \left[\binom{2n-1}{n} 2^{-2n}\right] (2\sqrt{p})^{2n} \tag{2.17}$$

Since the term in square brackets is monotonically decreasing,

$$\begin{aligned} P_{2n} &\leq \left[\binom{2n_o-1}{n_o} 2^{-2n_o}\right] (2\sqrt{p})^{2n} \\ &\triangleq \Gamma_{n_o} (2\sqrt{p})^{2n} \end{aligned} \tag{2.18}$$

for $n \geq n_o \geq 1$. Finally since $P_{2n} = P_{2n-1}$, $n = 1, 2, \ldots$, Eq.'s 2.14 and 2.16 can be rewritten as,

$$P_E \leq \Gamma_{no}\left(\frac{1}{2}[T(D) + T(-D)] + \frac{1}{2} D[T(D) - T(-D)]\right)_{D=2\sqrt{p}} \tag{2.19}$$

$$P_b \leq \Gamma_{no}\left(\frac{d}{dI}[T(D, I) + T(-D, I)] + \frac{1}{2} D[T(D, N) - T(-D, N)]\right)_{N=1, D=2\sqrt{p}} \tag{2.20}$$

respectively, where $2n_o$ is the index of the first even nonzero coefficient in $T(D)$. The bound is evaluated for rate-$1/2$ code with generators polynomials $1 + D + D^2$ and $1 + D^2$. The comparison shows significant improvement over union bound.

Later in [13], a method for explicit evaluation of Viterbi's union bound for a BSC with maximum-likelihood decoding is proposed. The method uses the observation that $P_{2n} = P_{2n-1}$, $n = 1, 2, \ldots$. Furthermore,

$$P_{2n+1} = P_{2n} - \left(\frac{1}{2} - p\right)\binom{2n}{n} p^n q^n \tag{2.21}$$

for $n = 1, 2, 3, \ldots$. Combining these results, the generating function F for the sequence $P_1, P_2, P_3, \ldots$ can be written as follows

$$F(z) = \sum_{k=1}^{\infty} P_k z^k = \frac{z}{1 - z}\left[\frac{1}{2} - \left(\frac{1}{2} - p\right)(1 - 4pqz^2)^{-1/2}\right] \tag{2.22}$$

Then, for evaluating the error-event probability, $T(D)$ is written as a rational function of the complex variable z which has the Taylor expansion

$$T(z) = \sum_{k=1}^{\infty} t_k z^k \tag{2.23}$$

around $z = 0$. Using partial fractions, $T(z)$ can be decomposed into a polynomial $t(z)$ and a sum of rational terms in z depending on the distinct zeros of T(z) and their multiplicities. It is shown that the latter term can be expressed as a sum of derivatives of F(z). Then the error-event probability is bounded by

$$P_E < t_1 P_1 + t_2 P_2 + \ldots + t_r P_r + terms\ of\ F^i(z) \tag{2.24}$$

where r is the degree of $t(z)$. The same approach is applicable to the explicit evaluation of the bit error probability if $\frac{dT(D,I)}{dI}\Big|_{I=1}$ is expressed as a rational function in z. Comparison with experimental results shows that the method provides an improvement over results of [5] and [12].

In [14], a method based on finite Markov chains is described to calculate the error-event probability of convolutional codes with maximum-likelihood decoding. Results are presented for short constraint length convolutional codes.

In [15], a new upper bound on error-event probability is proposed. Starting from the fact that Viterbi's union bound is quite tight when there are few channel symbols in error (high signal-to-noise ratio) but rather loose when there are many errors, the error-event is separated into two disjoint events corresponding to few (F) and many (M) errors. Then for a BSC with maximum-likelihood decoding, probability of error-event, $P[E]$ is given by,

$$\begin{aligned} P[E] &= P[E/F]P[F] + P[E/M]P[M] \\ &\leq P[E/F]P[F] + P[M] \\ &= P[E,F] + P[M] \end{aligned} \tag{2.25}$$

To obtain an upper bound on $P[M]$ a random-walk argument is used. If a metric s_e is used when a channel error occurs and a metric s_c when the channel symbol is correctly received, then the cumulative metric along the correct path is a random walk with $P[z_i = s_e] = p$ and $P[z_i = s_c] = (1-p)$, where z_i form a sequence of statistically independent, identically distributed random variables. If S_k denote the cumulative metric for the first k channel symbols which have j_k errors among them, then

$$S_k = j_k s_e + (k - j_k) s_c \tag{2.26}$$

Those error patterns for which $S_k \leq -f$ are said to have many errors. Based on a lemma given by Gallager [8], an upper bound on $P[M]$ is given as

$$P[M] = P\left[\min_k S_k \leq -f\right] \leq 2^{-f} \tag{2.27}$$

where f is a parameter to be chosen later. Similarly, those error patterns for which $S_k > -f$ said to have few errors. Equivalently,

$$\begin{aligned} &j_k s_e + (k - j_k)s_c > -f, \quad \text{for all } k \\ &j_k < \tfrac{f}{s_c - s_e} + k\tfrac{s_c}{s_c - s_e} \triangleq r_k, \quad \text{for all } k \end{aligned} \tag{2.28}$$

To upper bound $P[E, F]$, first union bound is used to obtain,

$$P[E, F] \leq \sum_k \sum_i P[E_{ki}, F], \tag{2.29}$$

where E_{ki} is the event that a path of length k and weight i is causing an error. A path has few channel errors only if it stays below the barrier in Fig. 2.7 for all k. If all paths

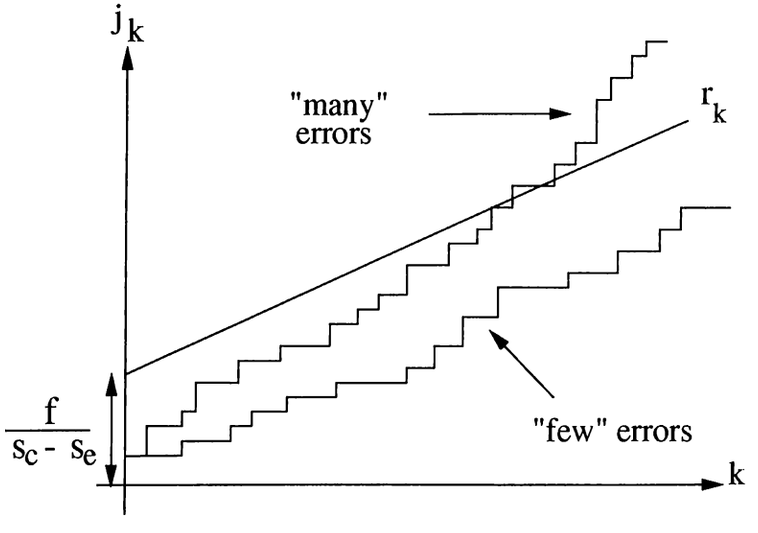

Figure 2.7: Barrier at r_k which separates paths with many and few errors

of length k with $j_k < r_k$ channel errors are counted, all paths with few channel errors together with the paths with many channel errors that take one or more detours above the barrier will be taken. Hence,

$$P[E_{ki}, F] \leq P[E_{ki}, j_k < r_k], \tag{2.30}$$

where

$$P[E_{ki}, j_k < r_k] = \sum_{j_k < r_k} \binom{k}{j_k} p^{j_k} q^{k-j_k} P[E_{ki}/j_k]. \tag{2.31}$$

In [15], it is shown that starting from Eq. 2.31, $P[E, F]$ is bounded by,

$$P[E, F] \leq \left(\frac{pq_0}{p_0 q}\right)^{f/(s_c - s_e)} T(D, I) \tag{2.32}$$

where,

$$p_0 = \frac{\eta p}{\eta p + q} \leq p \tag{2.33}$$

$$q_0 = 1 - p_0 \geq q \tag{2.34}$$

for $0 < \eta \leq 1$. The new upper bound on error-event probability is obtained by combining Eq. 2.27 and Eq. 2.32,

$$P[E] \leq \left(\frac{pq_0}{p_0 q}\right)^{f/(s_c - s_e)} T(D, I) + 2^{-f} \tag{2.35}$$

The above bound is valid for all f. The parameter f is chosen so as to minimize the right handside of Eq. 2.35. Comparisons show that this bound is better than Viterbi's bounds and bounds proposed in [12].

In [16], new upper and lower bounds and approximations on the sequence, event, first event, and bit-error probabilities of convolutional codes are presented. Each of these probabilities are precisely defined and the relationship between them described. Some of the new bounds are found to be very close to computer simulations at low signal-to-noise ratios. Also simple modifications to the traditional union upper bound are described for both hard- and soft-decision channels that allow better performance estimates to be made.

In [17], a Markovian technique is described to calculate the exact performance of a convolutional code with Viterbi decoding over binary symmetric channels. The Hamming distance d_H between two sequences, defined as the number of bits in which they differ, is used as the decoding metric. R^N denotes a received sequence of length N time units. Viterbi algorithm at each node γ of the trellis diagram chooses a maximum-likelihood path among the paths leading to that node and a metric D_γ^N is computed from the metrics at time unit $N-1$. For a maximum-likelihood path A_γ^N,

$$D_\gamma^N = d_H(A_\gamma^N, R^N). \tag{2.36}$$

The *relative metric*, $\overline{D}_\gamma^N$, at node γ is obtained by subtracting the minimum metric among all nodes from D_γ^N.

$$\overline{D}_\gamma^N = D_\gamma^N - \min_\delta(D_\delta^N) \tag{2.37}$$

The *metric vector* at time unit N is

$$\overline{D}^N = (\overline{D}_0^N, \overline{D}_1^N, \ldots, \overline{D}_{2^v-1}^N) \tag{2.38}$$

where 2^v is the number of states of the convolutional encoder. These internal states of the Viterbi decoder form the Markov chain, with the received symbol r in the sequence R^N at time N determining the transitions from one state to another. For example, the Hamming distance between a branch label and a channel output is at most two for a rate-$1/2$, 2-state convolutional code with generator matrix $[1, 1+D]$. The possible metric vectors or states of the Markov chain are therefore $(2,0), (1,0), (0,0), (0,1), (0,2)$. The transition probability matrix T for the resulting Markov chain can be easily formed by checking which received signals determine a transition from one metric vector to next. The conditional probability of this received signal defines the transition probability. The probability of bit error is computed from the steady-state behavior of the Markov chain.

At a given state of the Markov chain, the exact bit error probability for the current k information bits is computed by considering all future received sequences that stem from a particular decoded branch. Given metric state $\overline{D}$ and a received sequence r^l of l branches, $P(r^l, \overline{D}) = i/k$ if i information bits are decoded incorrectly. Averaging over all metric states and received sequences, the probability of error can be expressed as

$$P_b = \sum_{\overline{D}, r^l} \pi_{\overline{D}} q_{r^l} P(r^l, \overline{D}) \tag{2.39}$$

where q_{r^l} is the probability of receiving sequence r^l and $\pi_{\overline{D}}$ is the steady-state distribution of the metric state $\overline{D}$. It is understood that the received sequences in Eq. 2.39 are mutually disjoint and exhaust all possibilities that cause errors in the current time limit. The main problem encountered in calculating Eq. 2.39 is cataloging all possible future received sequences that cause information bits in a given time unit to be decoded incorrectly. In [17], three different approaches are introduced to overcome this difficulty. This method has limitations due to the rapid increase in the number of Markov states as the constraint length increases. These problems are addressed in [17]. For short constraint length codes, calculations are compared to simulations which show acceptable accuracy.

Chapter 3

Improved Bounds on Convolutional Code Performance

Estimating the error performance of a concatenated coding scheme requires the evaluation of the error performance of the inner code. Union bound for convolutional codes, in its classical form given by Viterbi [11], is a well known approach to estimate the error performance, provided that the bit error rate (BER) at the ML decoder output is very low, for example, around 10^{-3} (for constraint lengths up to seven) or 10^{-6} (for constraint lengths greater than seven). When convolutional codes are used as inner codes in concatenated systems, however, the codes should be optimized for a BER of around 10^{-2} to 10^{-3} (the outer code takes this error rate and produces an output error rate of 10^{-5} to 10^{-10}) [16]. Unfortunately, the accuracy of the union bound is limited in this latter range of operation (low SNR).

In this chapter, we propose two different methods for improving the union bound for binary convolutional codes. The first method is based on the ideas presented in [3] for linear block codes over a binary input AWGN channel. The results of [3] are extended to the convolutional codes over a BSC with hard decision decoding and an AWGN channel with soft decision decoding. In Section 3.1, this approach will be presented using a rate-1/2, constraint-length-3 convolutional code with the generator matrix $g(D) = [1+D+D^2, 1+D^2]$. Results will be compared to classical union bound and to simulation data.

The second method aims to improve the union bound in the low SNR region. In Section 3.3, a detailed analysis of this approach will be presented using the same example code. Results will be presented and be compared to classical union bound and to simulation data.

3.1 Improving Union Bound using Weight Distributions

We will first consider a linear block code of length n symbols over a binary input symmetric channel with BPSK modulation and following properties :

1. A binary input alphabet $X = \{0, 1\}$.
2. An output alphabet Y taking either continuous values with conditional probability density function $p_x(y)$ or discrete values with conditional probability $P_x(y)$, depending on the channel.
3. The symmetry condition $P_0(y) = P_1(-y)$ or $p_0(y) = p_1(-y)$ holds for all outputs y.

For such a code, all non-zero codewords can be grouped according to their weights d (or distance from the all-zero codeword) into subsets χ_d for $d_{min} \leq d \leq d_{max}$.

Then for a decoder which determines the most likely codeword according to some (optimum) metric on the received vector $\mathbf{y}$, the word error probability given all-zero codeword $\mathbf{x_0}$ was sent is equal to

$$\begin{aligned} P_{E0} &= P[\bigcup_{d=d_{min}}^{d=d_{max}} (\text{ any codeword in } \chi_d \text{ has metric} > \text{that of } \underline{x_0}\ )] \\ &< \sum_{d=d_{min}}^{d=d_{max}} P(\text{ any codeword in } \chi_d \text{ has metric} > \text{that of } \underline{x_0}\ ) \\ &\triangleq \sum_{d=d_{min}}^{d=d_{max}} P_{E0}(d) \end{aligned} \tag{3.1}$$

In [3], based on a result of [18], it is shown that

$$P_{E0}(d) < exp[-nE(\delta)] \tag{3.2}$$

where

$$\begin{aligned} E(\delta) &= \max_{0<\rho\leq 1}\{-\rho[r(\delta)+\delta\ln h(\rho)+(1-\delta)\ln g(\rho)] \\ &\quad -(1-\rho)\ln[h(\rho)+g(\rho)]\} \end{aligned} \tag{3.3}$$

Here $\delta \triangleq d/n$, $r(\delta) \triangleq (\ln K_d/n)$, and $K_d \triangleq$ Size of Subset χ_d, and

$$h(\rho) = \int_{\infty}^{\infty}[p_0(y)^{1/(1+\rho)}+p_0(-y)^{1/(1+\rho)}]^{-(1-\rho)}p_0(y)^{1/(1+\rho)}p_0(-y)^{1/(1+\rho)}\mathrm{d}y \tag{3.4}$$

$$g(\rho) = \int_{\infty}^{\infty}[p_0(y)^{1/(1+\rho)}+p_0(-y)^{1/(1+\rho)}]^{-(1-\rho)}p_0(y)^{2/(1+\rho)}\mathrm{d}y \tag{3.5}$$

for the continuous output case, and

$$h(\rho) = \sum_{y}[P_0(y)^{1/(1+\rho)}+P_0(-y)^{1/(1+\rho)}]^{-(1-\rho)}P_0(y)^{1/(1+\rho)}P_0(-y)^{1/(1+\rho)} \tag{3.6}$$

$$g(\rho) = \sum_{y}[P_0(y)^{1/(1+\rho)}+P_0(-y)^{1/(1+\rho)}]^{-(1-\rho)}P_0(y)^{2/(1+\rho)} \tag{3.7}$$

for the discrete output case. Eq. 3.1 results in a tighter union bound for linear block codes in terms of the weights of the codewords. Letting $\rho = 1$ in Eq. 3.3 results in the traditional union bound. Usually, the bit error probability is a more interesting measure on the performance. An upper bound on the bit error probability can be obtained, following the results of Section 2.3 and using Eq. 3.1, as

$$P_b < \sum_{d=d_{min}}^{d=d_{max}} \frac{d}{n} P_{E0}(d) \tag{3.8}$$

This method can be modified for use with convolutional codes. An apparent problem is that convolutional codes have infinite length sequences or paths as codewords. To overcome this difficulty, first consider the path r_{ji}, which is the ith path that diverges from the all-zero path at the origin and merges back at the jth trellis stage. Such a path has length $n_j = 2j$ bits for a rate-1/2 binary convolutional code. Let the distance of this path from the all-zero path be d_{ji}. We form a set S_j defined as

$$\begin{aligned} S_j &= \{\text{paths } r_{ji} \text{ of length } n_j \text{ for } i = 1, 2, \ldots, M-1 \\ &\qquad \text{and the all-zero path, } r_{j0}, \text{ of length } n_j\} \end{aligned} \tag{3.9}$$

where M is the total number of such paths. An example of such paths is shown in Fig. 3.1 for the case of rate-1/2, constraint-length-3 code with generator matrix $[1 + D + D^2, 1 + D^2]$. The set S_j can now be considered as a block code of block length

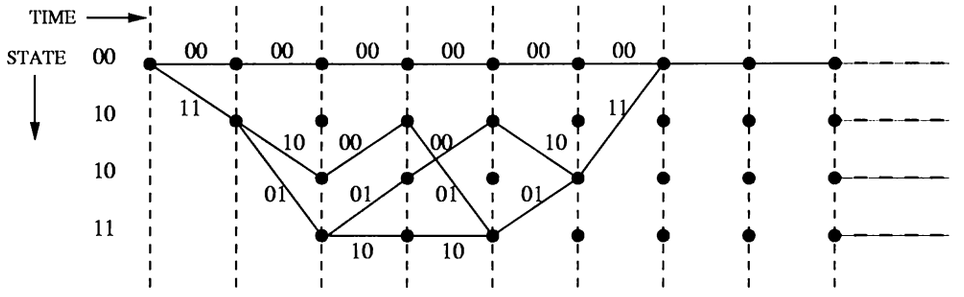

Figure 3.1: Paths forming the set S_6, for $j = 6$

n_j with M codewords. The elements of the set S_6 of Fig. 3.1 and their distances from the all-zero codeword are listed in Table 3.1. Note that the set contains two codewords of distance 7 and one codeword of distance 8. Then, the codewords in a set S_j can be further grouped into subsets χ_{jd} for $d_{j,min} < d < d_{j,max}$. With this grouping, the previous steps discussed for a linear block code can be directly applied to the set S_j. So, following

S_6	codeword	distance, d_{6i}
r_{60}	000000000000	–
r_{61}	111000010111	7
r_{62}	110101001011	7
r_{63}	110110100111	8

Table 3.1: Codewords and their distances from the all-zero codeword

Eq. 3.1, one can obtain a probability of error figure, $P_{j,E0}$ for the set S_j. The minimum value of j or the length of shortest path is a characteristic of the convolutional code and can be determined from its path enumerator polynomial $T(D)$. On the other hand the path lengths, j, can be increased indefinitely. As j increases, however, number of available paths increases very rapidly. Therefore, both the enumeration of the paths and the application of the algorithm become cumbersome. Since the probability of having a path that stays diverged for very large values of j decreases rapidly, it should be possible to truncate the maximum path length at some value L_{max} without affecting the accuracy of the bound significantly, provided that L_{max} is large enough. Then, the overall figure for the probability of sequence error is obtained as

$$P_{E0} < \sum_{j=L_{min}}^{L_{max}} \sum_{d=d_{j,min}}^{d=d_{j,max}} P_{j,E0}(d) \tag{3.10}$$

where $P_{j,E0}(d)$ is calculated using Eq.'s 3.3-3.7. Similarly, for the bit error probability we obtain

$$P_b < \sum_{j=L_{min}}^{L_{max}} \sum_{d=d_{j,min}}^{d=d_{j,max}} \frac{d}{n_j} P_{j,E0}(d) \tag{3.11}$$

Consider the path enumerator $T(D, L)$ for the example code:

$$\begin{aligned} T(D,L) &= \frac{D^5L^3}{1-DL(1+L)} \\ &= D^5L^3 + D^6L^4(1+L) + D^7L^5(1+L)^2 \\ &\quad + \ldots + D^{5+k}L^{3+k}(1+L)^k + \ldots \end{aligned} \tag{3.12}$$

where powers of D keep track of the weights of the paths and powers of L denote the path lengths. We note that the shortest path for this code has a length of 3 branches, hence $L_{min} = 3$. Before choosing a value for L_{max}, we consider a method that enumerates the number of paths that diverge from the all-zero path at the origin and merge at the jth trellis stage for $j \geq L_{min}$ and their weights. Note that all the required information can be read from the path enumerator. However, it is difficult to calculate the path enumerator and obtain its series expansion by the methods of Section 2.2 for convolutional codes with large constraint lengths. It is easier to obtain a state transition matrix by examining the state diagram of the code, possibly by a computer program.

To demonstrate the basic idea, we will once again make use of our running example. However, the method is valid for all binary convolutional codes with non-recursive (no feedback from output) encoder structures. Following the labeling of Section 2.2, the state diagram for the example code is redrawn in Fig. 3.2. The corresponding state transition

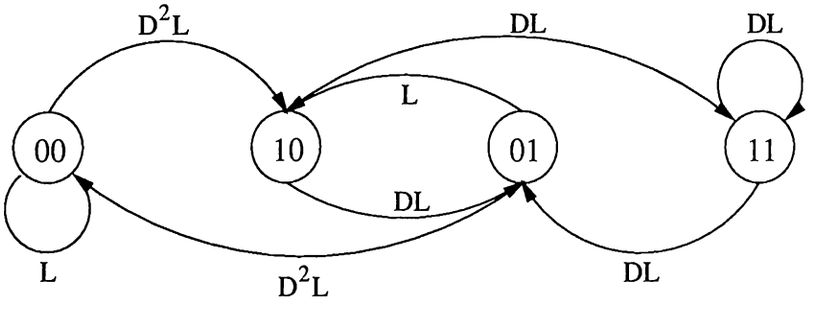

Figure 3.2: State diagram for rate-1/2 constraint-length-3 code

matrix is

$$T = \begin{pmatrix} L & 0 & D^2L & 0 \\ D^2L & 0 & L & 0 \\ 0 & DL & 0 & DL \\ 0 & DL & 0 & DL \end{pmatrix} \tag{3.13}$$

where the (i,j)th entry denotes the transition from state i to state j. From Fig. 3.1 and Fig. 3.2, we observe that once a path out of the 00 state is taken, then whatever the intermediate path is, there exists a single state from which it is possible to return to 00 state, namely the 01 state with weight D^2L. This observation is true for all rate-1/2 codes having a non-recursive encoder structure. The path definition we impose, requires that a path diverges from the all-zero path at the origin and then merges back at the jth trellis stage. Therefore, starting from the 00 state, the transitions dictated by any path in S_j should reach the 01 state after $j-1$ stages and should never visit the 00 state at any intermediate time. With these requirements, all transitions *to* the 00 are removed from the state transition matrix and then its $(j-1)$th power is calculated.

$$T_j = \begin{pmatrix} 0 & 0 & D^2L & 0 \\ 0 & 0 & L & 0 \\ 0 & DL & 0 & DL \\ 0 & DL & 0 & DL \end{pmatrix}^{j-1} \tag{3.14}$$

The element of T_j denoted by $T_j(00,01)$ has the complete list of the paths of our concern, only the final transition from state 01 to 00 is missing. This transition is common to all and its weight is known to be D^2L. Therefore, the paths forming the set S_j are given by the polynomial

$$S_j(D,L) = T_j(00,01)D^2L \tag{3.15}$$

For example, to list the paths of S_6, T_6 is calculated. $T_6(00,01)$ in that case turns out to be

$$T_6(00,01) = 2D^5L^5 + D^6L^5 \tag{3.16}$$

Hence,

$$S_6(D,L) = 2D^7L^6 + D^8L^6 \tag{3.17}$$

Comparison with Table 3.1, shows that information on all paths has been obtained correctly. Using this methodology, it is possible to enumerate all paths of the sets S_j for $j \geq L_{min}$.

We are now left with the decision on how to choose L_{max}. At high SNR, the probability of having a path that stays diverged from the all-zero path is very low and therefore, choosing a small value for L_{max} will provide sufficient accuracy. For low SNR region, on the other hand, a much larger value should be used. In the application of the method, L_{max} is increased until further contributions do not increase the accuracy over the SNR range of interest. As a criterion, the probability values can be calculated at L and $L+\Delta L$. If the absolute difference between the probability values are below some desired precision, then L_{max} is fixed at L.

3.2 Results and Discussions

In this section, results for the rate-1/2, constraint-length-3 code will be presented. Improved bound will be compared to simulations and to the classical union bound as given by Viterbi.

Simulations are performed using a Viterbi software decoder written in C language. Though the final program is extensively modified for use with different channel characteristics and with codes of different constraint length, the basic modules are based on Robert Morelos Zaragoza's Viterbi decoder program [19].

It is usually difficult to get a good statistical sample especially for very low error rates. The experiments are performed over $10^6 - 10^9$ bits blocks and are repeated several times to ensure that the mean value lies in an acceptable confidence interval. For experiments of this section all data points, x_i lie in an interval of $\Delta x_i = 0.01 x_i$ with 95% probability.

First, a BSC with BPSK modulation will be considered. The crossover probability in this case is given by

$$p = \frac{1}{2}\mathrm{erfc}(\sqrt{\frac{RE_b}{N_0}}) \tag{3.18}$$

where R denotes the rate of the code and E_b/N_0 is the SNR at the system input. Therefore, in Eq. 3.6 and Eq. 3.7, $P_0(1) = p$ and $P_0(0) = q = 1 - p$.

For this case, experiments show that truncating the length of possible paths at $L_{max} = 30$ which is 10-times the constraint length provide sufficient accuracy. The maximization in Eq. 3.3 is performed numerically. In Fig. 3.3, the results are plotted for a range of SNR values and compared to classical union bound and to simulation data. Simulation data

are obtained using Viterbi decoding with 1-bit quantization and hard decision decoding.

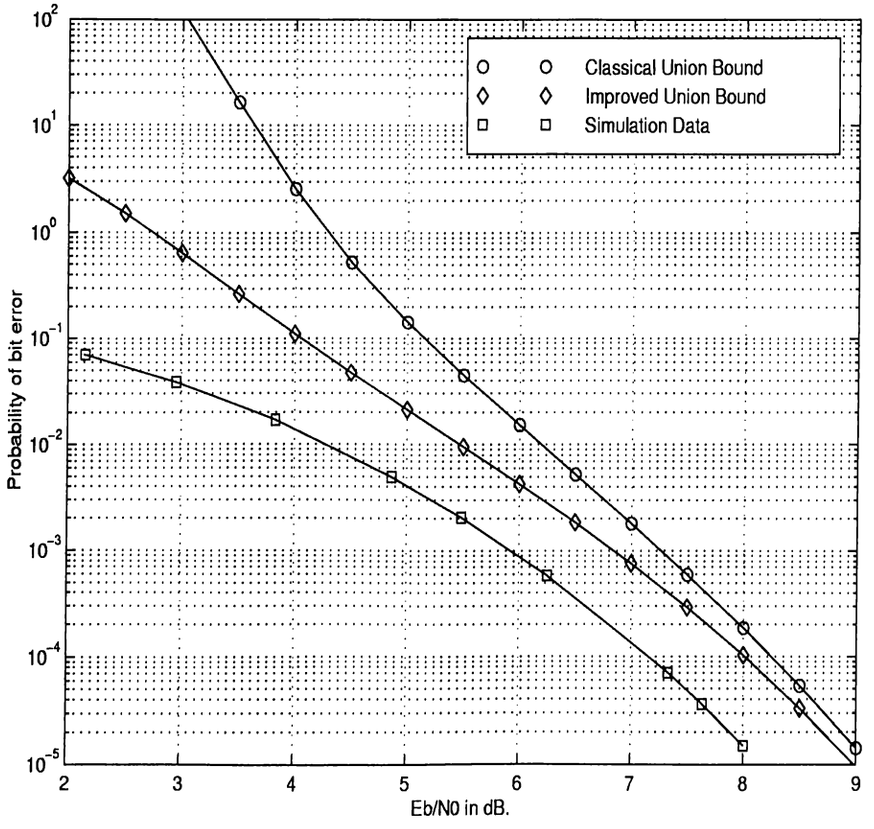

Figure 3.3: Probability of bit error for $R = 1/2$, $K = 3$ code over BSC with BPSK modulation

In Fig. 3.4, the relative change that would be obtained for $L_{max} = 35$ is plotted over a range of SNR values. It is seen that the change in the 2-6.5dB. region is very small, and the results are exactly the same up to machine precision for SNR values larger than 6.5dB. Therefore, truncation causes no significant error in estimations.

Next, an AWGN channel will be considered. For this case, $p_0(y)$ in Eq. 3.4 and Eq. 3.5 is given by

$$p_0(y) = \frac{1}{\sqrt{2\pi}} \exp[-\frac{(y - \sqrt{2RE_b/N_0})^2}{2}] \tag{3.19}$$

In Fig. 3.5, results of this case are plotted and compared to classical union bound and to

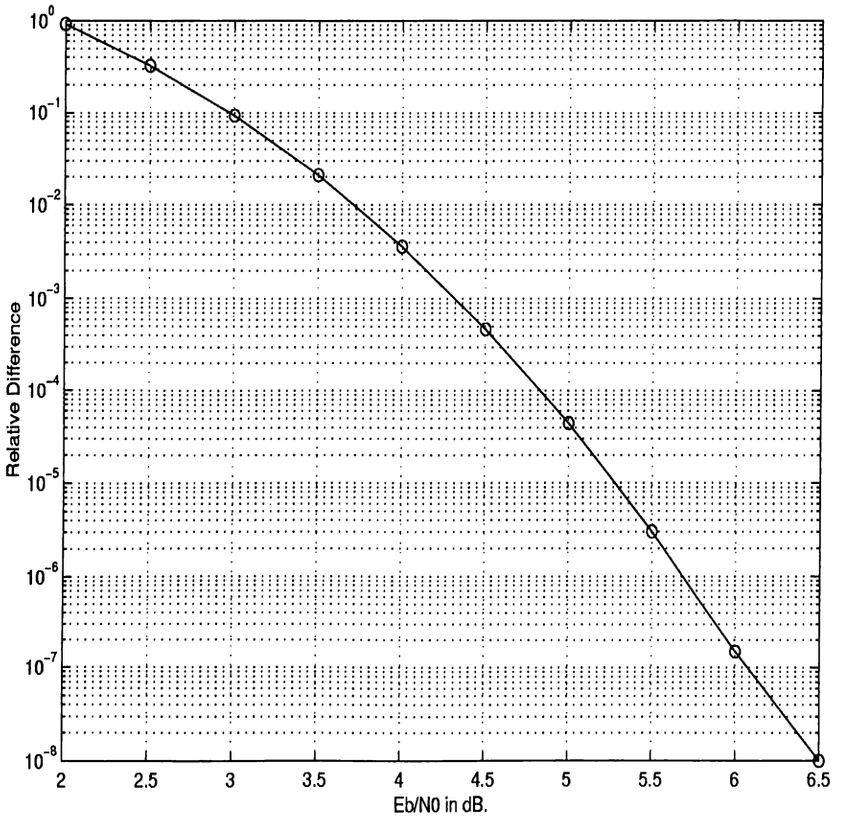

Figure 3.4: Difference that would result in calculations for $L_{max} = 35$

simulation data. Simulation results are obtained by performing Viterbi decoding, with 8-bit quantization and soft decision.

In calculations $L_{max} = 30$ was chosen. In Fig. 3.6 relative difference obtained for $L_{max} = 35$ is plotted to show that sufficient accuracy is reached. The results are exactly the same up to machine precision for SNR values larger than 4.0dB.

From the plots it is observed that, the method converges to the classical union bound for high SNR. This is reasonable since it is known that classical union bound provides a tight estimate in that region. For the low SNR region on the other hand, the method provides tighter estimates while the classical union bounds increases abruptly to values much larger than one.

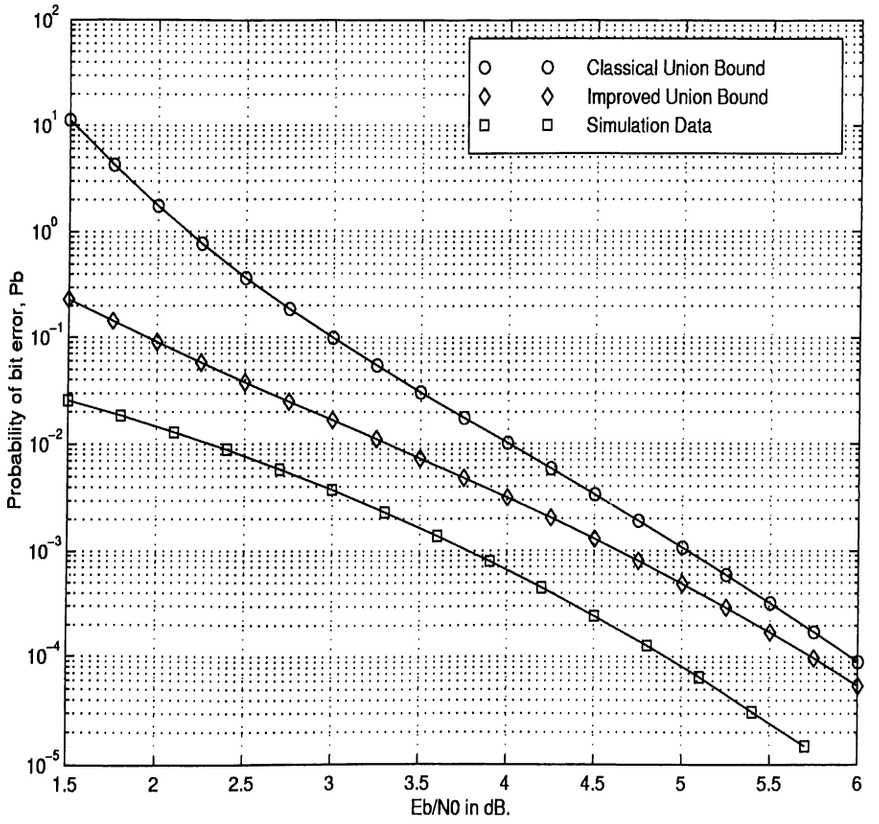

Figure 3.5: Probability of bit error for $R = 1/2$, $K = 3$ code over AWGN channel with BPSK modulation

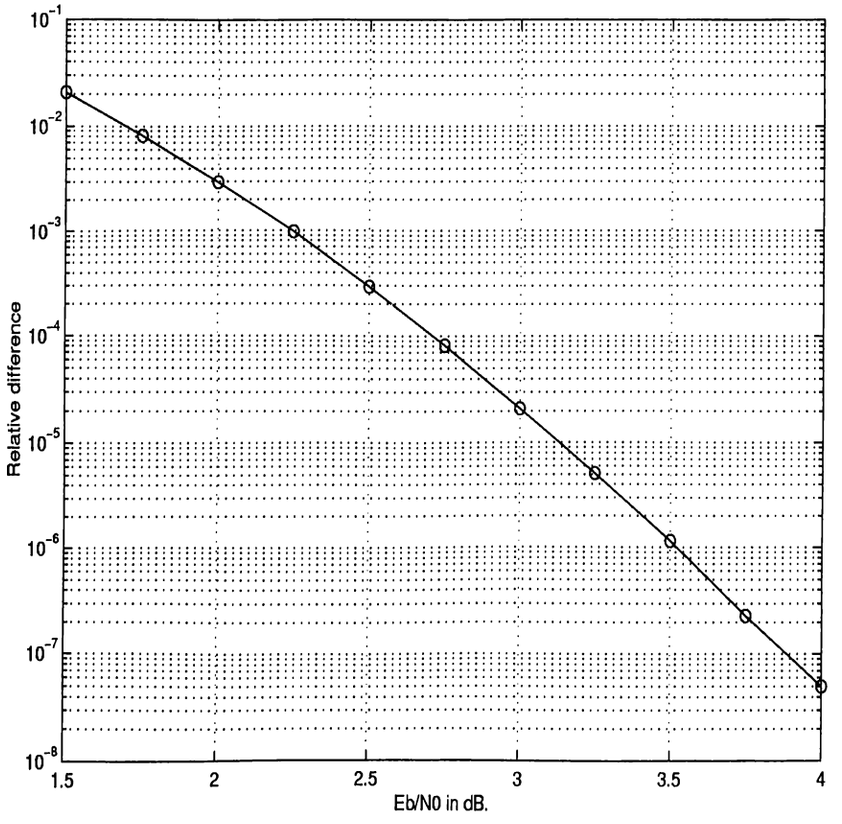

Figure 3.6: Difference that would result in calculations for $L_{max} = 35$

3.3 Improving Union Bound by Partitioning

It is known that the classical union bound as given by Viterbi [11] becomes loose at low SNR values. In this section, a method for improving the union bound over the low SNR region will be presented.

We will first consider the maximum-likelihood decoding of a received sequence under the assumption that the all-zero path was transmitted. The received path will be decoded into a path of weight d if and only if it is closer to that path than to all-zero path. Therefore, the probability that the received path will be decoded into a path of weight d decreases for increasing d.

In Section 2.3, it was shown that the first error event probability is bounded by $T(D)$ evaluated at $D = \gamma$ where $\gamma \triangleq \sum_y \sqrt{P_0(y)P_1(y)}$. In general,

$$T(D) = c_{d_f} D^{d_f} + c_{d_f+1} D^{d_f+1} + c_{d_f+2} D^{d_f+2} + \dots \tag{3.20}$$

where d_f is the free distance of the code. When evaluated at γ, the value of D^i which represents the probability that the received path will be decoded into a path of weight i decreases with increasing i. Typically, only the terms up to a certain power of D will dominate the sum $T(\gamma)$ depending on the SNR. However, the number of paths of weight i given by the coefficient of D^i increases at the same time. Since the union bound calculates the error probability by summing over all paths, the contribution from terms with high powers of D may become significant due to the multiplicity even though the individual probabilities are small.

We propose the following method for upper-bounding the probability of bit error. First of all, for all practical cases, it is difficult to obtain a closed form expression for $T(D)$. Therefore, $T(D)$ is represented by its polynomial expansion. Obviously, it is not possible to sum up infinitely many terms. Instead, $T(D)$ is truncated at some sufficiently large degree d_{max} (usually several times as the constraint length). As a result, union bound on first error event probability is given by

$$P_E < c_{d_f} \gamma^{d_f} + c_{d_f+1} \gamma^{d_f+1} + \dots + c_{d_{max}} \gamma^{d_{max}} \tag{3.21}$$

Instead of calculating the error bound in this way, we partition it into two parts. If a sequence of weight less than i is received, then we use the path enumerator polynomial to calculate the probability of error. A sequence of weight less than i can be decoded into any path of weight $d_f \leq d \leq 2(i-1)$ by a ML decoder. Therefore, only these terms

of the path enumerator polynomial is used in the summation. If a sequence of weight larger than i is received, we assume that the decoder always makes an error and hence in this case the probability of error is equated to the probability of receiving such a path. The overall probability of error is given by the sum of the two cases. The value of i changes the contribution of the two cases. Therefore, we repeat the calculations for i running from $\lceil d_f/2 \rceil$, which is the minimum number of channel errors that may cause the received sequence to be decoded into a wrong path, to $\lceil d_{max}/2 \rceil$ which is the number of channel errors that may cause the received sequence to be decoded into a path that is most far away from the correct path assuming that we use a truncated code. Then, the minimum is chosen as the error probability.

Consider a BSC with a crossover probability of p. In this case the probability of receiving sequences of weight $d \geq i$ is given by

$$P(d \geq i) = \sum_{d=i}^{d_{max}} \binom{d_{max}}{d} p^d (1-p)^{d_{max}-d} \tag{3.22}$$

Hence the following minimization over i is performed:

$$P_E \leq \min_{\lceil \frac{d_f}{2} \rceil \leq i \leq \lceil \frac{d_{max}}{2} \rceil} \{ \sum_{d=i}^{d_{max}} \binom{d_{max}}{d} p^d (1-p)^{d_{max}-d} + \sum_{w=d_f}^{2(i-1)} c_w \gamma^w \} \tag{3.23}$$

Similarly, the upper bound on bit error probability given by

$$P_b < \frac{dT(D,L,I)}{dI}\Big|_{D=\gamma, L=1, I=1} = T'(\gamma) \tag{3.24}$$

can be modified using the same algorithm. Let the truncated $T'(D)$ be given by,

$$T'(D) = c'_{d_f} D^{d_f} + c'_{d_f+1} D^{d_f+1} + \ldots + c'_{d_{max}} D^{d_{max}} \tag{3.25}$$

Then the following minimization over i is performed:

$$P_b \leq \min_{\lceil \frac{d_f}{2} \rceil \leq i \leq \lceil \frac{d_{max}}{2} \rceil} \{ \sum_{d=i}^{d_{max}} \binom{d_{max}}{d} p^d (1-p)^{d_{max}-d} + \sum_{w=d_f}^{2(i-1)} c'_w \gamma^w \} \tag{3.26}$$

For AWGN channel, an equivalent expression for Eq. 3.22 can be derived assuming white Gaussian noise with (two-sided) power spectral density $N_0/2$ and an energy of $\sqrt{E_s}$ per bit. The condition of receiving a path of weight $d \geq i$ is given by the probability

$$P\{\sqrt{N_1^2 + N_2^2 + \ldots + N_{d_{max}}^2} \geq \sqrt{4iE_s}\} \tag{3.27}$$

assuming BPSK with signal levels of $\{\sqrt{E_s}, -\sqrt{E_s}\}$. N_i are Gaussian with density $N(0, N_0/2)$. Normalizing to obtain unity variance, probability of decoding error is given by

$$P\{\sqrt{z_1^2 + z_2^2 + \ldots + z_{d_{max}}^2} \geq \sqrt{\frac{8iE_s}{N_0}}\} \tag{3.28}$$

From probability theory, $y = \sqrt{z_1^2 + z_2^2 + \ldots + z_{d_{max}}^2}$ has the following distribution

$$f_y(y) = \frac{ye^{-\frac{y^2}{2}}(\frac{y^2}{2})^{(\frac{d_{max}}{2})-1}}{(\frac{d_{max}}{2} - 1)!} \quad , y > 0 \tag{3.29}$$

Hence, the equivalent relation for Eq. 3.22 is given by the integral

$$P(d \geq i) = \int_{\sqrt{\frac{8iE_s}{N_0}}}^{\sqrt{\frac{8d_{max}E_s}{N_0}}} f_y(y)\mathrm{d}y \tag{3.30}$$

Therefore, the method is directly applicable to AWGN case if Eq. 3.22 is replaced by Eq. 3.30 and $\gamma = e^{-\frac{E_s}{N_0}}$ is used.

3.4 Results and Discussions

In this section, results for the rate-1/2, constraint-length-3 code will be presented. Improved bound will be compared to simulations and to the classical union bound as given by Viterbi. Simulation data are obtained under the same conditions of the previous cases.

Application of the algorithm requires the truncation of polynomial representing the series expansion of the path enumerator. In the example cases, polynomial of $T(D)$ is truncated at a degree $d_{max} = 30$. First a BSC with a crossover probability of p is considered. The probability of bit error is obtained following the discussions of the previous section and is plotted for a range of SNR values. In Fig. 3.7, results are compared to simulation data and classical union bound.

Next, an AWGN channel will be considered. For this case, the integral relation in Eq. 3.30 is evaluated. In Fig. 3.8, results are plotted over a range of SNR values.

Results show improvement in the low SNR region, where the union bound in its classical form becomes useless. One important consequence is that the bound in this form is always upper-bounded by one.

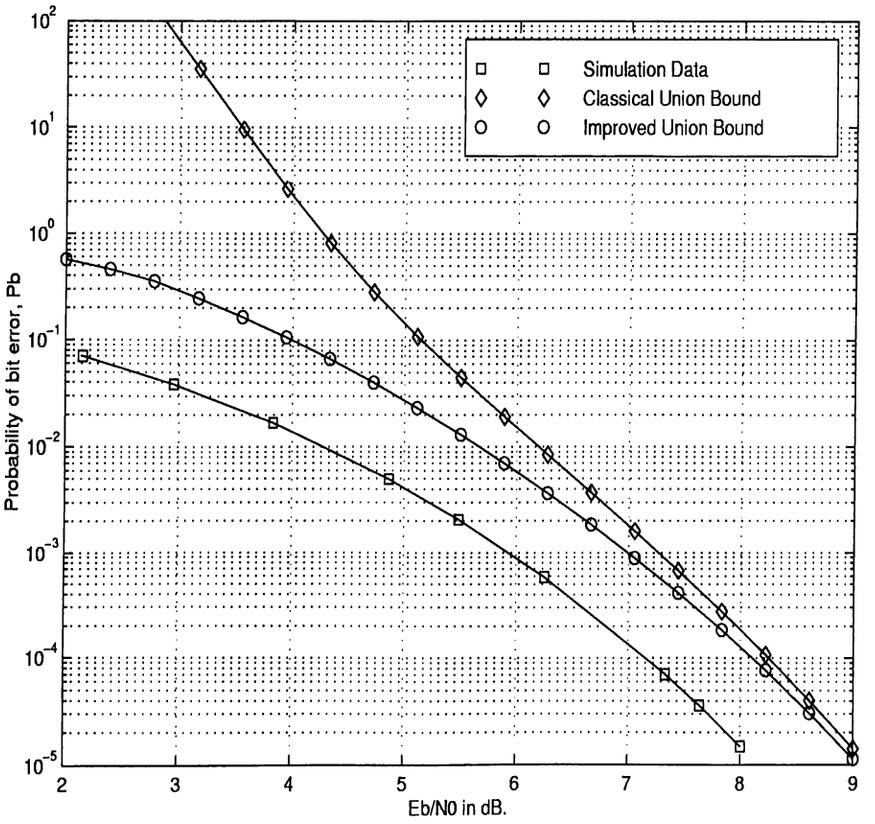

Figure 3.7: Probability of bit error for $R = 1/2$, $K = 3$ code over BSC with BPSK modulation

The methods of this chapter provide improvements over the classical union bound. It is important to get a tighter upper bound on the low SNR region since this is the range of interest in concatenated coding. In the next chapter, the results of this chapter will be used to evaluate the performance of an ideally interleaved concatenated scheme.

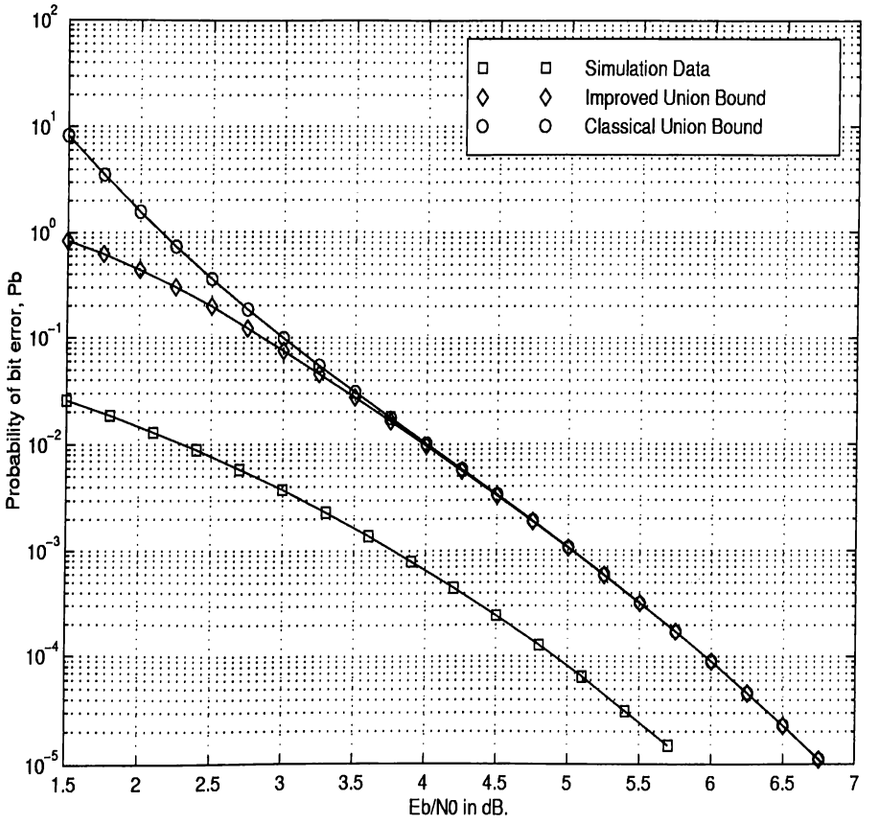

Figure 3.8: Probability of bit error for $R = 1/2$, $K = 3$ code over an AWGN channel

Chapter 4

Ideally Interleaved System Performance

In Chapter 3, methods were proposed to obtain a better estimate on the error performance of convolutional inner codes. This was critical because the estimated error rate at the convolutional decoder output was to be used as input to the RS decoder error rate calculation. First section of this chapter presents closed form expressions for the word, symbol and bit error rates of RS codes for known input symbol error rate. In the next section, performance of an ideally interleaved system with rate-1/2, constraint-length-7 convolutional inner code and (255, 223) RS outer code is evaluated.

4.1 Overview of Reed-Solomon Codes

Reed-Solomon codes are an important and popular subset of a more general class of block codes known as Bose-Chaudhuri-Hocquenghem (BCH) codes [20], [21]. Unlike the convolutional codes, structure of BCH codes has strong algebraic properties. For an $\alpha \in GF(q^m)$ and for any specified m_o and d_o, the code generated by the generator polynomial $g(x)$ is a BCH code if and only if g(x) is the polynomial of lowest degree over $GF(q)$ for which $\alpha^{m_o}, \alpha^{m_o+1}, \ldots, \alpha^{m_o+d_o-2}$ are roots. The blocklength of the code

is given by the order of α. It can be shown that the minimum distance of the code in this case turns out to be at least d_o, the *designed distance.* The generator polynomial of the code is given by

$$g(x) = LCM[f_{m_o}(x), f_{m_o+1}(x), \ldots, f_{m_o+d_o-2}(x)], \tag{4.1}$$

where $f_j(x)$ is the minimal polynomial of α^j, and $\alpha \in GF(q^m)$. Usually one desires a large block length and thus α is chosen as an element with largest order, that is, as a primitive element. In this case, the blocklength is given by $n = q^m - 1$. Furthermore, if m_o is chosen to be equal to one, then $d_o = 2t_o + 1$ holds, where t_o is the number of correctible errors.

Reed-Solomon codes are obtained by setting $m = m_o = 1$. Then for a primitive element α

$$n = q^m - 1 = q - 1 \tag{4.2}$$

The minimum polynomial of α^j is simply $(x - \alpha^j)$, and thus

$$g(x) = (x - \alpha)(x - \alpha^2)\ldots(x - \alpha^{d_o-1}) \tag{4.3}$$

This is always a polynomial of degree of $d_o - 1 = 2t_o$. The resulting code with blocklength of n symbols has $d_o - 1$ parity symbols and has minimum distance d_o. Hence

$$n - k = d_o - 1 = 2t_o \tag{4.4}$$

is satisfied, where k is the number of information symbols.

The Reed-Solomon codes are optimum in the sense that they satisfy the Singleton bound with equality. If $d_o = 2t_o + 1$ is the designed distance of the code, then the minimum distance d^* satisfies

$$d^* \geq d_o = 2t_o + 1 = n - k + 1 \tag{4.5}$$

But by the Singleton bound, for any linear code,

$$d^* \leq n - k + 1 \tag{4.6}$$

Hence $d^* = n-k+1$, and $d^* = d_o$. Codes satisfying the Singleton bound with equality are called MDS codes. RS codes are MDS. This is an important property for the performance analysis of Reed-Solomon codes because it makes the explicit calculation of the weight distribution of the code possible.

Calculating the weight distribution of a code is a difficult problem and is unsolved for most codes, but for the important case of Reed-Solomon codes (or any MDS code), an exact solution is known. If A_l denote the number of codewords of weight l in an (n, k) linear code, the $(n+1)$ dimensional vector with components A_l for $l = 0, 1, \ldots, n$ gives the complete weight distribution. Obviously, if the minimum distance is d^*, then $A_0 = 1$ and $A_1, \ldots, A_{d^*-1}$ are all zero. A_l for $l \geq d^*$ is given by [10]

$$A_l = \binom{n}{l}(q-1)\sum_{j=0}^{l-d^*}(-1)^j\binom{l-1}{j}q^{l-d^*-j} \tag{4.7}$$

4.1.1 Error Probability Calculation on Reed-Solomon Codes

In this section, closed form expressions for the word, symbol and bit error rates of Reed-Solomon (RS) codes are presented for known symbol error rate P_s at the decoder input.

Consider a RS code over $GF(q)$. This code has an alphabet size of $q = 2^m$ for some integer $m \geq 1$ and its block size is $n = q - 1$. If the minimum distance is given by $d^* = 2t + 1$, then this code is able to correct any pattern of t symbol errors.

In this section, a symmetric q-ary memoryless channel with conditional probabilities

$$p(y/x) = \begin{cases} 1 - P_s & y = x \\ \frac{P_s}{n-1} & y \neq x \end{cases} \tag{4.8}$$

is assumed. Then, assuming a BDD the word error probability is given by

$$P_{w,out} = \sum_{i=t+1}^{n}\binom{n}{i}P_s^i(1-P_s)^{n-i} \tag{4.9}$$

Under the worst case assumption that a received word with i symbol errors will be decoded into a codeword at a distance $i+t$ from the transmitted codeword, an upperbound on the symbol error probability can be calculated as

$$P_{s,out} \leq \sum_{i=t+1}^{n}\frac{i+t}{n}\binom{n}{i}P_s^i(1-P_s)^{n-i} \tag{4.10}$$

Finally, utilizing a factor of $1/2$ to account for the average number of information bit errors per symbol error, the bit error rate at the RS decoder output may be estimated from the bound,

$$P_{b,out} \leq \frac{1}{2}\sum_{i=t+1}^{n}\frac{i+t}{n}\binom{n}{i}P_s^i(1-P_s)^{n-i} \tag{4.11}$$

The performance of a specific concatenation scheme can be determined by estimating P_s for the particular inner code of interest and then evaluating P_b through Eq. 4.11

4.2 Results on Ideally Interleaved System

In this section performance of an ideally interleaved system is considered. In this case, the bit errors at the convolutional decoder output are assumed to be independent. Therefore, the symbol error rate at the RS decoder input can be calculated directly by

$$P_s = 1 - (1 - P_b)^m \tag{4.12}$$

for a Reed-Solomon code over $GF(2^m)$ which has m bit symbols. Given the symbol error rate, P_s, the word, symbol and bit error rates at the system output can be calculated using Eq. 4.9-Eq. 4.11.

In this section, the bit error rate at the convolutional decoder output will be calculated using the improved techniques of Chapter 3. We will refer to the methods described in Section 3.1 and Section 3.3 as Bound-1 and Bound-2, respectively. In Table 4.1, bit error rates obtained through simulation, Bound-1, Bound-2, and classical union bound are listed for the rate-1/2, constraint-length-7 convolutional inner code over AWGN channel with soft decision decoding. In the calculation of classical union bound and Bound-2, referring to [22], following portion of the path enumerator is used,

$$\begin{aligned} T'(D) &= \left.\frac{dT(D,I)}{dI}\right|_{I=1} \\ &= 36D^{10} + 211D^{12} + 1404D^{14} + 11633D^{16} + 77433D^{18} + 502690D^{20} \\ &\quad +3322763D^{22} + 21292910D^{24} + 1343663911D^{26} + \ldots \end{aligned} \tag{4.13}$$

Using the data of Table 4.1, Eq. 4.12 and Eq. 4.11 is used to calculate the bit error rate of the overall system. The results are plotted in Fig. 4.1 and compared to a bit error rate plot given in [22]

Next, the same system will be considered over a BSC with hard decision decoding. In Table 4.2, bit errors rates obtained through simulation, Bound-1, Bound-2, and classical union bound are listed for the rate-1/2, constraint-length-7 convolutional inner code.

Using the data of Table 4.2, Eq. 4.12 and Eq. 4.11 are used to calculate the bit error rate of the overall system. The results are plotted in Fig. 4.2. It is observed that

E_b/N_0 in dB.	Simulation	Bound1	Bound2	Union Bound
1.7	0.017832	0.402543	0.079796	0.899823
1.8	0.014577	0.260885	0.065036	0.611950
1.9	0.011841	0.167522	0.052691	0.414405
2.0	0.009537	0.106812	0.042436	0.279584
2.1	0.007644	0.067805	0.033974	0.188034
2.2	0.006073	0.042989	0.027042	0.126149
2.3	0.004647	0.027324	0.021402	0.084486
2.4	0.003750	0.017480	0.016844	0.056531
2.5	0.002920	0.011296	0.013185	0.037825
2.7	0.001730	0.004923	0.007953	0.016992
2.9	0.000967	0.002290	0.004705	0.007711
3.1	0.000527	0.001130	0.002734	0.003555
3.3	0.000286	0.000580	0.001561	0.001669

Table 4.1: Bit error rates at the RS decoder input for AWGN channel and soft decision decoding

E_b/N_0 in dB.	Simulation	Bound1	Bound2	Union Bound
3.5	0.019946	0.747797	0.027643	2.725300
3.6	0.016832	0.519286	0.024305	1.900948
3.7	0.014116	0.357821	0.021289	1.319518
3.8	0.011831	0.244900	0.018574	0.911748
4.0	0.008181	0.112930	0.013967	0.429993
4.2	0.005537	0.051483	0.010329	0.200104
4.4	0.003691	0.023521	0.007505	0.092310
4.6	0.002409	0.010924	0.005355	0.042450
4.8	0.001546	0.005213	0.003749	0.019582
5.0	0.000973	0.002580	0.002573	0.009118
5.2	0.000601	0.001323	0.001702	0.004307
5.4	0.000364	0.000697	0.001070	0.002070

Table 4.2: Bit error rates at the RS decoder input for BSC channel and hard decision decoding

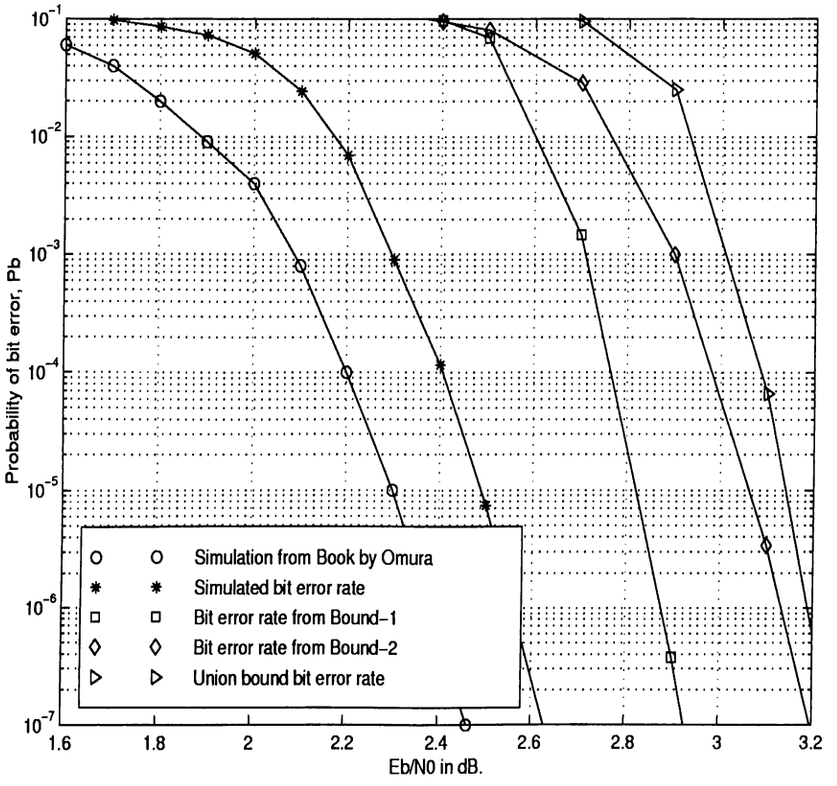

Figure 4.1: Bit error probability of ideally interleaved concatenated system over AWGN channel with soft decision

using the improved estimates of Chapter 3 for the bit error rate at the RS decoder input provides a better estimate for the overall system performance. At high SNR, all bounds eventually converge to the bound obtained by utilizing the classical union bound

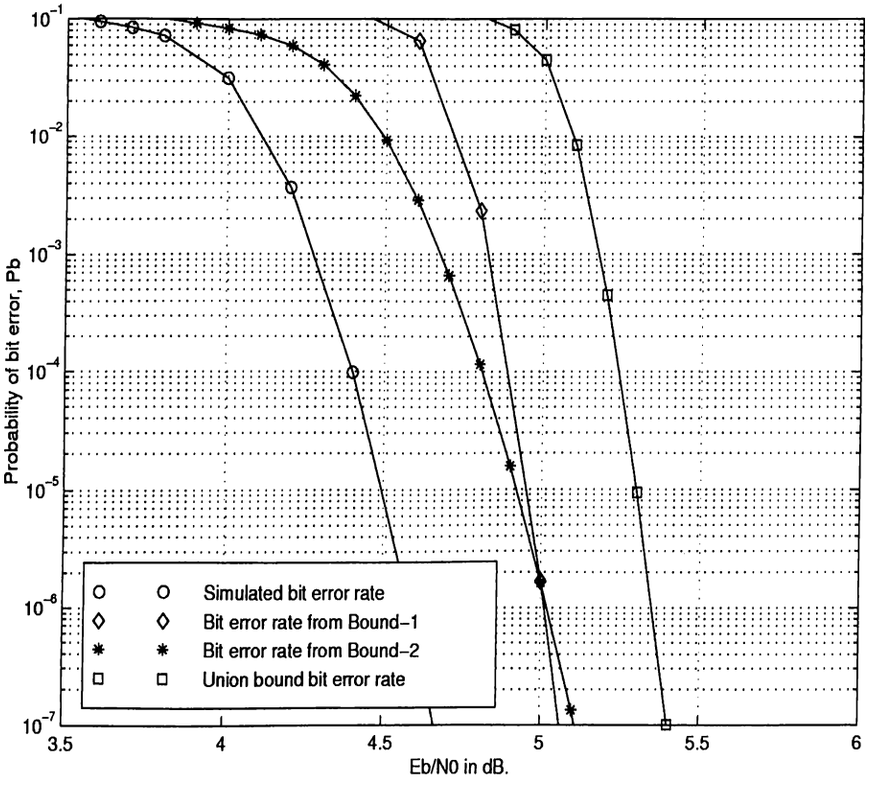

Figure 4.2: Bit error probability of ideally interleaved concatenated system over BSC with hard decision

Chapter 5

Non-Interleaved System Performance

For ideally interleaved systems, bit errors at the RS decoder input are assumed to be independent. Hence, Eq. 4.12 is used to calculate the symbol error. However, for non-interleaved systems, bit errors at the RS decoder input are not independent due to convolutional decoder memory and Eq. 4.12 is generally not valid.

In this section, we propose a method to directly determine the symbol error rate at the system output of non-interleaved systems without referring to ideas of Section 4.1.

First, the distribution of the errors at the Viterbi decoder output is considered. We know that channel errors may be decoded into an error burst by the Viterbi decoder. Hence, individual errors in an error burst are highly dependent. We must capture this dependence to successfully analyze a non-interleaved system. This requires a burst definition. A possible definition is to call any sequence of binary digits that start and end with a '1' digit and is separated from other such sequences by at least g successive '0' digits as a burst. There are two disadvantages of this definition. The first one is its dependence on g. Changing the value of g would result in a different burst distribution for the same sequence. Secondly, bursts are defined in terms of the binary output of the Viterbi decoder. However, the RS decoder stage works on this output in terms of symbols. Therefore, the binary output of the Viterbi decoder is grouped into symbols.

Then, assuming that all-zero message is transmitted, any L consecutive non-zero symbols is called a *burst* of length L (Fig. 5.1). With this definition extensive simulations are run to obtain burst histograms at the RS decoder input for a range of signal-to-noise ratios. Experiments are repeated several times to ensure the sample mean to lie in an acceptable confidence interval. The number of error free symbols between two successive bursts is called the *waiting time*. The waiting time distribution is also obtained by running simulations.

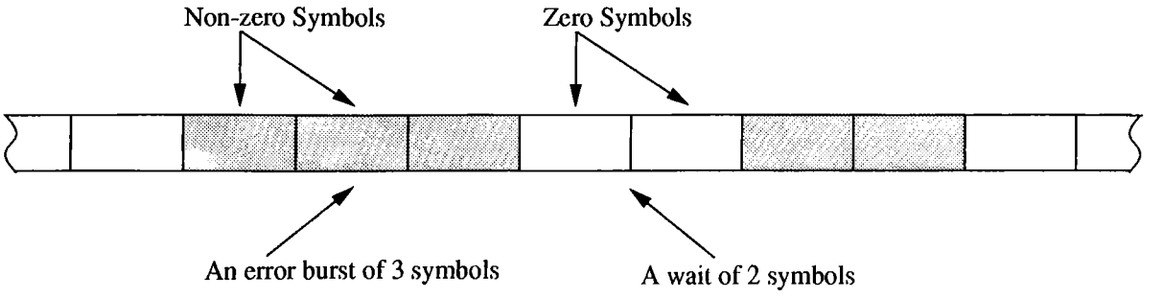

Figure 5.1: An example burst and wait structure

In Fig. 5.2 and Fig. 5.3 sample burst histograms for two different SNR values are plotted. The system considered employs a rate-1/2, constraint-length-7 convolutional inner code and a $(255, 223)$ RS code over $GF(2^8)$ for an AWGN channel with BPSK modulation. Viterbi decoding with soft decisions is performed at the inner decoder. Each RS symbol consists of 8-bit symbols.

From the graphs its is observed that the burst distributions can be approximated by a geometric distribution of the following form

$$P\{L = m\} = p(1-p)^{m-1} \quad , m = 1, 2, \ldots \tag{5.1}$$

where

$$p = \frac{1}{\overline{L}} \tag{5.2}$$

and $\overline{L}$ is the average burst length in symbols.

In Fig. 5.2 and Fig. 5.3 sample wait distributions are plotted for the same SNR values and system configuration. Wait distributions can also be approximated by a geometric distribution given by

$$P\{W = k\} = q(1-q)^{k-1} \quad , k = 1, 2, \ldots \tag{5.3}$$

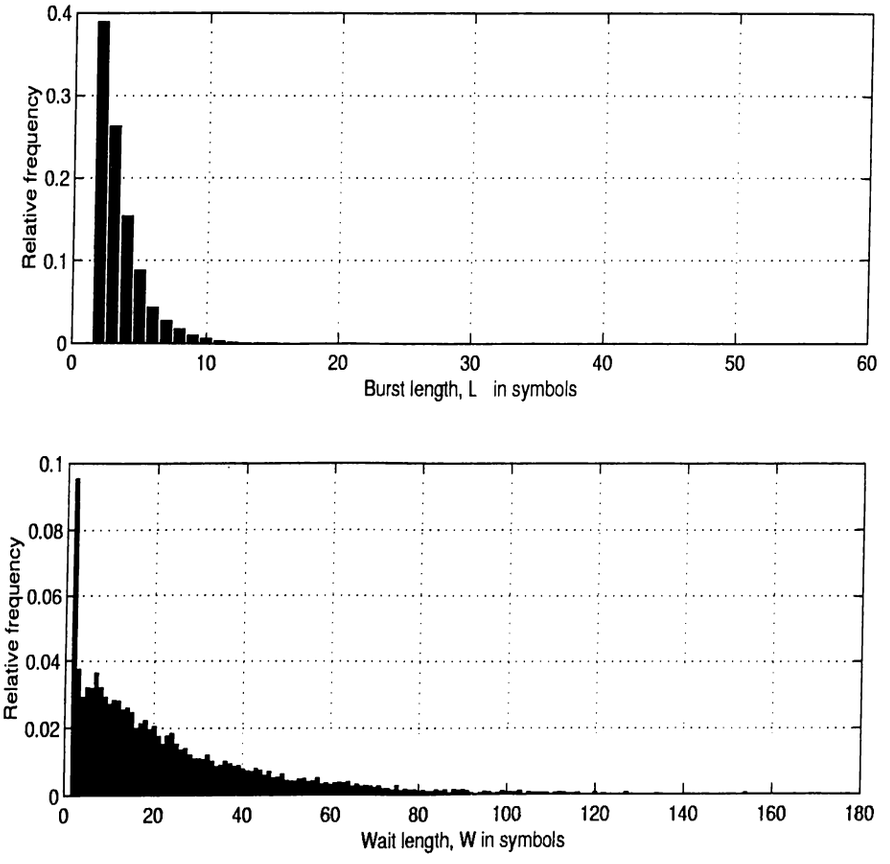

Figure 5.2: Burst and wait histograms for $E_b/N_0 = 1.2$dB.

where

$$q = \frac{1}{\overline{W}} \tag{5.4}$$

and $\overline{W}$ is the average wait length in symbols.

We adopt a two-state Markov Chain (MC) to represent the behavior of symbol errors at the RS decoder input. This MC representation is shown in Fig. 5.4. The state "0" represents the event of being in a waiting time and the state "1" represents the event of being in a bursty region.

The state transition matrix of this Markov Chain is given by

$$P = \begin{pmatrix} 1-q & q \\ p & 1-p \end{pmatrix} \tag{5.5}$$

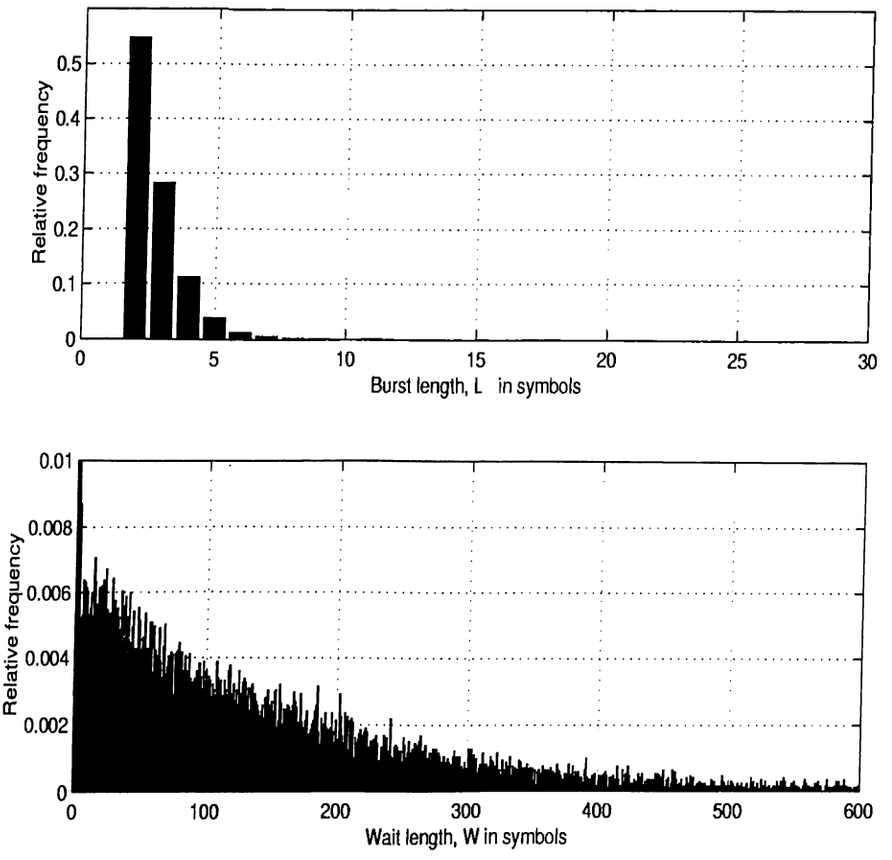

Figure 5.3: Burst and wait histograms for $E_b/N_0 = 2.3$dB.

and has the steady-state distribution, $\underline{\pi}$, given by

$$\underline{\pi}^t P = \underline{\pi} \tag{5.6}$$

where $\underline{\pi} = [\pi_0, \pi_1]^t$, and π_0 and π_1 are the steady state probabilities of being in the "0" and "1" states, respectively.

The symbol and word error probabilities at the decoder output can be obtained directly from the Markov Chain representation. Note that whenever a transition from the "0" to "1" state takes place, it initiates a symbol error. Any transition afterwards from "1" onto itself causes successive symbol errors. By keeping track of the state transitions, one can count the number of symbol errors. For this purpose, we introduce

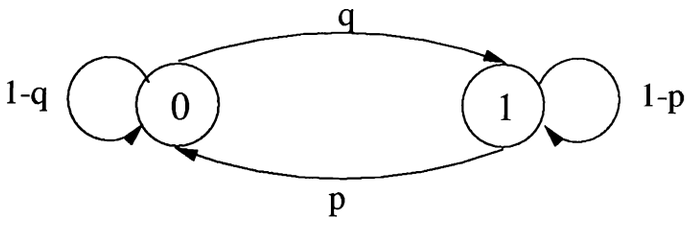

Figure 5.4: Markov Chain representation of symbol errors

a dummy variable L to the state transition matrix of Eq. 5.5 in the following form:

$$P = \begin{pmatrix} 1-q & qL \\ p & (1-p)L \end{pmatrix} \tag{5.7}$$

Consider a t error correcting RS code with block length n. If the behavior of the symbol errors at the decoder input is given by the above Markov chain structure, then the word and symbol error probabilities for this code can be calculated by the following procedure. Let P' be the nth power of P in Eq. 5.7. Then, elements of P' are polynomials in L. Sum of the first row elements give all the possible error patterns as powers of the polynomial variable L and their probabilities as the polynomial coefficients assuming that the system was initially in the "0" state. For example, let the sum of the first row elements be denoted by,

$$\begin{aligned} p'_0 &= p'_{00} + p'_{01} \\ &= c_0 + c_1 L^1 + c_2 L^2 + \ldots + c_n L^n \end{aligned} \tag{5.8}$$

Then, L^i for $i = 0, 1, \ldots, n$ denote the event of having i symbol errors in one codeword of n symbols and c_i denote the probability of having such a pattern. The same is true for the sum of the second row terms assuming that the system was initially at state "1". If the code is t error correcting then all error patterns having t or fewer symbol errors will be corrected. Therefore, all terms of degree less than $t+1$ are removed from the polynomials. If p''_0 denote the truncated form of p'_0, it is given by,

$$p''_0 = c_{t+1} L^{t+1} + c_{t+2} L^{t+2} + \ldots + c_n L^n \tag{5.9}$$

With above definitions, the probability of word error is given by

$$P_{w,out} \leq \pi_0 p''_0|_{L=1} + \pi_1 p''_1|_{L=1} \tag{5.10}$$

E_b/N_0 in dB.	$\overline{L}$	$\pm\Delta\overline{L}$	$\overline{W}$	$\pm\Delta\overline{W}$
1.1	2.50973	0.00724	19.44068	0.01038
1.2	2.39737	0.00747	22.63018	0.01881
1.3	2.33196	0.00815	25.30050	0.02579
1.4	2.23887	0.00746	29.76649	0.01836
1.5	2.17898	0.00741	33.68965	0.06362
1.6	2.09512	0.01009	40.28850	0.15253
1.7	2.04241	0.01297	46.04570	0.23528
1.8	1.97804	0.00825	54.29409	0.18050
1.9	1.92306	0.00578	64.48241	0.03363
2.0	1.86914	0.00583	76.66514	0.16515
2.1	1.82067	0.00342	90.94357	0.12123
2.2	1.77498	0.00288	107.05919	0.14933
2.3	1.72612	0.00195	126.18502	0.11852

Table 5.1: Mean burst and wait lengths in symbols for AWGN channel with soft decision decoding

Similarly, a bound on probability of symbol error can be obtained using Eq. 5.9 as

$$P_{s,out} \leq \frac{1}{n}\left[\pi_0 \frac{dp_0''}{dL}|_{L=1} + \pi_1 \frac{dp_1''}{dL}|_{L=1}\right] \tag{5.11}$$

We have applied this method to a non-interleaved concatenated system with a rate-1/2, constraint-length-7 convolutional inner code and a (255, 223) RS outer code. First, the system performance over an AWGN channel is considered. In order to obtain burst and wait time histograms, extensive simulations are run using a Viterbi software decoder written in C language. This is the same program used in Chapter 3 with slight modifications to count burst occurrences in terms of 8-bit symbols. Also a module is added to experimentally calculate the RS symbol and word error rates at the system output.

In Table 5.1 experimental data obtained for the system of our interest is listed for a range of SNR values. Sufficient experiments have been made to guarantee that the mean burst and wait time values lie in an interval of $\Delta\overline{L}$ and $\Delta\overline{W}$, respectively, with %95 probability. Using our model, word and symbol error probabilities are calculated. Results are tabulated in Table 5.2. In Table 5.3 corresponding simulation data are listed with their confidence intervals.

Data are plotted in Fig. 5.5. It is observed that calculated values closely follow the

E_b/N_0 in dB.	π_0	π_1	$P_{w,out}$	$P_{s,out}$
1.1	0.88566	0.11434	0.92413	0.11080
1.2	0.90420	0.09580	0.82325	0.08724
1.3	0.91561	0.08439	0.72028	0.07104
1.4	0.93005	0.06995	0.54146	0.04912
1.5	0.93925	0.06075	0.40661	0.03516
1.6	0.95057	0.04943	0.24033	0.01970
1.7	0.95753	0.04247	0.15148	0.01206
1.8	0.96485	0.03515	0.07857	0.00608
1.9	0.97104	0.02896	0.03724	0.00282
2.0	0.97620	0.02380	0.01630	0.00121
2.1	0.98037	0.01963	0.00686	0.00050
2.2	0.98369	0.01631	0.00284	0.00021
2.3	0.98651	0.01349	0.00109	0.00008

Table 5.2: Calculated word and symbol error probabilities at the system output for AWGN channel case

E_b/N_0 in dB.	$P_{w,out}$	$\pm\Delta P_{w,out}$	$P_{s,out}$	$\pm\Delta P_{s,out}$
1.1	0.921620	0.002490	0.110430	0.000420
1.2	0.820100	0.007480	0.086780	0.000410
1.3	0.716670	0.004080	0.070610	0.000430
1.4	0.538810	0.003660	0.048750	0.000360
1.5	0.404440	0.003300	0.034930	0.000300
1.6	0.239330	0.002530	0.019560	0.000210
1.7	0.150160	0.002460	0.011910	0.000190
1.8	0.075980	0.001220	0.005850	0.000100
1.9	0.035160	0.008860	0.002650	0.000070
2.0	0.014320	0.000450	0.001060	0.000030
2.1	0.005080	0.000160	0.000370	0.000010
2.2	0.001680	0.000080	0.000120	0.000006
2.3	0.000460	0.000002	0.000032	0.000002

Table 5.3: Simulation data for the word and symbol error probabilities at the system output

simulation results. The deviation at high SNR results from the fact that the geometric distribution model fails to correctly represent the error distribution at the RS decoder input. This is an expected outcome since at high SNR, symbol errors becomes independent of each other and hence distributed more uniformly. For high SNR values, the system performance resembles that of an interleaved system.

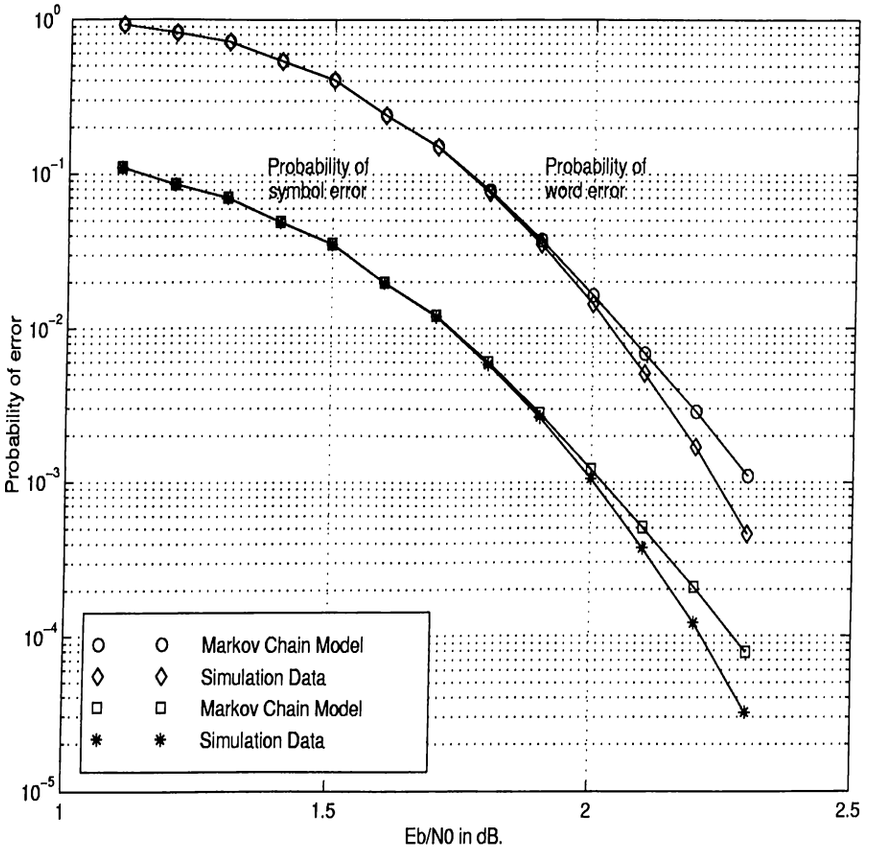

Figure 5.5: Word and symbol error probabilities for non-interleaved concatenated system over AWGN channel and soft decision

Next, a BSC with hard decision decoding will be considered. Similar to the previous case, extensive simulations are run to obtain burst and wait histograms at the Viterbi decoder output. The results are tabulated in Table 5.4. Using the same modeling, word and symbol error probabilities at the system output are obtained and tabulated in Table 5.5.

E_b/N_0 in dB.	$\overline{L}$	$\pm\Delta\overline{L}$	$\overline{W}$	$\pm\Delta\overline{W}$
2.8	2.86325	0.01562	19.13397	0.09689
3.0	2.69665	0.01000	23.89189	0.02076
3.2	2.54389	0.01015	30.42012	0.05032
3.3	2.47844	0.00587	34.53128	0.03547
3.4	2.41326	0.01454	39.38995	0.19000
3.5	2.35788	0.01244	45.00660	0.20661
3.6	2.30210	0.00780	51.88436	0.07805
3.7	2.25012	0.00664	60.14580	0.05249
3.8	2.20275	0.00733	69.81994	0.10668
4.0	2.11480	0.00331	95.57575	0.10965
4.2	2.03820	0.00291	133.247084	0.11678
4.4	1.97282	0.00300	185.556340	0.13410

Table 5.4: Mean burst and wait lengths in symbols for BSC with hard decision decoding

E_b/N_0 in dB.	π_0	π_1	$P_{w,out}$	$P_{s,out}$
2.8	0.86984	0.13016	0.95644	0.12838
3.0	0.89858	0.10142	0.84296	0.09396
3.2	0.92283	0.07717	0.62339	0.06004
3.3	0.93033	0.06697	0.49311	0.04495
3.4	0.94227	0.05773	0.36482	0.03176
3.5	0.95022	0.04978	0.25612	0.02151
3.6	0.95752	0.04248	0.16639	0.01355
3.7	0.96394	0.03606	0.10139	0.00805
3.8	0.96942	0.03058	0.05910	0.00460
4.0	0.97835	0.02165	0.01720	0.00130
4.2	0.98493	0.01507	0.00432	0.00032
4.4	0.98948	0.01052	0.00106	0.00008

Table 5.5: Calculated word and symbol error probabilities at the system output for BSC case

E_b/N_0 in dB.	$P_{w,out}$	$\pm\Delta P_{w,out}$	$P_{s,out}$	$\pm\Delta P_{s,out}$
2.8	0.964280	0.001726	0.128255	0.000325
3.0	0.857820	0.003523	0.094193	0.000493
3.2	0.636380	0.004436	0.060008	0.000446
3.3	0.499433	0.002448	0.044485	0.000233
3.4	0.362880	0.004330	0.030795	0.000369
3.5	0.245827	0.002501	0.020095	0.000208
3.6	0.152360	0.002000	0.012047	0.000162
3.7	0.085060	0.001403	0.006563	0.000110
3.8	0.045260	0.001045	0.003422	0.000079
4.0	0.009819	0.000139	0.000721	0.000010
4.2	0.001565	0.000063	0.000112	0.000004
4.4	0.000170	0.000023	0.000012	0.000002

Table 5.6: Simulation data for word and symbol error probabilities at the system output

In Table 5.6 corresponding simulation data are listed with their corresponding confidence intervals. Data are plotted in Fig. 5.6. Similar to the AWGN channel case, the model follows the simulation data closely but deviates slightly as moved to the high SNR case.

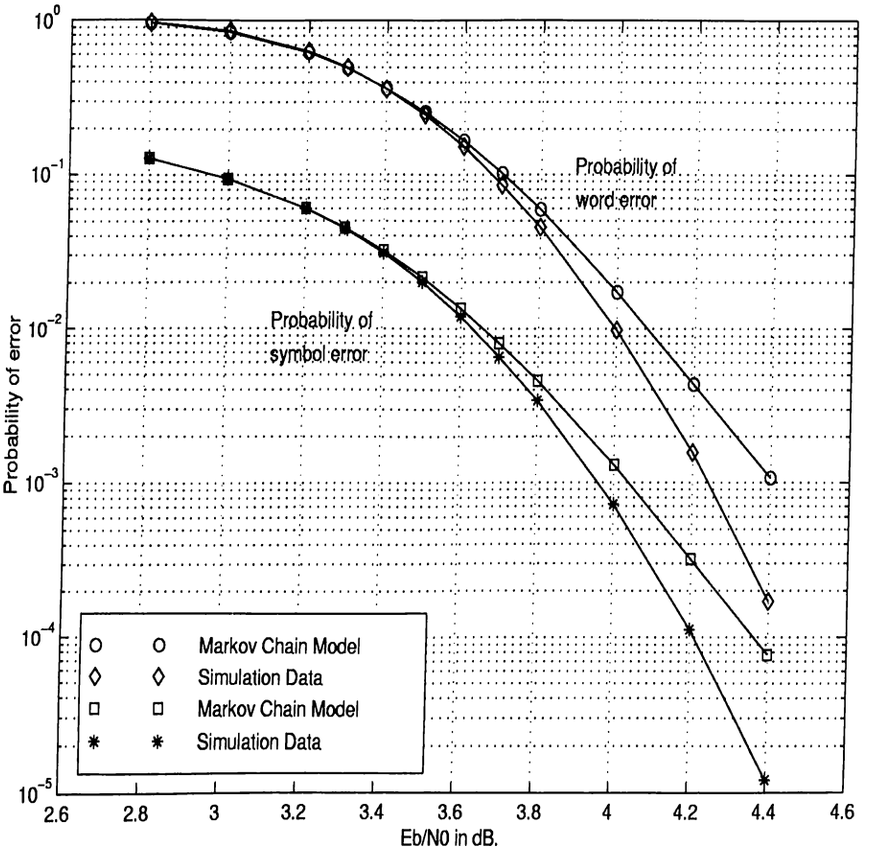

Figure 5.6: Word and symbol error probabilities for non-interleaved concatenated system over BSC and hard decision decoding

Chapter 6

Concluding Remarks

Performance of a particular coding scheme can be evaluated based on the output error rate. Computer simulations are an important tool for error rate estimation. However, their use has always been limited by the computational cost of getting a reliable statistical sample. Another possibility is to use error bounds to estimate the performance. Error bounds provide approximations to the exact behavior of the system through mathematical models. Obviously, any useful bound should be as tight as possible on the actual system performance and remain computationally simple.

In this thesis, we have concentrated on finding tight upperbounds on the output error rate of concatenated coding systems with convolutional inner codes and RS outer codes. Performance of such a system can be estimated by first calculating the error rate of the inner code and then by evaluating the outer code performance. Therefore, we have investigated bounds on the error rate of convolutional codes. Union bound on convolutional codes given by [11] is a well-known technique to estimate the error rate. Though it provides a very tight estimate on the error rate at relatively high SNR values, it increases abruptly at low SNR values.

Two different methods are proposed to improve the union bound on convolutional codes. The first method modifies the ideas presented in [3] for use with convolutional codes. Performance of the algorithm is tested over an AWGN channel with soft decisions decoding and a BSC channel with hard decisions decoding. It considerably improves the classical union bound in the low SNR region. Comparison with simulations show that an

order of magnitude improvement in the low SNR region is possible. At high SNR region the new bound converges to the classical union bound. Application of the algorithm requires the enumeration of the trellis paths. An effective approach based on calculation of successive powers of the state transition matrix of the code is proposed. The only difficulty is to decide on when to truncate the enumeration of the paths. The same problem exists in the calculation of the classical union bound when the transfer function of the code can not be obtained as a closed form expression. This is a highly code dependent decision. The only observation is that at high SNR region truncation at relatively short length paths is sufficient to get a good estimate since the probability that a path remain diverged from the all-zero path decreases in this region. However, for low SNR region longer path lengths are required.

The second approach aims to improve the classical union bound by avoiding the over counting of events with low occurrence probabilities. The method is tested over an AWGN channel with soft decisions decoding and a BSC channel with hard decisions decoding. It provides tight estimates in the low SNR region where the union bound mostly suffers from the over counting of events.

We have investigated the performance of an ideally interleaved system. Inner code performance is calculated using the the improved methods. Under the ideal interleaving assumption, bit error rate estimates at the convolutional decoder output are directly used to calculate the symbol error rate at the RS decoder input. The results show that having better estimates for the inner code performance directly effects the estimates on the overall system performance. The use of improved methods provide a two fold improvement over the SNR range when compared to estimates that use union bound to predict the inner code performance.

For the non-interleaved systems, bit error rates at the convolutional decoder output can not be used to compute the symbol error rate. Therefore we have proposed a new model for the analysis of non-interleaved systems. For this case, burst histograms should be obtained through simulation. Burst histograms are used to set up a Markov chain representation of the system. The results are very satisfactory. The model almost exactly represent the actual system performance over a considerable SNR range. The method shows deviation from the actual case as we move to high SNR region. This is because the underlying assumption of geometrically distributed error bursts fails to model the actual distribution. However, the methods proposed for the ideally interleaved system can be employed in the high SNR region as the errors become independent. The

main drawback of this method is the requirement for simulation based burst histograms. They are effected by the problems common to all simulation the based data such as long computational times and statistical accuracy. Analysis of the state diagrams may help in finding mathematical models to represent the burst histograms. This may decrease the computational cost of the algorithm.

Bibliography

[1] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. Journal*, vol. 27, pp. 379–423(Part I); 623–656(Part II), 1948.

[2] G. D. Forney, *Concatenated Codes.* Cambridge, Massachusetts: MIT Press, 1966.

[3] A. M. Viterbi and A. J. Viterbi, "Improved union bound on linear codes for the input-binary AWGN channel, with applications to turbo codes," in *Proceedings of ISIT 1998*, August 1998.

[4] S. J. Mason and H. J. Zimmerman, *Electronic Circuits, Signals and Systems.* New York: Wiley, 1960.

[5] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inf. Theory*, vol. IT-13, pp. 260–269, April 1967.

[6] A. J. Viterbi and J. K. Omura, *Principles of Digital Communication and Coding.* New York: McGraw-Hill, 1979.

[7] G. C. Clark and J. B. Cain, *Error-Correction Coding for Digital Communications.* New York: Plenum Press, 1988.

[8] R. G. Gallager, *Information Theory and Reliable Communication.* New York: Wiley, 1968.

[9] C. Schlegel, *Trellis Coding.* New York: IEEE Press, 1997.

[10] R. E. Blahut, *Theory and Practice of Error Control Codes.* Massachusetts: Addison-Wesley Publishing Company, 1984.

[11] A. J. Viterbi, "Convolutional codes and their performance in communication systems," *IEEE Trans. Commun. Technol.*, vol. COM-19, pp. 751–772, October 1971.

[12] L. V. D. Meerberg, "A tightened upper bound on the error probability of binary convolutional codes with viterbi decoding," *IEEE Transactions on Information Theory*, pp. 389–391, May 1974.

[13] K. A. Post, "Explicit evaluation of Viterbi's union bounds on convolutional code performance for the binary symmetric channel," *IEEE Transactions on Information Theory*, pp. 403–404, May 1977.

[14] J. Pieter, M.Schalkwijk, K. A. Post, and J. P. J. C. Aarts, "On a method of calculating the event error probability of convolutional codes with maximum likelihood decoding," *IEEE Transactions on Information Theory*, vol. IT-25, pp. 737–743, November 1979.

[15] M. Cedervall, R. Johannesson, and K. S. Zigangirgov, "A new upper bound on the first-event error probability for maximum-likelihood decoding of fixed binary convolutional codes," *IEEE Transactions on Information Theory*, vol. IT-30, pp. 762–766, September 1984.

[16] S. S. Pietrobon, "On the probability of error of convolutional codes," *IEEE Transactions on Information Theory*, vol. IT-42, pp. 1562–1568, September 1996.

[17] M. A. Best, M. V. Burnashev, Y. Levy, A. Rabinovich, P. C. Fishburn, A. R. Calderbank, and D. J. Costello, "On a technique to calculate the exact performance of a convolutional code," *IEEE Transactions on Information Theory*, vol. IT-41, pp. 441–447, March 1995.

[18] R. G. Gallager, *Low-density Parity-Check Codes.* MIT Press, 1963.

[19] "Viterbi decoder software." http://www.asti.dost.gov.ph/mia/ecc/codes.html.

[20] R. C. Bose and D. K. Ray-Chaudhuri, "On a class of error correcting binary group codes," *Inf. Control*, vol. 3, pp. 68–79, March 1960.

[21] A. Hocquenghem, "Codes corecteurs d'erreurs," *Chiffres*, vol. 2, pp. 147–156, 1959.

[22] M. K. Simon, J. K. Omura, R. A. Scholtz, and B. K. Levitt, *Spread Spectrum Communications Handbook.* New York: McGraw-Hill, Inc., revised ed., 1994.