INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

Université d'Ottawa · University of Ottawa

STUDY OF LOW RATE TURBO CODES:
Performance Comparison and Improvement

By

Davan Long, B. Eng

A thesis submitted to the
School of Graduate Studies and Research of the
University of Ottawa in partial fulfillment of the requirements
for the degree of Master of Applied Science in Electrical Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering
School of Information Technology and Engineering
Faculty of Engineering
University of Ottawa

Ottawa, Ontario, Canada
26 May 2000

I hereby declare that I am the sole author of this thesis. Furthermore, I authorize the University of Ottawa to lend this thesis to other institutions or individuals for the purpose of scholar research.

Davan Long

I further authorize the University of Ottawa to reproduce this thesis by photocopying or by other means, in part or in whole at the request of other institutions or individuals for the purpose of scholar research.

Davan Long

The University of Ottawa requires the signature of all persons using or photocopying this thesis. Please sign, date and provide your address below.

Acknowledgments

I wish to express a sincere gratitude to my director, Dr. J-Y Chouinard, for his encouragement, guidance and supervision. Over the past three years, he provided many valuable advises including the area of research and the related references, kept me aware of the latest published works in the field of error control coding, and helped me to stay on the right track. His mentoring, guidance and support were instrumental in ensuring the timely and successful completion of the thesis.

I would like to thank my wife, Ravine Ly, for her true love, support and putting up with me during these demanding years. Without her unconditional support and understanding, I would never be able to finish this work

Finally, I would like to express my utmost respect and gratitude to my mother, Porsrim Tork, who after the passing off my father has worked extremely hard to raise the children, flee them from the war torn country, and ensure that they receive adequate education in the host country. Her courage, dedication and determination are my greatest source of inspiration and the strength behind all my accomplishments.

Abstract

Turbo Codes have so far shown to outperform all known codes with comparable size and decoding complexity. These powerful codes, which can be constructed from simple component codes, have achieved a large coding gain closed to the theoretical limit set forth in Shannon's Theory. Because of their superior performance, Turbo Codes have spontaneously captured worldwide interests and have been a subject of many works and publications.

Most of the existing published research works on Turbo Codes are centered on the design and analysis of high rate Turbo Codes, i.e. whose overall code rates equal to 1/3 and higher. Furthermore, these research works mainly focus on analyzing and explaining the performance of Turbo Codes achievable with different component codes and decoding algorithms.

This thesis investigates the performance and a possible area of improvement to the original low rate Turbo Codes in the region of low SNR, and studies some potential variants for addressing such weakness. The thesis begins with examining the design and performance of the current low rate Turbo Codes, i.e., with code rate equal to 1/3 and below, and then determining the code rate which yields optimum performance at low SNR. In particular, it attempts to demonstrate through simulations that for a power limited transmission and low SNR, rate 1/3 Turbo Codes generally achieve the best performance, while reducing the code rate degrades rather than improves the performance. Theoretical explanation is then provided to shed some light on such performance degradation. Next, the thesis highlights a weakness of the original rate 1/3 Turbo Codes, and investigate some possible variants for addressing this weakness. Simulations are conducted to compare the performances of these variants with the current rate 1/3 Turbo Codes. Finally, the thesis concludes with a formulation of some important characteristics that concatenated codes should have in order to outperform the current rate 1/3 Turbo Codes.

Contents

List of Acronyms

APP	A Posteriori Probability
AWGN	Additive White Gaussian Noise
BCJR	Bahl, Cocke, Jelineck and Raviv
BER	Bit Error Rate
BSC	Binary Symmetric Channel
BPSK	Binary Phase Shift Keying
CC	Constituent or Component Code
CWEF	Conditional Weight Enumerating Function
ENC	Encoder
DEC	Decoder
GF	Galois Field
HCC	Hybrid Concatenated Code
HCBC	Hybrid Concatenated Block Code
HCCC	Hybrid Concatenated Convolutional Code
HDD	Hard Decision Decoding
IIR	Infinite Impulse Response
IOWEF	Input Output Weight Enumerating Function
LLR	Log-Likelihood Ratio
LTC	Low rate Turbo Codes
MAP	Maximum A Posteriory
ML	Maximum Likelihood
NSC	Non Systematic Convolutional
PCC	Parallel Concatenated Code
PCBC	Parallel Concatenated Block Code
PCCC	Parallel Concatenated Convolutional Code
RSC	Recursive Systematic Convolutional
SCC	Serial Concatenated Code
SCBC	Serial Concatenated Block Code
SCCC	Serial Concatenated Convolutional Code
SDD	Soft Decision Decoding or Soft Decision Decoder
SISO	Soft-In Soft-Out
SNR	Signal to Noise Ratio
SOVA	Soft Output Viterbi Algorithm
SPC	Single Parity Check
TC	Turbo Codes
VA	Viterbi Algorithm
WEF	Weight Enumerating Function

List of Symbols

A^{C_C}	Convolutional code input-output weight enumerating function
$A_d^{C_{SCBC}}$	Number of SCBC codewords of weight d
$A_{w,d}^{C_{SCBC}}$	Number of SCBC codewords of weight d produced by weight w input words
$A_{w,d}^{C_{SCCC}}$	Number of SCCC codewords of weight d produced by weight w input words
$A_{w,d,j}^{C_{SCCC}}$	$A_{w,d}^{C_{SCCC}}$ with j error events
$A_{d_j}^{TC}$	Number of TC codewords with weight d_j
$\mathbf{C}_B$	Linear block codes
$\mathbf{C}_C$	Convolutional codes
$\mathbf{C}_I$	Inner codes
$\mathbf{C}_O$	Outer codes
$\mathbf{C}_{PCBC}$	Parallel concatenated block codes
$\mathbf{C}_{PCC}$	Parallel concatenated codes
$\mathbf{C}_{SCBC}$	Serial concatenated block codes
$\mathbf{C}_{SCC}$	Serial concatenated codes
$\mathbf{C}_{TC}$	Turbo codes
d^{ENC}	Hamming weight of the parity sequence generated by encoder ENC
d^{PCBC}	Hamming distance or weight of PCBC
d^{SCBC}	Hamming distance or weight of SCBC

d_f^o	Free distance of SCC outer codes
d_{free}^{TC}	Free distance of Turbo Codes
d_j^{TC}	Hamming weight of TC codeword produced by a weight j input sequence
$d_{j,min}^{TC}$	Minimum weight of TC codewords produced by weight j input sequences
E_B	Energy per information bit
E_S	Energy per coded bit
E_{SP}	Transmitted energy per parity bit
E_{SS}	Transmitted energy per systematic bit
G^{PCBC}	PCBC asymptotic coding gain
G^{SCBC}	SCBC asymptotic coding gain
G_c	Generator polynomial of codes C
$g_{c,i}$	Coefficient i of G_c
$L(x)$	Soft value or LLR of x
L_C	Channel reliability
$L_E(x)$	Extrinsic information of x
$L_{\text{SOVA}}(\tilde{v}_k)$	Soft decoded v_k produced by SOVA
$M_k^{(\rho)}$	The metric of the path ρ at a relative time index $t = k$
N	Interleaver length
$N_{j,min}^{TC}$	Number of TC codewords of weight $d_{j,min}^{TC}$
P_B	Bit error probability
P_C	Channel cross-over probability
$Q(x)$	Error complement function
R	Code rate
R_C	Convolutional code rate
R_{PCBC}	Code Rate of parallel concatenated block codes
R_{SCBC}	Code rate of serial concatenated block codes

$\boldsymbol{S}^{(\rho)}$	A sequence of encoder states $(S_1^{(\rho)}, S_2^{(\rho)}, ..., S_N^{(\rho)})$ associated with a path ρ
$\boldsymbol{S}_{t\leq k}^{(\rho)}$	A sequence of k encoder states $(S_1^{(\rho)}, ..., S_k^{(\rho)})$ associated with a path ρ
$S_{w=j}$	Set of all input sequences or words of weight $w = j$
$S_{w=j}^{2}$	Set of all weight-2 parity sequences produced by input sequences of weight j
u_k	k^{th} information bit
$\bar{u_k}$	Decoded u_k
x_k^i	Transmitted coded bit produced by ENC i of the conventional TC
$\boldsymbol{X}^{(\rho)}$	A sequence $(X_1, X_2, \ldots, X_N)$ associated with a path ρ
y_k	Corrupted or received u_k
z_k^i	Received x_k^i
w	Hamming weight of an input sequence
σ^2	Noise variance

List of Figures

Chapter 1

Introduction

1.1 Background

In digital communications, information is often transmitted over a noisy channel which can introduce errors to the received data. To increase the reliability of the received data, controlled redundancy is added to the transmitted data stream, allowing the receiver to detect and possibly correct errors. Such technique is known as error control coding or channel coding.

The field of error control coding began with the second Shannon's paper published in 1948 [46]. In that paper, Shannon introduced the concept of channel capacity which can be associated with every channel. He then showed that as long as the information stream is transmitted at a rate below the channel capacity, there exist error control codes such that information stream can be transmitted and received at an arbitrarily low bit error rate. He did not however specify what these codes are or how to derive them.

From 1948 to 1993, a large number of error control codes have been developed for deep space communications, yet none of which has performance close to that set forth in Shannon's theory. It is well known since the early days of Shannon theory that increasing the codeword length N would yield better performance. As such, large codes were developed in an attempt to increase the performance. Unfortunately, the decoding complexity of such codes also increases with N up to a level where decoding becomes physically unrealizable [12].

The way one understands coding and resolves decoding problems have completely been revolutionized with the advent of Turbo Codes. Introduced in 1993 by Berrou, Glavieux and Thitimajshima [1], Turbo Codes have surprisingly achieved a BER performance close to the Shannon's promise. So far, Turbo Codes have a strength unmatched by any known code in the sense that they can provide highly reliable data transmission with moderate decoding complexity at low signal to noise ratios. The outstanding BER performance of Turbo Codes can be attributed to the revolutionized and powerful iterative decoding, and the application of the soft decoding algorithm introduced by Bahl, Cocke, Jelinek and Raviv in 1972 [4] to iteratively reduce the decoding bit error rate.

1.2 Motivation

A Turbo Code (TC) codeword generally comprises one systematic and several parity segments. Obviously, the larger the number of the parity segments, the lower the corresponding TC code rate and the less transmitted power available per coded bit. As TC are particularly designed to work in the region of low SNR, it would be interesting to find out the optimum code rate which yields the best BER performance at low SNR. This is one of the several motivations of the thesis. Another motivation is related to the observation that the high performance of TC is achieved by iteratively decoding and improving the reliability of the systematic segment of the received codeword. However, the parity segments which are used to improve the systematic segment are not protected against the channel noise, and as such the overall TC performance can be adversely affected by the low reliability of the parity segments. This view leads to the following postulation which is investigated throughout the thesis.

> " *The BER performance of a TC can be improved if the reliability values of the corresponding individual parity segments are enhanced without actually decreasing the reliability values of the systematic segment or/ and parity diversity .*"

1.3 Contributions

This thesis attempts to make contributions to the field of error control coding, specifically Turbo Codes, by conducting a detailed study on the behaviors of the low rate Turbo Codes whose constituent codes are rate 1/2 recursive systematic convolutional (RSC) codes. The study includes simulating and reporting different impacts of critical parameters such as code rate, puncturing, decoding delay and complexity on the overall BER performance in the region of low SNR. Furthermore, it suggests some changes to improve the reliability of the parity bits of the original Turbo Codes, and investigates the BER performances of the modified Turbo Codes referred to as TC variants.

The BER performances of these TC variants were simulated and compared to those of the original rate 1/3, 1/4 and 1/5 Turbo Codes. As part of the simulations, a modified SOVA was also developed to decode one of the proposed TC variants. Finally, the simulation results obtained and the conclusions drawn from these TC variants are used to conceptually highlight or formalize the characteristics of a new class of concatenated codes which could be viewed as a superset of the original Turbo Codes and which have the potential to outperform the original Turbo Codes across the entire range of SNR.

1.4 Thesis Outline

The remaining of the thesis is organized in three chapters and one appendix. Chapter 2 begins with a detailed coverage on the design and analysis of Turbo Codes from the theoretical stand point. Emphasis is on the analysis of Turbo Codes with low rates, particularly the rate 1/3, 1/4 and 1/5. The analysis covers the overall Turbo Codes encoding and decoding operations, the roles of the basic elements forming the Turbo Codes such as component codes and interleavers. The theoretical bounds on the BER is loosely derived to predict the overall performance versus the code rate for a given range of signal to noise ratio. The performances of various Turbo Codes are simulated to illustrate the BER as a function of several independent parameters such as the TC code rate, the interleaver length, the generator polynomials of the constituent codes and the number of decoding iterations.

Chapter 3 represents the most important contribution of the thesis. It discusses a possible area of improvement for the original Turbo Codes, introduces some potential TC variants and investigates their BER performances. As part of the investigation, we propose and develop a modified SOVA which independently minimizes the BER of the individual inputs of a rate 2/3 convolutional codes. The findings from the study of the proposed TC variants are then used to describe the characteristics of superior concatenated codes which can outperform the current rate 1/3 TC.

Chapter 4 concludes the thesis with a summary of findings, and a postulation or formulation of the characteristics of superior concatenated codes which can outperform the current rate 1/3 TC across the entire range of SNR. The detailed development of these superior codes is however left to the future research works.

The appendix provides an introduction to concatenated codes. Various concatenating schemes including serial and parallel concatenations are highlighted. Initial emphasis is on the overview of different concatenated codes, followed by a detailed discussion of serial concatenated codes. Parallel concatenated codes is briefly discussed in the generic context in order to draw some comparison with the serial concatenated codes. The appendix is completed with an introduction to the principle of iterative decoding which is essential to the understanding of the Turbo Codes.

Chapter 2

Turbo Codes

2.1 Introduction

Turbo Codes were introduced by Berrou, Glavieux and Thitimajshima in a 1993 conference paper entitled "Near Shannon Limit Error Correcting Coding and Decoding: Turbo Codes" [1]. These codes can be viewed as a very successful application of two important coding ideas – concatenated coding scheme proposed by Forney in 1966 [6], and replication decoding proposed by Battail in 1979 [5]. Constructed from two simple convolutional codes and an interleaver, Turbo Codes can achieve a performance unmatched by any previously known codes at a moderate decoding complexity. This stunning performance has caused many coding theorists to review their understanding of efficient codes and to consider a new approach to solving decoding problems.

This chapter provides a comprehensive study of Turbo Codes with an emphasis on both theoretical analysis and simulations. The theoretical analysis begins with a brief introduction to the important characteristics or components of Turbo Codes (TC) as well as the architectures of the associated encoder and decoder from the overall design view point. It then proceeds to carry out an in depth analysis on the encoding and decoding operations to establish some performance optimization criterion and performance bounds. Several simulations are performed to illustrate the BER performance of low rate TC as a function of several independent parameters including the interleaver length, the generator polynomial of the associated constituent codes, decoding iterations and the TC code rate. The results of these simulations are then discussed in detail.

2.2 Turbo Encoding and Decoding Principle

Turbo Codes (TC) represent a sub-group of parallel concatenated codes (PCC), namely the parallel concatenated convolutional codes (PCCC). For instance, a rate 1/3 TC basically contains two recursive systematic convolutional codes, RSC1 and RSC2, concatenated in parallel. A typical rate 1/3 TC encoder is depicted in figure 2.1 It is constructed from two identical recursive systematic convolutional (RSC) encoders, ENC1 and ENC2, linked by an interleaver. This particular TC encoder can be viewed as a ($3N$, K) linear encoder in which a stream of K information bits $\boldsymbol{u} = (u_1\ u_2\ u_3 \ldots\ u_K)$ is first padded by (N-K) tail bits and then encoded by the two convolutional encoders ENC1 and ENC2. The (N-K) tails bits are required for the trellis termination, i.e. forcing the two encoders back to the all-zeros state. In practice, it is very difficult to force both encoders back to all-zeros state with (N-K) tail bits, and as such only the encoder ENC1 is usually forced back to all-zeros state. The overall code rate for the TC encoder is $R = K/3N$, where in practice K is often very large compared to the number of tail bits (N-K), therefore the code rate R can conveniently be approximated to $R \sim 1/3$. For clarity, it should be pointed out that two encoders, ENC1 and ENC2, only output the parity sequences of the corresponding RSC1 and RSC2 which constitute the rate 1/3 TC.

The overall code rate of the TC encoder can be increased, up to unity, through puncturing, a technique which periodically removes the parity bits from ENC1 and ENC2 encoders. Puncturing has some useful applications in telecommunication systems where unequal error protection is required. For instance, during data transmission, synchronization frames require more protection than other frames. As such, puncturing is disabled for the synchronization frames to provide adequate error protection but enabled for other frames to increase the overall code rate. Likewise, as the channel becomes noisier, the encoder can stop puncturing to provide more protection to the transmitted data. In this thesis, only non-punctured Turbo Codes whose rates are 1/3 and lower are considered.

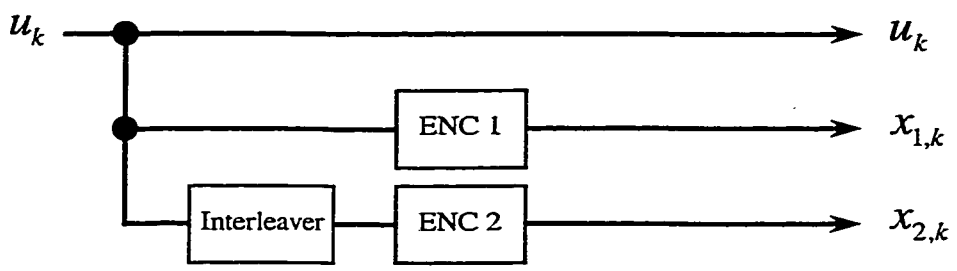


Figure 2.1: A typical rate 1/3 Turbo Codes encoder

As far as encoding is concerned, the TC encoder of figure 2.1 encodes a data block of $K = (N - m)$ bits where m denotes the number of tail bits into a codeword consisting of three distinct components of N bits each, namely one systematic component which is the replica of the input block and two parity sequences. In that context, an information sequence or input block $\boldsymbol{u} = (u_1, u_2, u_3, \ldots, u_k, \ldots u_N)$ is encoded twice by the two identical recursive systematic convolutional encoders ENC1 and ENC2, where the second encoding takes place only after interleaving the input block. There is an important reason for using identical ENC encoders to construct a TC encoder. That reason will become relevant later: for the moment one may consider that a coincidental advantage of using identical constituent encoders is low implementation complexity for the TC encoder and decoder. Furthermore, since both ENC1 and ENC2 are systematic encoders, only one systematic sequence needs to be transmitted, hence reducing the system bandwidth.

Suppose that the systematic component $\boldsymbol{u} = (u_1, u_2, u_3, \ldots, u_k, \ldots u_N)$, and the two parity components $\boldsymbol{x_1} = (x_{1,1}, x_{1,2}, x_{1,3}, \ldots, x_{1,k}, \ldots, x_{1,N})$ from ENC1 and $\boldsymbol{x_2} = (x_{2,1}, x_{2,2}, x_{2,3}, \ldots, x_{2,k}, \ldots, x_{2,N})$ from ENC2 are multiplexed and then transmitted over an AWGN channel using bipolar waveforms. Assume that the triplet $(\boldsymbol{u}, \boldsymbol{x_1}, \boldsymbol{x_2})$ is corrupted and received as $(\boldsymbol{v}, \boldsymbol{y_1}, \boldsymbol{y_2})$, where $\boldsymbol{v} = (v_1, v_2, v_3, \ldots, v_k, \ldots, v_N)$, $\boldsymbol{y_1} = (y_{1,1}, y_{1,2}, y_{1,3}, \ldots, y_{1,k}, \ldots, y_{1,N})$ and so on. Expressing v_k and $y_{j,k}$ in bipolar forms, one obtains

$$v_k = (2u_k - 1) + n_{ik} \tag{2.1}$$

$$y_{j,k} = (2x_{j,k} - 1) + n_{jk} \tag{2.2}$$

where n_{ik} and n_{jk} ($j = 1, 2$) are independent noise samples having the same variance σ^2. Using eq. (A.21) and (A.22) of Appendix A, the channel output, or soft value associated with the received v_k, can be expressed as [14]:

$$\begin{aligned} v_k \cdot L_C &= \ln\left[\frac{p(v_k \mid u_k = +1)}{p(v_k \mid u_k = -1)}\right] \\ &= \ln\left[\frac{\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{1}{2}\left[\frac{v_k - 1}{\sigma}\right]^2\right)}{\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{1}{2}\left[\frac{v_k + 1}{\sigma}\right]^2\right)}\right] \\ &= \frac{2}{\sigma^2} v_k \end{aligned} \tag{2.3}$$

Similarly, it can be shown

$$y_{j,k} \cdot L_C = \frac{2}{\sigma^2} x_{j,k} \tag{2.4}$$

The received codeword (v, y_1, y_2) can be iteratively decoded using a two-stage feedback decoder as depicted in figure 2.2. The decoder contains a demultiplexer, two interleavers, two de-interleavers and two identical soft decision decoders (DEC1 and DEC2) serially cascaded. The two interleavers are identical to the one used in the encoder of figure 2.1. The de-interleavers perform the inverse operations of the interleavers. The two soft decision decoders (SDD) are symbol-by-symbol maximum a posteriory (MAP) decoders which compute the soft value of each channel output bit using a soft decision decoding algorithm. The overall decoding process is carried out in an iterative fashion during which the current extrinsic value produced by one decoder is shared with the other to improve the next soft decision.

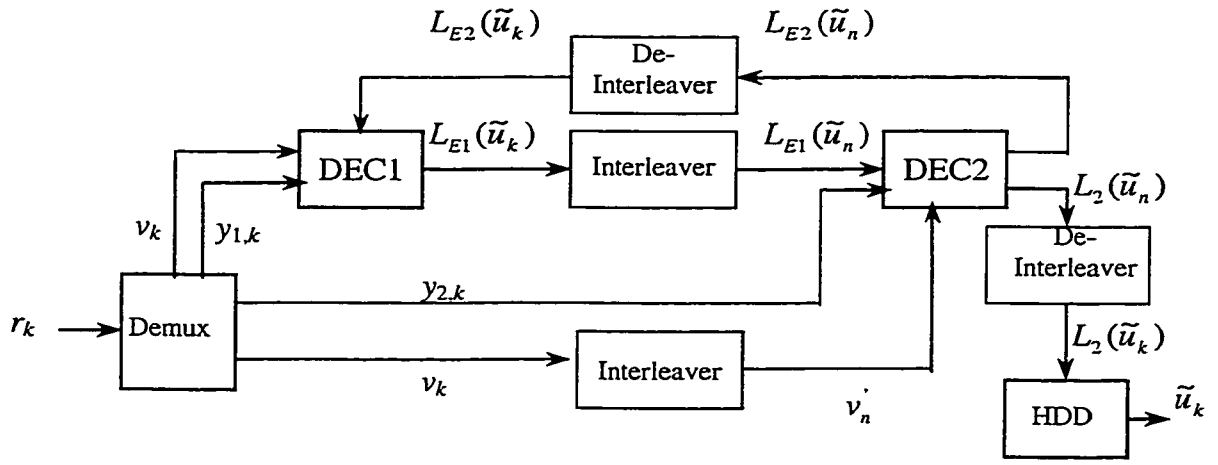


Figure 2.2: A typical rate 1/3 Turbo Codes decoder with feedback.

It is well understood that Viterbi algorithm (VA) is an optimal decoding algorithm for convolutional codes. Unfortunately, this algorithm outputs the "hard decisions", i.e. 0 or 1, as opposed to the a posteriori probability (APP) or the soft value for each decoded bit. As such, it is not suitable for the iterative decoding of Turbo Codes. The appropriate algorithm which produces soft decision for each decoded bit was originally introduced by Bahl, Cocke, Jelineck and Raviv in 1974 [4], and subsequently modified by Berrou et al. [2] for turbo iterative decoding. This modification is necessary to adapt the (BCJR) algorithm to the systematic nature of the Turbo Codes.

The main objective of the VA algorithm is to minimize the probability of sequence error or to select the codeword which minimizes the error probability by finding a ML estimate of the transmitted sequence. The objective of the BCJR algorithm is however to minimize the bit error through the APP estimation of the individual bits. Although it produces an optimal MAP decision for each decoded bit, the BCJR algorithm has not always been the most attractive because of its computational complexity and large storage requirements. An alternative to the BCJR algorithm is Soft-Output Viterbi Algorithm (SOVA) which is sub-optimal but less complex to implement. The mathematical derivations of the SOVA are provided in next chapter.

Going back to the decoder in figure 2.2, the received codeword $\boldsymbol{r} = (\boldsymbol{v}, \boldsymbol{y}_1, \boldsymbol{y}_2)$ is iteratively decoded after multiplied by the channel reliability, L_C. The demultiplexer (Demux) forwards the two pair of $(\boldsymbol{v}, \boldsymbol{y}_1)$ and $(\boldsymbol{v}', \boldsymbol{y}_2)$, where $\boldsymbol{v}'$ denotes the interleaved $\boldsymbol{v}$, to DEC1 and DEC2 respectively. The DEC1 computes the extrinsic value, $L_{E1}(\tilde{u}_k)$, for each decoded bit $\tilde{u}_k$. The extrinsic sequence $\boldsymbol{L}_{E1}(\tilde{\boldsymbol{u}}) = \{L_{E1}(\tilde{u}_1), L_{E1}(\tilde{u}_2), \ldots, L_{E1}(\tilde{u}_k), \ldots, L_{E1}(\tilde{u}_N)\}$ from DEC1 is interleaved and then input along with $(\boldsymbol{v}', \boldsymbol{y}_2)$ to the DEC2 which in turn outputs the soft output sequence, $\boldsymbol{L}_2(\tilde{\boldsymbol{u}}) = \{L_2(\tilde{u}_1), L_2(\tilde{u}_2), \ldots, L_2(\tilde{u}_k), \ldots, L_2(\tilde{u}_N)\}$ and the extrinsic sequence $\boldsymbol{L}_{E2}(\tilde{\boldsymbol{u}}) = \{L_{E2}(\tilde{u}_1), L_{E2}(\tilde{u}_2), \ldots, L_{E2}(\tilde{u}_k), \ldots, L_{E2}(\tilde{u}_N)\}$. The latter is de-interleaved and routed to DEC1 for use in the next iteration. At the beginning, i.e. the first half iteration, $L_{E2}(\tilde{u}_k)$ $(k = 1, 2, \ldots, N)$ is set to zero. At

the end or last iteration, the soft output sequence $\boldsymbol{L}_2(\tilde{\boldsymbol{u}})$ is de-interleaved and forwarded to the hard decision decoder (HDD) to produce a sequence of hard decoded bits $(\tilde{u}_1, \tilde{u}_2, \tilde{u}_3, ..., \tilde{u}_k, ...)$, where $\tilde{u}_k$ = sign $[L_2(\tilde{u}_k)]$. Clearly $L_{E2}(\tilde{u}_k)$ is a function of $L_1(\tilde{u}_n)\big|_{(k \neq n)}$. Since the latter depends on the observations of both $\boldsymbol{v}$ and $\boldsymbol{y}$, $L_{E2}(\tilde{u}_k)$ is correlated with the observations of both v_k and $y_{1,k}$. However, the larger the distance $|n - k|$ produced by the interleaving is, the less the correlation between $L_{E2}(\tilde{u}_k)$ and the observations of v_k and $y_{1,k}$ is [2]. In that sense, $L_{E2}(\tilde{u}_k)$ can be thought of as a diversity effect and can be shared with the DEC1 to achieve a better MAP estimation for each decoded bit. The same logic also holds true for $L_{E1}(\tilde{u}_k)$.

The two decoders (DEC1 and DEC2) of figure 2.2 have three inputs namely the systematic, the parity and the extrinsic. The extrinsic input of DEC1 is provided by DEC2, and vice versa. When performing iterative decoding, it is crucial to ensure that : (i) the a priori information generated by one DEC does not get circulated back to the same DEC at subsequent iteration; and (ii) the extrinsic information output by one decoder remains independent or uncorrelated from the systematic and parity inputs of the other decoder as much as possible. Such independence or non-correlation is mostly accomplished through interleaving action as described in the previous paragraph. In the absence of the interleaving, there is a correlation between the extrinsic information output by one decoder and the systematic and parity inputs of the other decoder. With such correlation in place, the iterative decoding does not work in the sense that the BER remains almost constant regardless of the number of decoding iterations, as will be shown by simulation later.

An obvious, but nevertheless important, observation can be made with respect to the iterative decoding of the Turbo Codes decoder (figure 2.2). At the beginning, one has no knowledge on the statistics of the transmitted data bits u_k's. Assume these data bits are generated by an equiprobable source, i.e. $P(u_k = 0) = P(u_k = 1) = 1/2$ or equivalently $L_E(u_k) = 0$. The first stage decoder (DEC1) decodes the triplet $(L_E(\tilde{\boldsymbol{u}}), \boldsymbol{v}, \boldsymbol{y}_1)$ to obtain $\boldsymbol{L}_1(\tilde{\boldsymbol{u}})$, an estimate of the transmitted data $\boldsymbol{u}$, which can be expressed as

$$L_1(\tilde{u}) = L_C \cdot v + L_{E1}(\tilde{u}) + L_E(u) \quad (2.5)$$

$$L_{E1}(\tilde{u}) = L_1(\tilde{u}) - L_C \cdot v - L_E(u) \quad (2.6)$$

where L_C denotes the channel reliability. The extrinsic information $L_{E1}(\tilde{u})$ is then used along with (v', y_2), where v' denotes the interleaved v, at the second decoding stage to produce

$$L_2(\tilde{u}) = L_C \cdot v' + L_{E2}(\tilde{u}) + L_{E1}(\tilde{u}) \quad (2.7)$$

$$L_{E2}(\tilde{u}) = L_2(\tilde{u}) - L_C \cdot v' - L_{E1}(\tilde{u}) \quad (2.8)$$

The decoding process can be repeated again with this time using $L_E(u)$ = de-interleaved $L_{E2}(\tilde{u})$ rather than $L_E(u) = 0$. Clearly, one can achieve a better $\tilde{u}_k$'s for every additional iteration up to a certain value. The minimum achievable BER depends on the strength of the component codes, i.e. RSC1 and RSC2 generated by ENC1 and ENC2 respectively, and the reliability of the received codeword $r = (v, y_1, y_2)$. For the current Turbo Codes, only the reliability of v is improved during the decoding. On the other hand, the reliability of y_1 and y_2 remain unchanged. Indeed, the same y_1 and y_2 are repeatedly used to improve v. Since both y_1 and y_2 are not error protected, they are likely to contain channel errors, hence adversely limiting the improvement of v at each decoding stage. In the next chapter, it will be shown through simulation that increasing the reliability of y_1 and y_2 can lead to a better BER performance.

2.3 Performance Analysis

The principles of Turbo Codes encoding and decoding were previously discussed. Although the decoding aspects were discussed in details, the encoding counterparts were only discussed in general terms. In the following, a detailed analysis on the Turbo Codes is presented from the encoding point of view. The main objective is to demonstrate that the remarkable performance of the Turbo Code is not only attributable to the strength of the iterative decoding, but also to the powerful code structure and parameters which can be optimized through proper encoding. Furthermore, it is intended to show that Turbo Codes possess a powerful code structure decomposable into several segments. These segments can be manipulated independently of each other, hence make it relatively simple to optimize the dominant design parameters such as the minimum distance and the number of codewords with minimum distance.

2.3.1 Effective Free Distance

Consider a rate 1/3 Turbo Codes whose encoder is depicted in figure 2.1. Recall that encoders ENC1 and ENC2 are two identical rate 1/2 RSC encoders which only output parity bits. For a given input block $\boldsymbol{u} = (u_1 u_2 \ldots u_k \ldots u_N)$, ENC1 generates a parity sequence $\boldsymbol{x}_1 = (x_{1,1} x_{1,2} \ldots x_{1,k} \ldots x_{1,N})$. Similarly, the encoder ENC2 produces another parity sequence $\boldsymbol{x}_2 = (x_{2,1} x_{2,2} \ldots x_{2,k} \ldots x_{2,N})$ from $\boldsymbol{u}'$, which is the interleaved version of $\boldsymbol{u}$. Together, the triplet $(\boldsymbol{u}, \boldsymbol{x}_1, \boldsymbol{x}_2)$ constitutes the three segments of a Turbo Codes codeword. Let w, d^{ENC1} and d^{ENC2} denote the respective Hamming weights or distances of the triplet $(\boldsymbol{u}, \boldsymbol{x}_1, \boldsymbol{x}_2)$. The total weight, d^{TC}, of the corresponding Turbo Code codeword is given by [11]

$$d^{TC} = w + d^{ENC1} + d^{ENC2} \tag{2.9}$$

which can be extended to a rate $1/(q+1)$ Turbo Code as

$$d^{TC} = w + d^{ENC1} + d^{ENC2} + \ldots. + d^{ENCq} \tag{2.10}$$

where d^{ENCq} is the weight of the parity sequence generated by ENC q. Recent works have indicated that when operating at low SNR regions of an AWGN channel, the Turbo Codes performances are largely influenced by the minimum weights of their codewords which are generated by weight-two input sequences [10] [11] [32]. For clarification, all Turbo Codes codewords generated from weight-two input sequences are referred to as d_2^{TC} codewords, while the minimum weight of the d_2^{TC} codewords is referred to as the effective free distance $d_{eff.\,free}^{TC}$ or equivalently $d_{2,min}^{TC}$. The reason why the Turbo Codes performance is determined by the minimum weight of the d_2^{TC} codewords is straightforward for the following obvious reason. The d_1^{TC} codewords has a very large weight due to the infinite impulse response (IIR) nature of the RSC encoders, and thus they can be ignored. On the other hand, the $d_{j,min}^{TC}$ codewords ($j > 2$) tend to have a decreasingly low probability of occurrences because the weight j input sequences which produce $d_{j,min}^{TC}$ codewords are likely to be permuted by the interleaver. Because of the interleaver action, the number of $d_{j,min}^{TC}$ codewords ($j > 2$) is negligibly small [31]. Thus at low and moderate SNR, maximizing the performance of a rate $1/(q+1)$ Turbo Codes ($q = 2, 3, \ldots$) is equivalent to maximizing its effective free distance, $d_{2,min}^{TC}$, which is given by

$$d_{2,min}^{TC} = 2 + d_{2,min}^{ENC1} + d_{2,min}^{ENC2} + \ldots + d_{2,min}^{ENCq} \qquad (2.11)$$

where $d_{2,min}^{ENCq}$ corresponds to the minimum weight of the parity sequences generated by the encoder ENCq from the weight-two input sequences. Eq. (2.11) clearly suggests that another way of optimizing the Turbo Codes performance is to minimize the probability that all elementary encoders ENC1, ENC2, ..., ENC q simultaneously output minimum weight parity sequences for a weight-two input sequence. This can be accomplished through clever design of interleavers.

2.3.2. Evaluation Of Bit Error Probability

Again consider the rate 1/3 Turbo Codes Encoder depicted in figure 2.1, assuming that the encoders ENC1 and ENC2 are two identical rate 1/2 RSC encoders which only output parity bits. For an information block N much larger than the memory m of ENC1, the performance of the rate 1/3 Turbo Codes is almost identical to that of the equivalent $(3N, N\text{-}2m)$ parallel concatenated block codes whose codewords are codewords of the Turbo Codes of length $3N$ which start and end at zero states of both encoders ENC1 and ENC2 [12]. The performance bounds for the Turbo Codes can be computed using the weight enumerating function (WEF) as in the case of the serial concatenated codes (SCC) [10][11][12][33][46]. The input-output weight enumerating function (IOWEF) of the rate 1/2 RSC, denoted by $A^{C_c}(L,W,D)$, is defined as [33]:

$$A^{C_c}(L,W,D) = \sum_{w}\sum_{l}\sum_{d} L^l \cdot W^w \cdot D^d \cdot A^{C_c}(l,w,d) \qquad (2.12)$$

where L, W and D are dummy variables, and $A^{C_c}(l,w,d)$ denotes the number of paths of length l, input weight w and output weight d. These paths start and end at the all-zero states. The conditional probability, $P(d|w)$, of producing an ENC1 codeword of length N and weight d given a random input block of weight w is:

$$P(d|w) = \frac{A^{C_c}(N,w,d)}{\sum_{d} A^{C_c}(N,w,d)}$$

$$= \frac{A^{C_c}(N,w,d)}{\binom{N}{w}} \qquad (2.13)$$

where the denominator $\sum_{d} A^{C_C}(N,w,d)$ is the total number of ENC1 codewords of input weight w. Suppose that d^{ENC1} and d^{ENC2} represent the parity weights produced by the two encoders ENC1 and ENC2 from a given input of weight w, the probability of mapping an input sequence of weight w to a Turbo Codes (TC) codeword of weight d_w^{TC} $= w + d^{ENC1} + d^{ENC2}$, is given by:

$$\begin{aligned} P(w, d^{ENC1}, d^{ENC2}|w) &= P(w|w) \cdot P(d^{ENC1}|w) \cdot P(d^{ENC2}|w) \\ &= P(d^{ENC1}|w) \cdot P(d^{ENC2}|w) \end{aligned} \qquad (2.14)$$

For a binary phase shift keying (BPSK) signaling over AWGN channel, the conditional probability that a maximum-likelihood (ML) decoder will "prefer" a TC codeword having weight $d_w^{TC} = w + d^{ENC1} + d^{ENC2}$ over the all-zeros codeword, or equivalently the pairwise error probability between an all-zeros codeword and the codeword of weight d_w^{TC}, is given by:

$$\begin{aligned} P_S(d_w^{TC}) &= Q(\sqrt{(2d_w^{TC} E_S / N_O)}) \\ &= Q\left(\sqrt{2d_w^{TC} R E_B / N_O}\right) \end{aligned} \qquad (2.15)$$

where R denotes the TC code rate. Based on our previous assumption that $N >> m$ and that the TC performance is almost identical to the equivalent linear block code ($3N$, N-$2m$), the codeword error probability, P_S, is upper bounded by [46]:

$$P_S \leq \sum_{j=2}^{2^N} P_S(j) \qquad (2.16)$$

$$P_S(j) = Q\left(\sqrt{2d_j R E_B / N_O}\right)$$

where $P_S(j)$ is the pairwise error probability between the all-zeros codeword and the codeword j of weight d_j . Note that $d_j \in (1, 2, \ldots, \gamma)$ and there are $A_{d_j}^{TC}$ codewords of weight d_j . Grouping all weight d_j codewords together and replace d_j by d, the codeword error probability equation can be expressed as:

$$P_S \leq \sum_{d=1}^{\gamma} A_d^{TC} \cdot Q\left(\sqrt{2dRE_B / N_O}\right) \tag{2.17}$$

Since $A_d^{TC} = 0$ for $d <$ free distance of the Turbo Codes, eq. (2.17) can be approximated to

$$P_S \approx \sum_{d=d_{free}^{TC}}^{d_{max}} A_d^{TC} \cdot Q\left(\sqrt{2dRE_B / N_O}\right) \tag{2.18}$$

where d_{free}^{TC} is the free distance of the Turbo Codes and d_{max} is the distance d beyond which the erfc $Q(.)$ is negligibly small. Recall that the free distance is the minimum distance separating an all-zero and a non-zero codewords. For low to moderate SNR and large N, the TC bit error probability, P_B, can be approximated by [10] [33]:

$$\begin{aligned} P_B &\approx (\alpha/N^{q-1}) \cdot Q\left(\sqrt{(2d_{2,min}^{TC} RE_b / N_o)}\right) \\ &\approx (\alpha/N^{q-1}) \cdot Q\left(\sqrt{(2RE_b / N_o)\left[\sum_{j=1}^{q} d_{2,min}^{ENCj} + 2\right]}\right) \end{aligned} \tag{2.19}$$

where is α a constant independent of N, the quantity $(1/N^{q-1})$ is commonly referred to as an interleaver gain, and $d_{2,min}^{ENCj}$ is the minimum weight of the parity sequences produced by the encoder ENC j from weight-two input sequences. It is obvious from eq. (2.19) that the TC bit error probability varies as a function of the interleaver length N and the effective free distance $d_{2,min}^{TC}$. In particular, P_B decreases as N and/or $d_{2,min}^{ENCj}$ get larger.

The value of N depends on the tolerable decoding delay for each specific application. As such, it can not be indefinitely large as this can cause indefinite decoding delay. With respect to the effective free distance, an easy way to augment $d_{2,min}^{ENCj}$ is to increase the memory length m of the corresponding ENCj encoder. Unfortunately, this would also increase the decoding complexity substantially. The decoding complexity of convolutional codes grows exponentially as $m \cdot 2^m$, and as such, the practical value of m is often limited to six. Due to this constraint, an optimal design criteria for a rate $1/q$ TC often consists of optimizing $d_{2,min}^{ENCj}$ for a given m rather than optimizing m. Another optimal design criteria is to minimize the occurrences of TC codewords whose weight are $d_{2,min}^{TC}$. These two criteria can be independently optimized as will be shown below.

2.3.3 Constituent Codes Encoder

Constituent codes are referred to as the elementary codes used to construct Turbo Codes. For rate 1/3 Turbo Codes, the two constituent codes are rate 1/2 RSCs. It was discussed earlier that in the low to moderate SNR, the BER performance of Turbo Codes is primarily influenced by its effective free distance, $d_{2,min}^{TC}$, which in turn depends on the $d_{2,min}^{ENC}$ produced by the corresponding convolutional encoders (ENC). The optimization criteria for the constituent codes of a Turbo Codes is therefore not the same as the one used for the stand-alone convolutional codes. The following investigates the characteristics of the convolutional encoders of memory length m which yield good performances for the resulting Turbo Codes.

The constituent encoders (ENC) of a given TC encoder can be either systematic convolutional or non-systematic convolutional (NSC) encoders and do not have to be identical. However, common practice is to use identical systematic convolutional encoders for the following reasons: (*i*) the use of identical constituent encoders generally simplifies the design of the corresponding TC decoder; (*ii*) the use of systematic convolutional encoders rather then the non-systematic ones increases the TC code rate

since only the systematic component from one constituent encoder needs to be transmitted; and (*iii*) it is well known that for a given memory m, the NSC code has generally lower BER than the RSC code at high SNR. At low SNR region, it is the other way around [2]. As the TC is mainly designed to have optimum performance at low SNR region, the systematic convolutional code would be the logical choice.

A simple form of the systematic convolutional encoder is the non-recursive systematic convolutional (non-RSC) encoder. However, the non-RSC encoder is not a good choice since it yields poor effective free distance for the resulted TC. For instance, any weight-one input sequence causes all non-RSC encoders to simultaneously output lowest non-zero weight parity sequences whose concatenation produces a TC codeword with poor distance. Such undesirable occurrence can be prevented if the RSC encoders are used. Because of their infinite impulse response behavior, the RSC encoders normally output large weight parity sequences for a given weight-one input. Consequently, the concatenation of these large weight parity sequences also produces large weight TC codewords. The draw back is that it generally requires some computations in order to properly terminate the trellis of the RSC encoder. A non-RSC encoder with memory m simply requires a tail of m zeros to bring the encoder back to all-zeros state. However, a tail of m zeros does not bring the RSC encoder back to all-zeros state due to the presence of the feedback. To find the tail bits which can bring the RSC back to the all-zeros state, one needs to solve a state-variable equation at the feedback loop. The following examines some parameters of good RSC encoders which, when combined with interleaving, yield high performance for the resulted TC.

Figure 2.3 depicts a typical rate 1/2 NSC encoder with constraint length K and memory $m = K$-1. The encoder accepts an input bit u_k and delivers two output bits $x_{0,k}$ and $x_{1,k}$,

$$x_{0,k} = \sum_{i=0}^{m} u_{k-i} \cdot g_{ai} \tag{2.20}$$

$$x_{1,k} = \sum_{i=0}^{m} u_{k-i} \cdot g_{bi} \tag{2.21}$$

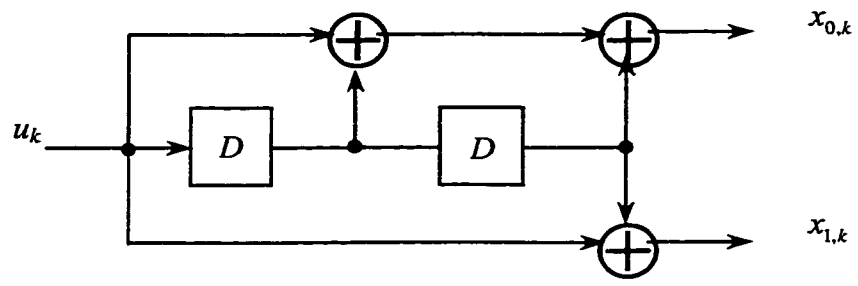


Figure 2.3: A typical rate 1/2 non-recursive non-systematic convolutional encoder ($m = 2$ or $K = 3$)

where g_{ai} and g_{bi} ($0 \leq i \leq m$) denote the coefficients of the two polynomial generators G_a and G_b of the encoder. For the rate 1/2 NSC encoder of figure 2.3, $G_a = [1+ D + D^2]$ and $G_b = [1+ D^2]$. A recursive systematic convolutional RSC encoder can be obtained from the corresponding NSC encoder by using a feedback loop and equating one of the output $x_{0,k}$ (or $x_{1,k}$) to the input u_k as depicted in figure 2.4. The generator matrix for the resulted RSC encoder is given by

$$G = \left[I, \frac{G_b}{G_a} \right] \tag{2.22}$$

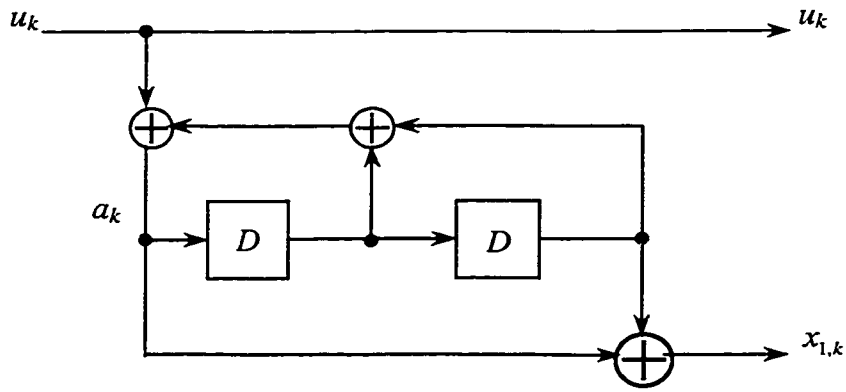


Figure 2.4: A typical rate 1/2 RSC encoder (K=3)

A string of m zeros brings the NSC encoder back to the all-zero state, i.e. terminates the trellis. This is however not the case for the RSC encoder due to the presence of the feedback loop. To terminate the trellis, one needs to solve a state variable equation at the feedback loop; in particular, one needs to set $a_k = 0$ and solve the equation $a_k = u_k \oplus a_{k-1} \oplus a_{k-2}$ for u_k, where $\oplus$ denotes modulo-two addition. This would yield $u_k = a_{k-1} \oplus a_{k-2}$, which is the input required for trellis termination. Another interesting characteristic of the RSC encoder is that its impulse response is infinite but periodic with a period $\tau \leq 2^m - 1$, where m is the degree of the primitive polynomial of the feedback loop, i.e. the degree of G_α, which is 2 for the encoder of figure 2.4.

In general, different RSC encoders tend to produce different average bit error probability performances for the resulted TC encoder. Some design guidelines need therefore to be established and applied in the selection of the RSC encoders with a given memory length m in order to produce a high performance TC encoder. As an example, consider two rate 1/3 TC encoders, TC1 and TC2 encoders, both of which are constructed using identical interleavers of size N, but different RSC encoders. Since both TC1 and TC2 use the same interleavers, let's ignore the interleaver effects for simplicity and focus on the effects of the RSC encoders. Suppose that the encoder TC1 employs two identical RSC encoders, say ENCα. Similarly, the TC2 encoder uses two identical RSC encoders, i.e. ENCβ. The generator matrices for ENCα and ENCβ, denoted by G_α and G_β are given by

$$G_\alpha = \left[1, \frac{n_\alpha(D)}{d_\alpha(D)}\right] = \left[1, \frac{1+D+D^2}{1+D^2}\right] \tag{2.23}$$

$$G_\beta = \left[1, \frac{n_\beta(D)}{d_\beta(D)}\right] = \left[1, \frac{1+D^2}{1+D+D^2}\right] \tag{2.24}$$

All input sequences of same weight w have different patterns. Some input sequences of weight w can produce parity sequences of small weight d, some other input sequences of weight w can produce parity sequences of large weight d' and so on. There are several weight-two ($w = 2$) input sequences which cause the encoder ENCα to produce minimum weight parity sequences. These weight-two input sequences can easily be determined from the associated generator matrix G_α (eq. 2.23). They are in the form of $[1 + D^2]$, which comprises all weight-two sequences in which the two non-zero elements are separated by one bit position, for instance the sequences (1010...0...0), (0...01010...0) and (0...0..0101) are three of them. For a given weight-two input sequence of the form $[1 + D^2]$, the corresponding parity sequence produced by ENCα can be computed from eq. 2.23, and is found to be in the form of $[1+D+D^2]$. With no interleaving, the resulted TC1 codeword is therefore in the form of $[1+D^2, 1+D+D^2, 1+D+D^2]$ with effective free distance, $d_{2,min}^{TC1}$,

$$\begin{aligned} d_{2,min}^{TC1} &= 2 + d_{2,min}^{ENC\alpha} + d_{2,min}^{ENC\alpha} \\ &= 2 + 3 + 3 \\ &= 8 \end{aligned}$$

where $d_{2,min}^{ENC\alpha}$ denotes the minimum weight of the parity sequence produced by ENCα from an input sequence of the form $[1 + D^2]$. Similarly, the sequence of the form $[1+D^3]$, e.g. (0...010010...0), denotes one of the weight-two input sequences which cause ENCβ to output the lowest weight parity sequences. These parity sequences can be computed from eq. 2.24, and are in the form of $[1 + D + D^2 + D^3]$ with weight $d_{2,min}^{ENC\beta} = 4$. The resulted TC2 codeword is in the form of $[1+D^3, 1+D+D^2+D^3, 1+D+D^2+D^3]$ with effective free distance $d_{2,min}^{TC2} = 2 + 4 + 4 = 10$, which is larger than $d_{2,min}^{TC1}$. TC2 therefore outperforms TC1 according to eq. (2.19). It is worth noting that for TC1, the free distance is coincidentally equal to the effective free distance which is 8. For TC2, the effective free distance is 10, but the free distance which is equal to the weight of the output sequence of the form $[1+D+D^2, 1+D^2, 1+D^2]$ is 7.

Obviously without interleaving, TC1 would have superior performance to TC2 because it has larger free distance. However by using a large size interleaver, it would be possible to minimize the occurrence of the undesired weight-three input sequences [$1+D+D^2$], and thus making their contributions to BER probability insignificant compared to the weight-two input sequences [$1 + D^3$]. The performance of the TC2 encoder is determined by the weight-two rather than the weight-three input sequences as a result of interleaving. The TC2 encoder has thus better performance than the TC1 encoder since $d_{2,min}^{TC2} > d_{2,min}^{TC1}$.

From the previous rate 1/3 Turbo Codes TC1 and TC2 examples, it is clear that $d_{2,min}^{ENC}$ rather than the free distance of the recursive convolutional codes generated by ENC should be the optimal parameter for designing Turbo Codes. For a given memory length m, the larger $d_{2,min}^{ENC}$, the better the performance of the resulting Turbo Codes. Another important design parameter to be optimized is the number of the convolutional codewords, $N_{2,min}^{ENC}$, associated with the distance $d_{2,min}^{ENC}$. This is simply because the number $N_{2,min}^{TC}$ of TC codewords is related to $N_{2,min}^{ENC}$, i.e. $N_{2,min}^{TC}$ decreases with $N_{2,min}^{ENC}$. Obviously, the smaller $N_{2,min}^{TC}$, the less frequent the occurrence of TC codewords with $d_{w,\min}^{TC}$ and thus the lower the TC BER probability. Benedetto and Montorsi [11] have shown that for a given rate $1/q$ RSC encoder (ENC) with a memory m and a generator matrix

$$G = \left[1, \frac{n_1(D)}{d_1(D)}, \frac{n_2(D)}{d_2(D)}, \ldots\ldots, \frac{n_{q-1}(D)}{d_{q-1}(D)}\right] \qquad (2.25)$$

it is possible to achieve $d_{2,min}^{ENC} = (q-1)(2^{m-1} + 2)$. For rate 1/2 RSC encoder (ENC) with a memory m:

$$G = \left[1, \frac{n(D)}{d(D)}\right] \qquad (2.26)$$

The general guidelines for selecting the rate 1/2 RSC codes for the TC codes are [11]:

i) $d(D)$ is a primitive polynomial of degree m.

ii) For all $n(D)$'s which yield $d_{2,min}^{ENC} = (2^{m-1} + 2)$, select the one which produces good combinations of $(d_{w,\min}^{ENC}, N_{w,\min}^{ENC})$, i.e. few codewords with small weights and many codewords with large weights, where $2 \leq w \approx 5$ for self-terminating input sequences.

2.3.4 Self-Terminating Sequences

In the previous analysis of a rate 1/3 TC, it was shown that the TC performance is predominantly determined by the effective free distance, i.e. $d_{2,min}^{TC}$. In the absence of interleaving, it was observed that the two constituent codes encoders (ENC) output parity sequences of weight $d_{2,min}^{ENC}$ for an undesired weight-two input and that the corresponding $d_{2,min}^{TC}$ is always equal to 2 $(1 + d_{2,min}^{ENC})$. In the following, the full effect of the interleaving on the TC performance is explored. In particular, it is demonstrated that interleaving the input sequences prior to encoding by one of the ENC encoders minimizes the probability that both ENC encoders simultaneously output parity sequences of weight $d_{2,min}^{ENC}$. This is equivalent to minimizing the number of TC codewords with $d_{2,min}^{TC}$.

By definition, a self-terminating sequence is referred to as a sequence which can drive the encoder to the all-zero state without adding tail bits. For instance, the input sequence (00...01010...0) is considered as a non-self-terminating since it does not drive our rate 1/2 RSC encoder to all-zero state without adding tail bits. Consider a rate 1/2 RSC encoder depicted in figure 2.4. Some of its input sequences are self-terminating while others are non-self-terminating. The corresponding parity sequence produced from this input sequence is (00...00110101111...) with the accumulation of 1's until terminated by tail bits.

Non-self-terminating sequences have negligible effect on the TC bit error performance because they produce TC codewords which accumulate large weights until artificially terminated by tail bits. Self-terminating sequences, however, adversely affect the TC performance as some of them tend to produce TC codewords with relatively low weights. Among all self-terminating sequences, weight-two self-terminating sequences constitute the most important TC design consideration. Higher weight self-terminating sequences are of successively decreasing importance as there is a large probability that the action of interleaving would successfully transform them into non-self-terminating sequences. It should be noted that self-terminating sequences are generally unique for each encoder.

For the previous ENCβ whose generator matrix G is defined in eq. (2.24), the sequence (00...010010...00) is a self-terminating sequence. The corresponding parity sequence produced by such an encoder from this particular sequence is a weight-four sequence (0...011110...0), and the weight of the resulted TC2 codeword would be $2 + 4 + 4 = 10$ if there is no interleaving. The sequence (00...010010...00) is not the only weight-two self-terminating sequence. It is evident that any other weight-two sequence with its 1's separated by a distance $\lambda = 3\delta$ bit positions where δ =1,2,3,... is also a self-terminating sequence [21]. In general, the weight of the output corresponding to such input is linearly increased with δ. For example, a self-terminating input sequence (00...010000010...00) [$\delta = 2$ and $\lambda = 6$] would cause the ENCβ encoder to output a weight-six parity sequence (00...00111011100...00). The total weight of the resulted TC2 codeword would thus be $2 + 6 + 6 = 14$, provided that no interleaving is used. Likewise, the sequence (0...010000000010...0) [$\delta = 3$, $\lambda = 9$] would cause the same encoder to output a parity sequence (00...00111011011100...00), hence a total TC2 weight of $2 + 8 + 8 = 18$, if there is no interleaving. For any weight-two sequence in which the two 1's are separated by a distance $\lambda = 3\delta$ bit positions, the total weight of the related rate 1/3 TC2 codeword with no interleaving is given by

$$d_2^{TC2} = d_{2,min}^{TC2} + 2\sum_{i=1}^{2}(\delta^i - 1)$$

$$= 10 + 4\,(\delta - 1) \qquad (2.27)$$

where $\delta = 1,2,3,4,\ldots$. With interleaving, two things can happen: (*i*) the self-terminating sequences may become non-self-terminating sequences, in which case their contributions to the TC2 bit error probability are negligible; (*ii*) the self-terminating sequences may remain to be self-terminating, but the distances separating the two 1's are most likely to change after interleaving, and the overall weight d_2^{TC2} would be equal to

$$d_2^{TC2} = d_{2,min}^{TC2} + 2 \cdot [(\delta - 1) + (\delta' - 1)]$$

$$= 10 + 2 \cdot [(\delta - 1) + (\delta' - 1)] \qquad (2.28)$$

where δ' is the distance separating the 1's after interleaving. Without interleaving, $\delta = \delta'$, and d_2^{TC2} is reduced to our previous $d_{2,min}^{TC2}$ when $\delta = 1$. In any case, interleaving certainly guarantees a better TC2 performance for a given choices of ENCβ encoders since there is a large probability that the resulted TC2 codeword will have a weight greater than $d_{2,min}^{TC2}$ for a given self-terminating weight-two input sequence whose 1's are separated by three bits position.

Obviously, eq. (2.28) is unique for each choice of ENCβ. For a ENCβ encoder of different generator matrix G_β would yield different expression for d_2^{TC2}. The coefficient '2' in eq. (2.28) is a measure of how fast the parity weight of each ENCβ encoder grows as a function of the distance separating the two 1's. This coefficient can not be made larger than 2 for a ENCβ encoder with memory $m = 2$ [32]. However, it can be made smaller. For instance, by permuting the numerator and denominator of the generator matrix G_β in eq. (2.24), one obtains a different total weight equation as shown below

$$d_2^{TC2} = 8 + [(\delta - 1) + (\delta' - 1)] \qquad (2.29)$$

which grows only half as fast as eq. (2.28). Both eq. (2.28) and (2.29) can be used to fine tune the earlier statement on the selection of the constituent codes for an optimal TC. Indeed, these two equations suggest that for a given memory length m, one should choose the constituent codes which maximize not only $d_{2,min}^{TC}$ but also the growth rate of d_2^{TC}. The latter is a function of two elements - the coefficient and the summation of the distances separating the two 1's, i.e. the sum of δ's values. This summation can be maximized through the proper choice of interleaver.

2.3.5 TC Weight Distributions

The role and the effect of interleaving within a TC can be conveniently explained through the observation of the weight distribution of the corresponding TC codewords. Again, consider the rate 1/3 TC depicted in figure 2.1. For a given input sequence, the two identical ENC encoders (ENC1 and ENC2) generate different parity sequences as a result of interleaving prior to the second encoding. The objective of the TC encoding is to concatenate or to pair a low-weight parity sequence from ENC1 with a high-weight parity sequence from ENC2 such that the total weight of the result TC codeword is as much higher than $d_{2,min}^{TC}$ as possible.

The weight distribution of the TC codewords thus depends on how parity sequences from ENC1 are concatenated with those from ENC2. If the two encoders are identical but non-RSC encoders, both of them will produce minimum weight codewords for a weight-one input sequence (00..010..00) whether or not an interleaver is used. However, this is not the case when ENC1 and ENC2 are RSC encoders. With their infinite impulse response characteristic, both encoders produce parity sequences whose weights are only kept finite due to the finite length of the weight-one input sequences. The input sequences which cause the two encoders to generate parity sequences with minimum weight are no longer weight-one, but the weight-three sequences with three consecutive 1's in the form of $[1+D+D^2]$ as shown in the previous section. However, such sequences can easily be permuted or altered by an interleaver, eliminating the possibility that the two encoders

output minimum weight sequences simultaneously. As a result, $d_{3,min}^{TC}$ would be substantially larger than $d_{2,min}^{TC}$ and the contributions of the weight-three input sequences to the TC bit error probability are negligibly small.

The presence of a feedback loop in the encoder does not alter the weight distribution of the parity sequences produced by that encoder. It simply changes the input-output mapping relationship. Instead of mapping the weight-one input sequences to the minimum weight parity sequences, it maps the weight-three input sequences, hence providing interleavers a "chance" to break up the undesirable pattern of the weight three input sequences responsible for the occurrence of minimum weight parity sequences. The feedback loop of the encoder is said to produce a favorable input-output mapping relationship which makes it possible for the interleaver to improve the weight distribution of the TC codewords, i.e. more high weight codewords and less low weight codewords.

2.3.6 Interleavers

TC codewords are made up of several segments including the input sequence and the parity sequences from different ENC encoders. As such, the TC weight distribution is affected by the way parity sequences from ENC encoders are concatenated. Ideally, it is desirable to avoid or at least to minimize the probability of concatenating the low weight parity sequences from one ENC encoder with the low weight parity sequences from other ENC encoders. Interleavers generally do a very good job in matching the low weight with high weight parity sequences for most of the time.

From the probabilistic point of view, the interleaver design goal is to effectively combat or permute the undesired weight-two input sequences. It was shown that weight-two self-terminating input sequences are the ones having the most influence on the bit error rate, and that d_2^{TC} linearly increases with the bit positions separating the two 1's within these input sequences. Good interleavers should be able of spreading the distance between the two 1's within a weight-two self-terminating sequence as far as possible. They should

furthermore preserve the distance between the two 1's in a self-terminating sequence with large distance between ones. In general, such distance, and hence the TC performance, increases with the interleave sizes up to a certain value, and large interleavers would yield better performance for the resulted TC. Unfortunately, large interleaver sizes also imply long systematic decoding delays which may not be suitable for many real time applications, hence a trade-off between the BER performance and decoding delay needs to be made.

It is clear that interleaving has no effect on weight-one input sequences in the sense that whether or not these sequences are interleaved, all ENC encoders will produce the same codeword. But since ENC encoders are RSC encoders, they will produce large weight codewords from the weight one input sequences, and consequently the resulted TC codewords will also have large weight. Because of the IIR characteristic of the RSC encoders, one does not need to be concerned about the fact that the interleaving operation has no effect on weight-one input sequences. Note that this would not be the case if the ENC encoders are non-recursive.

It should be pointed out that in addition to minimizing the probability of concatenating the minimum weight parity sequences from the constituent encoders (ENCs) together, interleavers play a more important role in the iterative decoding of Turbo Codes. In fact, it is easy to minimize the probability of concatenating the minimum weight parity sequences together through clever selection of ENCs, i.e. using different ENCs. For instance, by using the generator matrices defined in eq. 2.23 and 2.24 for ENC1 and ENC2 respectively, one can minimize the probability of matching together the low weight parity sequences generated by the two ENCs without using an interleaver. This would solve the problem of low weight parity sequences in the encoding, but introduce another problem in the decoding.

Interleavers can be grouped into random and non-random interleavers. It was suggested that random interleavers yield better performance than the non-random ones. However, the latter have advantage of terminating the trellis of both constituent encoders (ENCs),

while using random interleaver, one often terminates only the trellis of one encoder and leaves the other open or unterminated [21][39]. Designs of some non-random interleavers based on block interleaving and circular shifting are considered here. It is shown that for a given choice of identical ENC encoders, increasing the interleaver length results in noticeable performance improvement over a wide range of the bit error probability. In particular, using a non-random interleaver, the TC effective free distance grows roughly as $\sqrt{2N}$, where N is the interleaver length or size.

2.3.6.1 Non-Random Block Interleaver

A non-random block interleaver of size N can be thought of as an $(n \times m)$ matrix, where $n \times m = N$. The input sequence is written into the matrix in a row-by-row fashion from the top to the bottom, and read out in column-by-column fashion from the right to the left. Two distinct applications can be associated with non-random block interleaver: intra-block interleaving and inter-block interleaving. The former consists of interleaving input bits within a single data block whereas the latter involves permuting data bits within a multiple of data blocks. In general, a non-random block interleaver which is designed to be optimal for inter-block interleaving may not be optimal for intra-block interleaving.

For inter-block interleaving, a non-random block interleaver is the most effective in the situation where all 1's of the low weight input sequence are confined to one row, but the least effective to permute other patterns of low weight input sequences. Figure 2.5 shows a pattern which can not be permuted by the block interleaver. A solution to address such problem is to write to and read from the matrix in a specific order rather than the normal top-to-bottom and right-to-left order. The modification allows the block interleaver to permute this particular pattern, although it may fail to permute others.

For intra-block interleaving, the pattern in figure 2.5 corresponds to a weight-four input sequence and it is no longer considered as a critical pattern. Recall the TC performance is determined by weight-two rather than weight-four inputs. As such, the non-random block interleaver should be designed to permute or break pattern of weight-two inputs where the two 1's are confined to one or two consecutive columns as shown in figure 2.6. The design of non-random block interleaver therefore consists of identifying the most critical pattern and to specify the writing and reading orders in such a way that the permutation of that specific pattern can be maximized.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 2.5: An example of inter-block interleaving where a block interleaver is not effective.

(a)
$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(b)
$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 2.6: Two examples of intra-block interleaving where a block interleaver is not effective.

2.3.6.2 Non-Random Circular Shift Interleaver

A non-random circular shift interleaver can be viewed as a mapping device which maps an element $j \in [1, N]$ into another element $\mu \in [1, N]$. A bit position j at the input of the interleaver is placed at position $\mu(j)$ at the output of the interleaver according to the following mapping relationship [32]:

$$\mu(j) = (\alpha j + \beta) \bmod N \tag{2.30}$$

where $\beta < N$ is an offset and $\alpha < N$ denotes a step size which is relative prime to N. In practice, β is often set to zero. As an example, setting $N = 32$ and $\alpha = 7$ will yield the following permutation:

j	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
μ (j) =	0	7	14	21	28	3	10	17	24	31	6	13	20	27	2	9
	16	23	30	5	12	19	26	1	8	15	22	29	4	11	18	25

With these values of α and N, it is observed that while any two consecutive elements of j's is separated by a unit distance, the two consecutive elements of $\mu(j)$ is separated by a distance of either 7 or 25 units. This implies that if the two 1's of a weight-two sequence is separated by $d_1 = 1$ bit position, the same two 1's will be separated by $d_1^I = 7$ or 25 bit positions in the corresponding permuted sequence. Similarly, if $d_1 = 2$, then $d_1^I = 14$ or 18 (50 mod 32) after the sequence has been permuted. Continuing this exercise, it can be shown that $(d_1 + d_1^I) \geq 8$. The above example can be generalized as following [32]: if the block size N is half of a perfect square and the step size is selected to be $\alpha = \sqrt{2N} - 1$, then $(d_1 + d_1^I) \geq \sqrt{2N}$ for all possible combinations of d_1 and d_1^I.

With a non-random circular shifting interleaver, it was demonstrated that it is possible to increase the minimum $(d_1 + d_1^I)$ without bound for an unbounded N. However, this does not imply that one has found an ideal interleaver for a TC encoder. In fact, what has been shown is an interleaver which can effectively combat the undesired weight-two sequences. This may not be true for undesired sequences of higher weights. For instance, this interleaver maps a weight-four sequence (10010100100000000000000000000000) to (10010000000000000000010010000000). As a result, the weight of the corresponding TC codeword remains unchanged, whether or not the TC encoder uses the interleaver. In this case, the interleaver is not effective in the sense that its action does not increase the weight of the resulted codeword. The design of circular shifting interleaver to combat the undesired weight-four and higher-weight sequences while preserving its optimality with respect to the weight-two sequences remains so far an unsolved challenge.

The above analysis on the circular shifting interleaver ignores the edge effects, i.e. it does not take into account the trellis termination. In order to properly terminate the trellises of both ENC encoders using m tail bits where m is the memory cells of the ENC encoders, the mapping function of the interleaver should be in the form of [46]

$$j \bmod p = \mu(j) \bmod p \qquad (2.31)$$

where $p = 2^m$-1 is the period of the ENC encoder. As example, setting $N = 34$ (including 2 tail bits) and $m = 2$ yields the following

j:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34
$\mu(j)$:	16	29	9	10	14	6	31	8	12	22	17	33	34	23	24	19	32
	30	13	2	21	25	26	3	28	20	27	7	5	15	4	11	18	1

This interleaver can not combat all undesirable weight-two sequences. For instance, an undesirable weight-two sequence (00100100000000...00) will be permuted into another undesirable weight-two sequence (0000010010000...00). It is clear that there is a major difference between these two modulo N and the modulo p interleavers. The first one permutes a self-terminating weight-two sequence into another self-terminating weight-two sequence while at the same time attempting to optimize the distance separating the two 1's in the permuted sequence. The modulo p interleaver, however, tries to optimize the mapping of the self-terminating weight-two sequences into non-self-terminating weight-two sequences.

2.3.6.3 Random Interleaver

A random interleaver can be viewed as a device which randomly permutes the bit positions within a sequence. Random interleavers do a very good job in matching the low weight with high weight sequences for most of the time. However, a purely random interleaver is also likely to produce a few undesired weight-two sequences, i.e. there are several occasions when a purely random interleaver fails to break the undesired weight-two sequences. It was shown previously that the undesired weight-two sequences can be effectively combat using a non-random interleaver at the expense of amplifying the bad

effects of the sequences with weight-four and higher. In principle, it should thus be possible to design a non-random interleaver to outperform the random one. However, this is not the case in practice. To date, most of the Turbo Codes constructed with random interleavers have shown to achieve lower decoded bit error rate than those constructed using non-random interleavers [32]. A possible explanation as to why the former tend to yield better BER performance than the latter is as following. The non-random interleavers can be designed to be more effective than the random ones in terms of combating specific undesired self-terminating sequences. However, due to its nature of randomness, it is believed that the random interleavers are more effective than the non-random ones in term of breaking the correlation between the extrinsic information output by one decoder (DEC) and the observed bits input to the other decoder.

2.4 Simulation Results

This section presents simulations, plots and discusses the BER performances of Turbo Codes (TC) whose component or constituent codes are rate 1/2 RSC codes. These simulations are conducted to illustrate the typical behavior of the BER performances in the region of low SNR, i.e. (0.25 – 3.00 dB), versus several important parameters such as the number of decoding iterations, the interleaver length, the generator polynomial of the component codes and the number of the constituent codes or the TC code rate. All simulations are performed using Soft Output Viterbi Algorithm (SOVA) with Monte Carlo simulations. Given a large number of simulations to be performed, the SOVA algorithm rather than the BCJR algorithm is used to reduce the simulation times. Recall that the SOVA algorithm is roughly twice faster than the BCJR algorithm.

Figure 2.7 shows the BER performance of a rate 1/3 TC constructed with a single random interleaver of length N = 900 bits, and two identical rate 1/2 RSC constituent codes whose generator polynomials are given by $G = [1, \ 1+D^2/1+D+D^2]$. It should be pointed out that these two parameters (N = 900 and $G = [1, \ 1+D^2/1+D+D^2]$) are specifically chosen in order to compare the simulation results with the ones presented in [22] using the SOVA algorithm. It is observed that at the sixth iteration, the BER curve of figure 2.7 closely agrees with the one obtained in [22], shown in broken line. It should be pointed out that only the BER at sixth iteration is plotted in [22].

In general, the BER decreases as the number of iterations increases. The BER improvement over the number of iterations appears to be minimal at very low SNR, but becomes quite significant as the SNR gets larger. The reason should be obvious. At very low SNR, received codewords are likely to contain more errors than can be corrected by the constituent codes, resulting in large probability of decoding errors regardless of the number of iterations. Also, it is observed that most of the BER reduction is achieved within the first four to six iterations while there is only slight improvement for additional iterations once the BER has reached some value or threshold (refer to figure 2.8). This clearly suggests that there is an error floor associated with each SNR. Such error floor is strongly affected by the noise immunity of the parity segments of the corresponding TC. In the next chapter, it will be shown that it is possible to lower such error floor by raising the noise immunity of the parity segments.

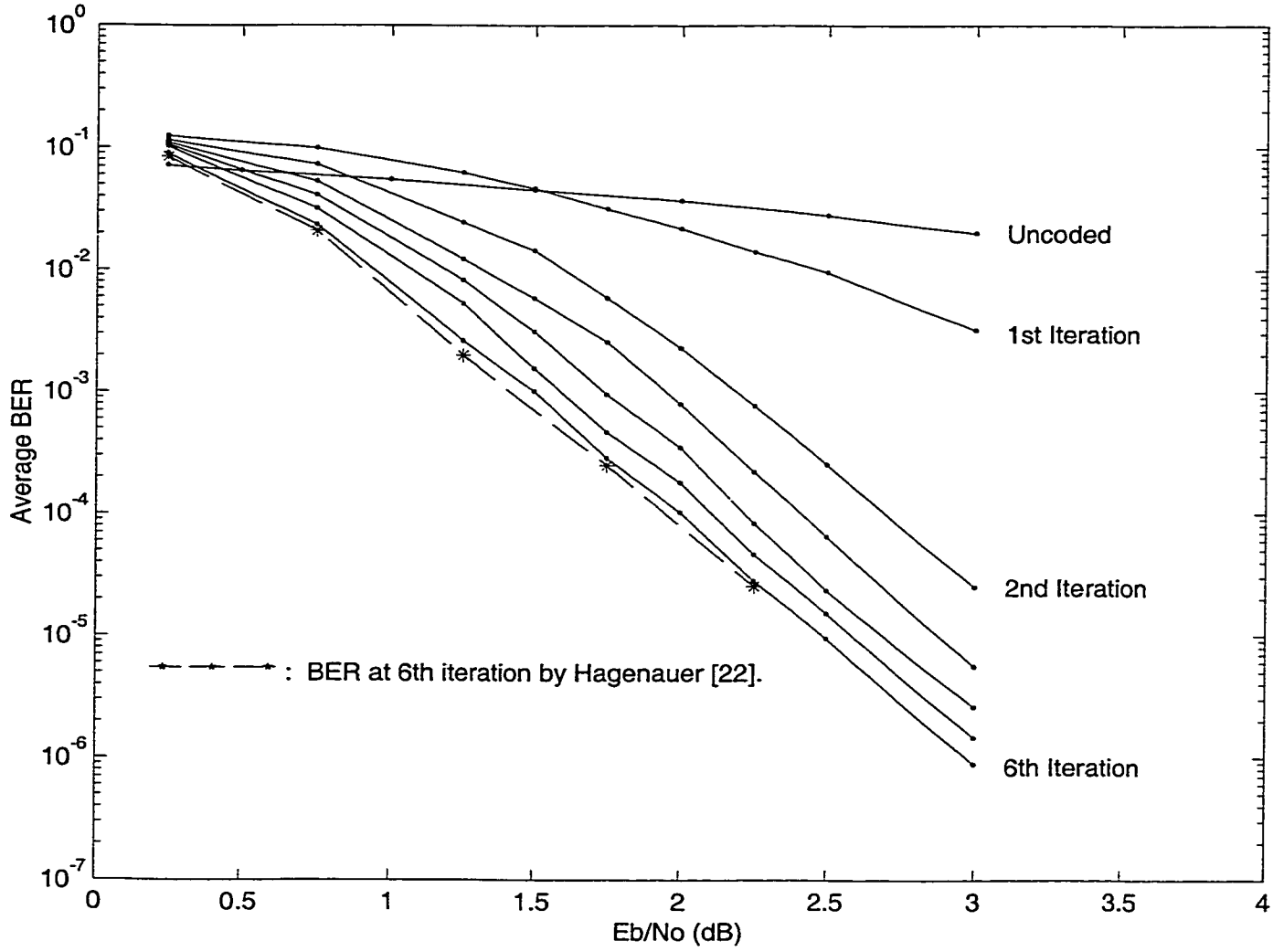


Figure 2.7: BER performance of a rate 1/3 TC with random interleaver ($N = 900$, $G = [1, 1 + D^2 / 1 + D + D^2]$).

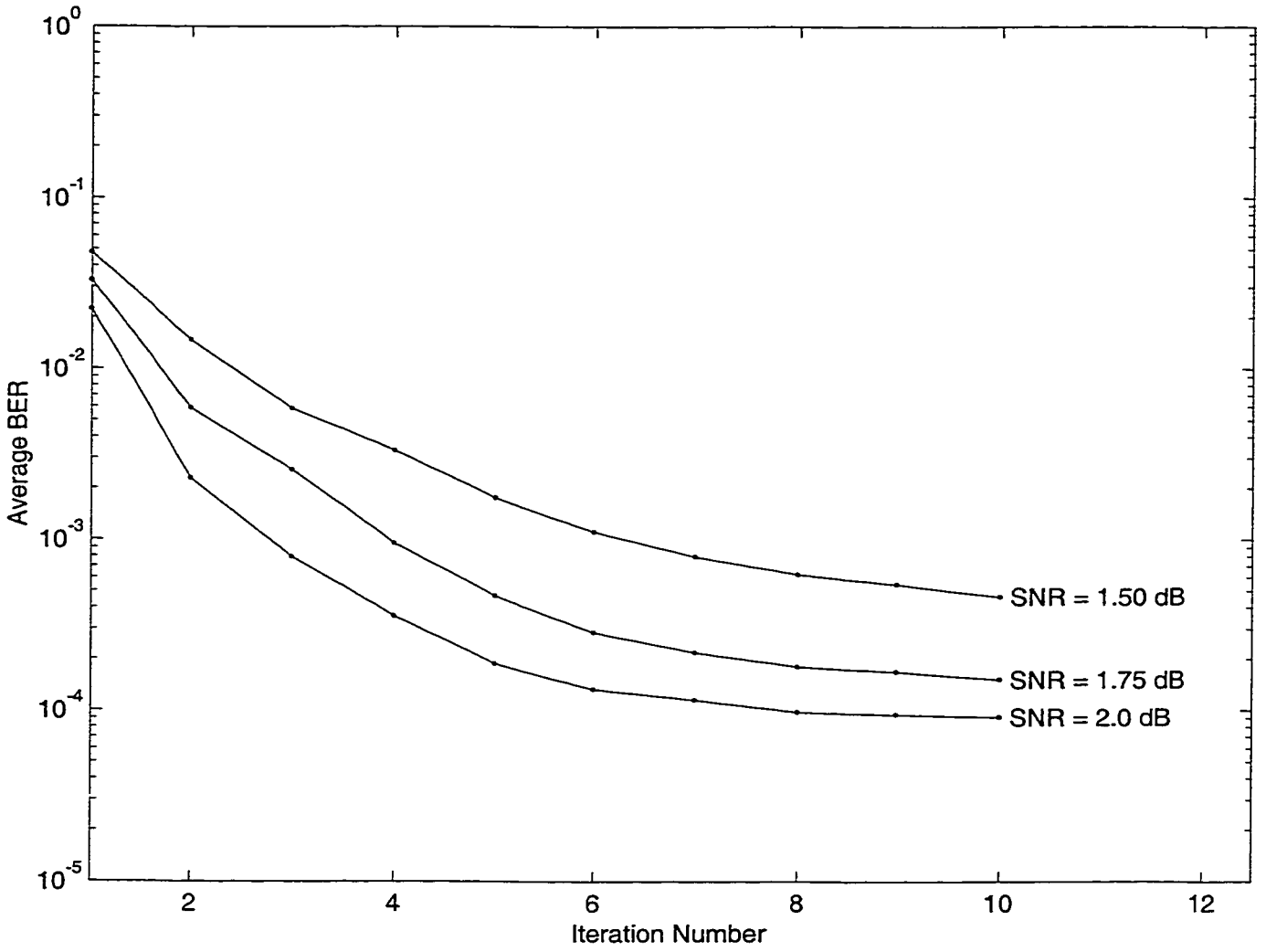


Figure 2.8: BER performance vs decoding iterations for rate 1/3 TC with random interleaver ($N = 900$, $G = [1, 1+D^2/1+D+D^2]$).

Figure 2.9 and 2.10 depict the BER performances of an unpunctured rate 1/3 TC with $G = [1,\ 1 + D^2 / 1 + D^2 + D^2]$ as before, but using a random interleaver of variable length this time. Specifically, these simulations are intended to demonstrate the crucial role of interleaving in iterative decoding, and to show the impact of varying the interleaver length on the BER performance. It was stated earlier that interleavers are essential for iterative TC decoder to work properly. Without interleavers, there is a correlation between the extrinsic information provided by one elementary decoder (DEC) and the observed constituent codewords input to the other elementary decoder (DEC), hence making iterative decoding ineffective. By using interleaver, such correlation can be avoided and it has been shown that the Turbo Codes BER can be reduced by increasing the interleaver length. To put these results in perspective, the performances of unpunctured rate 1/3 TC were simulated using random interleavers of different lengths, i.e. $N0 = 0$, $N1 = 100$, $N2 = 500$, $N3 = 1000$, $N4 = 2500$, $N5 = 5000$ and $N6 = 10000$ bits. Here, $N0 = 0$ indicates no interleaving. This is equivalent to removing the interleaver from the TC encoder.

Figure 2.9 plots the BER performances for the seven values of N (i.e. $N0,\ldots,N6$) at SNR = 1.50 dB over six decoding iterations, whereas figure 2.10 plots the BER performance at the fifth iteration over the SNR in the range of (0.25 – 2.50 dB). As predicted, these two figures confirm the general reduction of BER as a function of increasing interleaver length. Of particular interest is figure 2.9 which clearly shows that without interleaving ($N0 = 0$), the BER remains almost the same despite a number of decoding iterations carried out. One may suspect that the absence of the BER improvement over the number of decoding iteration could be attributed to the increase in the probability or the frequency of concatenating low weight parity sequences from the two constituent encoders (ENCs) together. To show that this is not the case, the BER performance of rate 1/3 TC with no interleaving and two different constituent encoders, i.e. ENC1 $\neq$ ENC2 to ensure that they do not output low-weight parity sequences simultaneously, is simulated. The result is almost the same as in the case where ENC1 and ENC2 are identical.

At SNR = 1.50 dB, there is basically no change to the BER at the first iteration as a result of increasing the interleaver length from $N1$ to $N6$. However, BER improvement is increasingly achieved at subsequent iterations, starting with a small improvement at the second iteration. Also, for the interleaver of length $N1$, very slight reduction of BER is achieved as a function of decoding iterations. For instance, the BER at first iteration is roughly 5.00×10^{-2} and 1.50×10^{-2} at sixth iteration. As the interleaver length increases, the reduction of BER as a function of decoding iterations becomes increasingly large. For example, the interleaver of length $N5$ yields BER of 4.40×10^{-2} and 4.3×10^{-5} at first and sixth iterations respectively. Finally by increasing the interleaver from $N1$ to $N6$, an overall BER reduction in the order of (10^3) is obtained at sixth iteration. This is one order higher than the approximated theoretical value. It should be pointed out that the theoretical expression for the BER is very loose and is expressed only in terms of the minimum distance, the interleaver length and the SNR, but not the number of decoding iterations.

The BER performances for the six values of interleaver lengths ($N1$, $N2$, ..., $N6$) at the fifth decoding iteration and in the low region of SNR are plotted in figure 2.10. Again, it is observed that changing interleaver lengths have little or no impact on the BER performance at very low SNR (0.25dB). As the SNR increases, increasing the interleaver length yields significant BER reduction. Although this makes it possible to reduce BER by making the interleaver indefinitely large, it is not realizable in practice due to the long decoding delay and large memory storage. For instance, $N5$ can have a maximum decoding delay approximately equal to (50 $\times$ time required to decode $N1$).

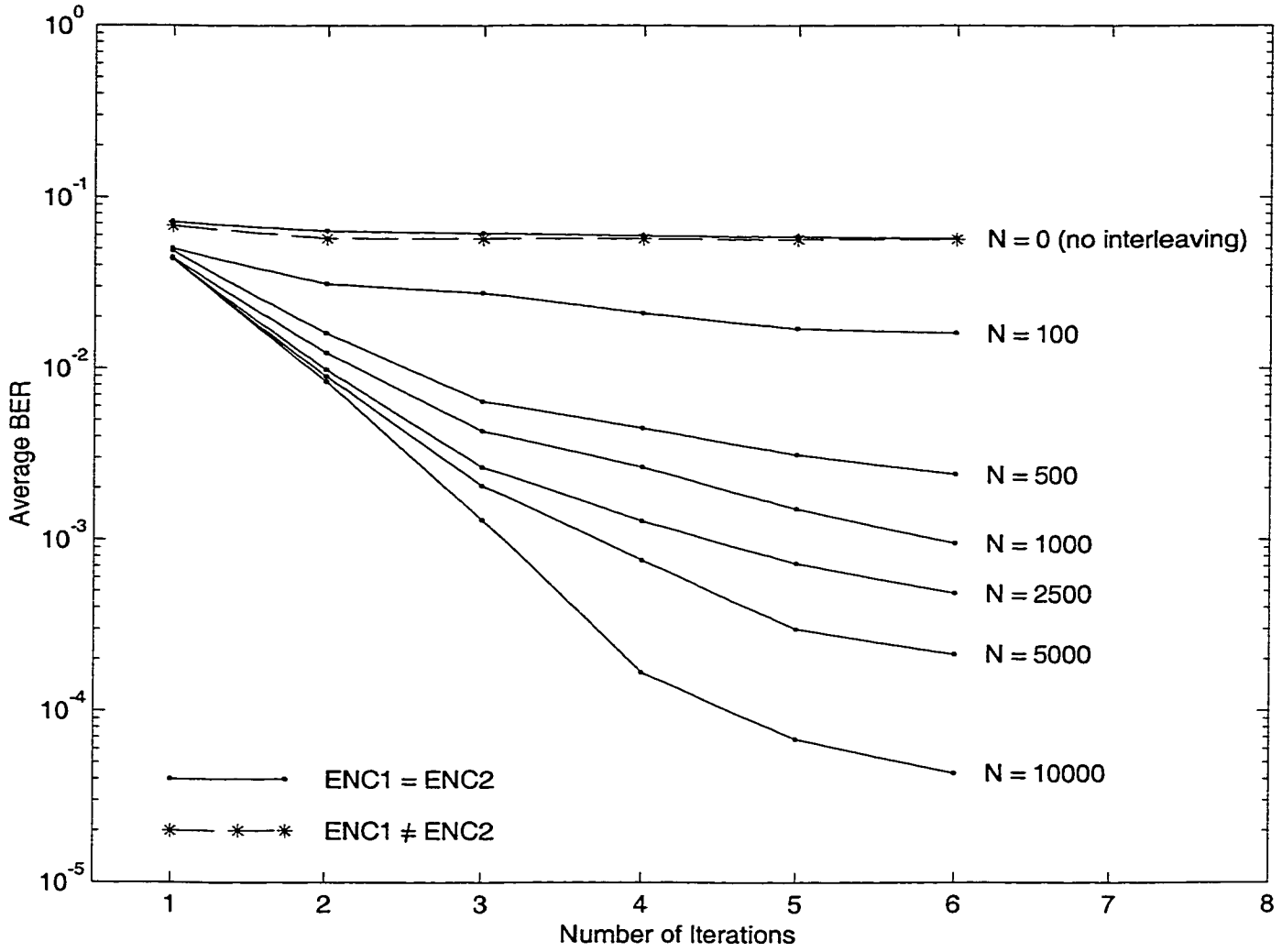


Figure 2.9: Rate 1/3 TC BER performance vs decoding iterations for different interleaver lengths ($G = [1, 1 + D^2 / 1 + D + D^2]$; SNR = 1.5 dB and $N = 0$, 100, 500, 1000, 2500, 5000 and 10000).

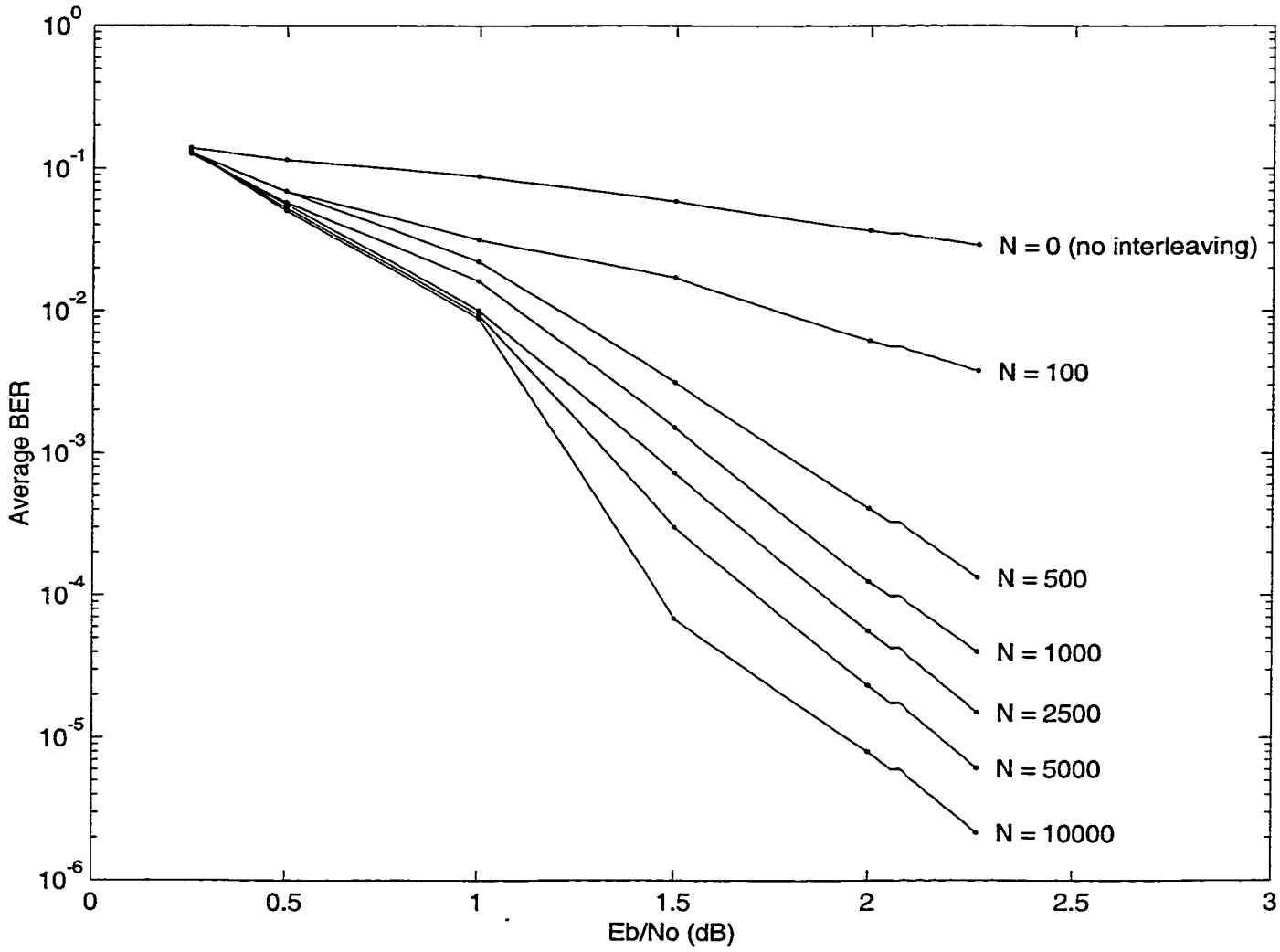


Figure 2.10: Rate 1/3 TC BER Performance vs SNR for different interleaver length at 5th decoding iteration ($G = [1, 1+D^2/1+D+D^2]$; $N = 100, 500, 1000, 2500, 5000$ and 10000).

Figure 2.11 and 2.12 depict the simulated BER performances of three unpunctured rate 1/3 Turbo Codes, TC1, TC2 and TC3, which are constructed with identical random interleavers of length N = 900 bits, but distinct constituent codes. Specifically, the generator polynomials for the constituent codes of the three Turbo Codes are $G1 = [1, 1 + D^2/1 + D + D^2]$, $G2 = [1, 1 + D + D^3/1 + D + D^2 + D^3]$ and $G3 = [1, 1 + D^4/ 1 + D + D^2 + D^3 + D^4]$ respectively. The constraint lengths associated with the three generator polynomials are $K_1 = 3$, $K_2 = 4$ and $K_3 = 5$.

The BER performances for the three Turbo Codes after the fifth decoding iteration are depicted in figure 2.11. It is observed that for SNR < 2.00 dB, increasing the constraint length of the constituent codes does not yield significant performance improvement for the resulting Turbo Codes. At SNR = 2.0dB, raising the constraint length from 3 to 5 produces a gain of roughly 0.25 dB. Higher gain is expected at larger SNR. The BER performances of the three TCs over six decoding iterations at SNR = 2.00 dB is shown in figure 2.12. Again, it is clear that increasing the constraint length of the constituent codes can lead to performance improvement. It should be noted however that such improvement is achieved at a cost of increasing the decoding complexity by a factor F, which is approximately given by

$$\mathrm{F} \sim (R^{-1} - 1)[2^{K_2 - 1}(K_2 - 1) - 2^{K_1 - 1}(K_1 - 1)]$$

where R denotes the unpunctured TC code rate. The above work factor F is computed using the assumption that the decoding complexity of convolutional codes grows exponentially as $(K-1)\cdot 2^{K-1}$. As a specific example, increasing the constraint length of the constituent codes of the rate 1/3 TC from K = 3 to K = 5 would increase the TC decoding complexity by a factor F ~ 112 for each decoding iteration. The simulation results shown in the last four figures (figure 2.9 – 2.12) suggest that the interleaver length rather than the constraint length of the constituent codes is the most efficient parameter to be optimized for minimizing the decoding errors at SNR ~ [0.25 – 2.25 dB].

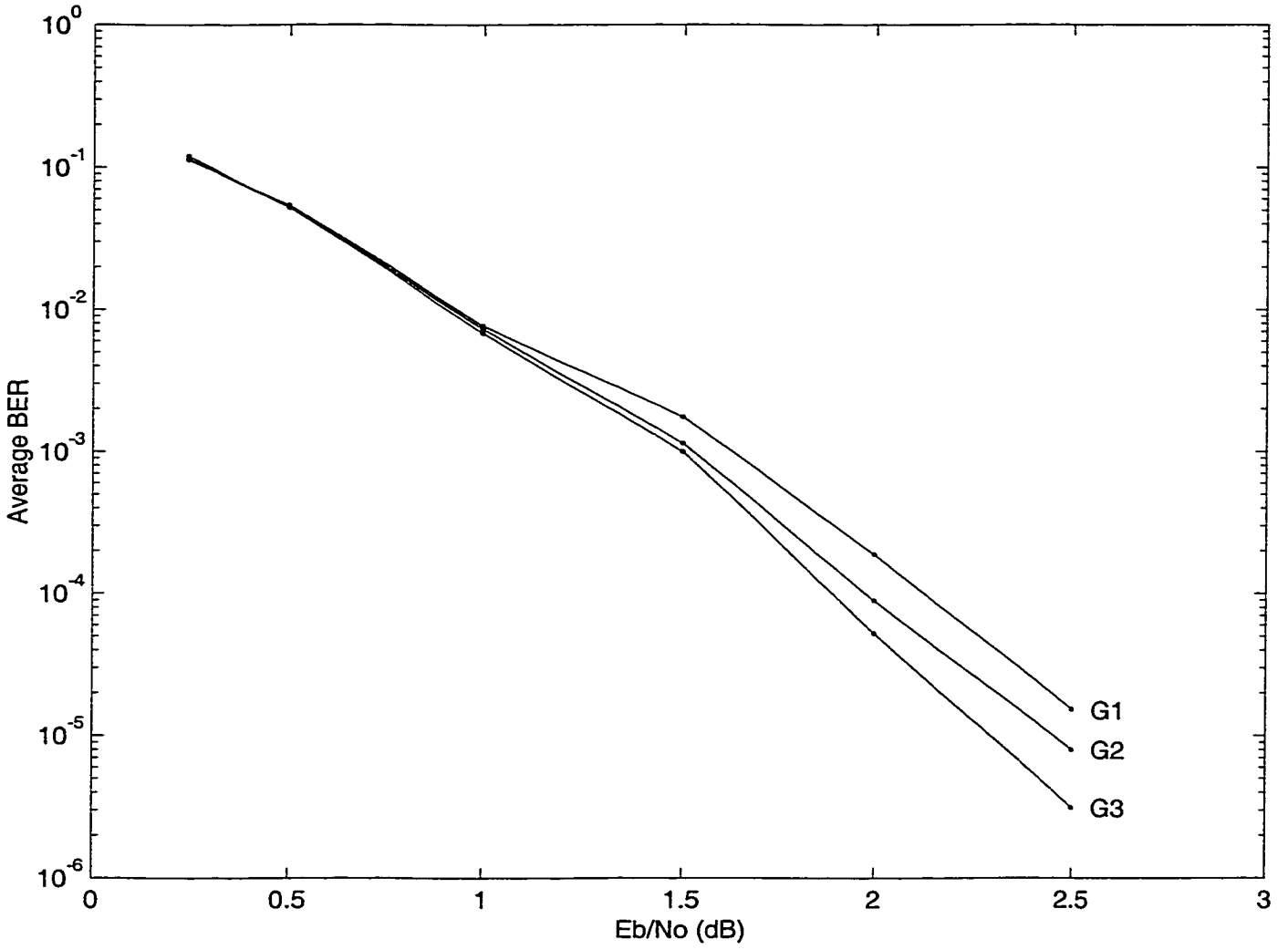


Figure 2.11: Rate 1/3 TC BER performance vs SNR for random interleaver and various G at 5th decoding iteration ($N = 900$; $G1 = [1, 1 + D^2/1 + D + D^2]$; $G2 = [1, 1 + D + D^3/1 + D + D^2 + D^3]$; $G3 = [1, 1 + D^4/1 + D + D^2 + D^3 + D^4]$).

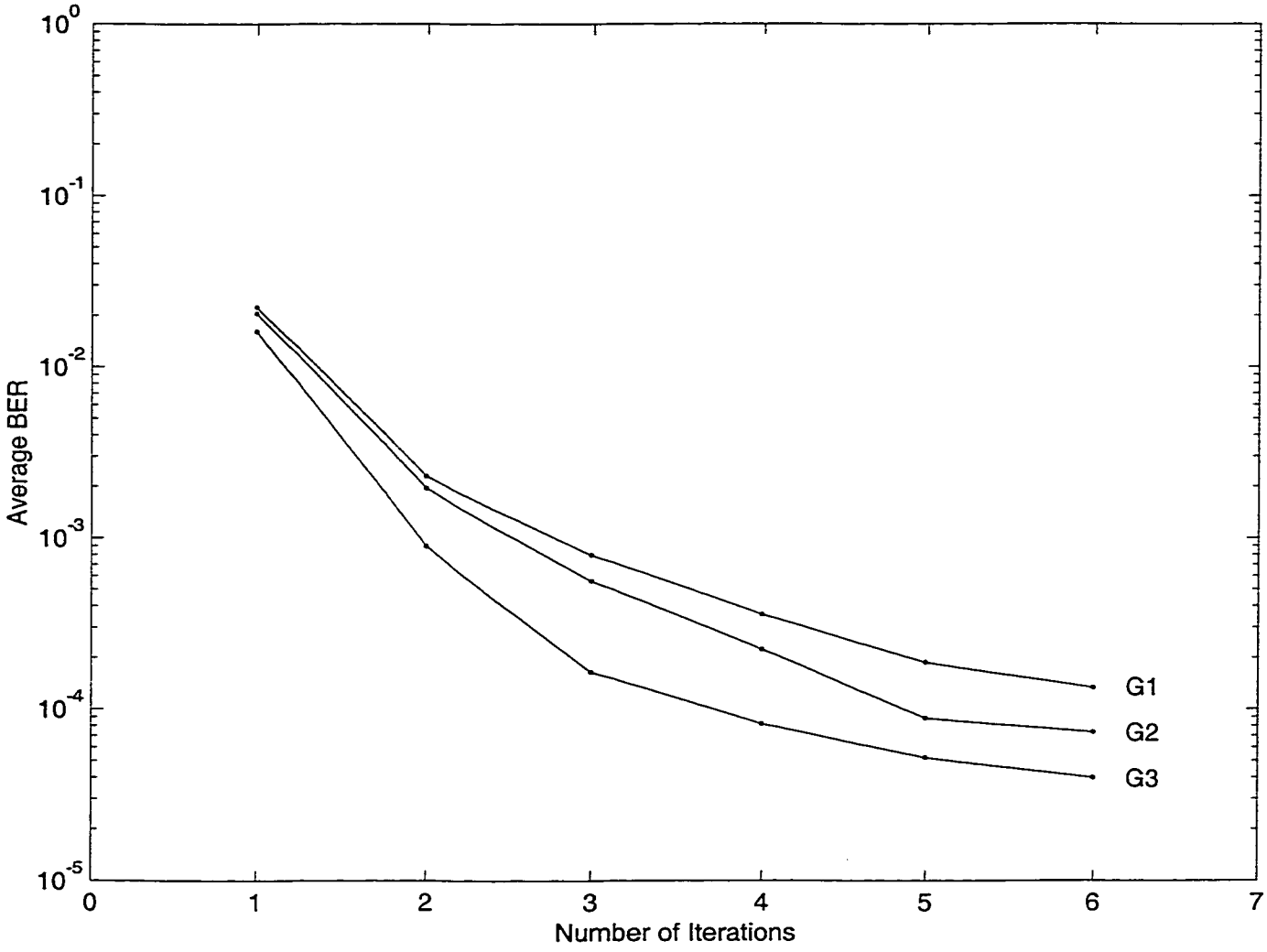


Figure 2.12: Rate 1/3 TC BER performance vs decoding iterations for random interleaver and various G at SNR = 2.0dB ($N = 900$; $G1 = [1, 1 + D^2/1 + D + D^2]$; $G2 = [1, 1 + D + D^3/1 + D + D^2 + D^3]$; $G3 = [1, 1 + D^4/1 + D + D^2 + D^3 + D^4]$).

Figure 2.13 illustrates the BER performances of three Turbo Codes of different rates, i.e. R = 1/3, 1/4 and 1/5, at the fifth decoding iteration. The three Turbo Codes, TC1, TC2 and TC3, are constructed with identical constituent codes ($G = [1, 1+D^2 / 1+D+D^2]$) and random interleavers of length N = 900. Obviously, the TC2 encoder contains two interleavers and three rate 1/2 RSC encoders cascaded in parallel, whereas the TC3 encoder has three random interleavers and four rate 1/2 RSC encoders which are also cascaded in parallel.

The simulation results clearly show that at SNR<2.50 dB, TC1 has the best performance. However, both TC2 and TC3 start to outperform TC1 at SNR $\approx$ 2.50 dB and 3.00 dB respectively. Furthermore, by projecting the TC2(R2=1/4) and TC3(R3=1/5) curves, it is estimated that the rate 1/5 TC eventually outperform the rate 1/4 TC at SNR $\geq$3.50 dB. It is interesting to observe how fast both TC2 and TC3 are catching up with TC1. For instance, the TC2 BER begins with a relatively large value at SNR $\leq$ 1.00 dB but rapidly drops at SNR>1.00 dB. Similarly, the TC3 BER starts with a relatively high value which decreases very slowly at SNR < 2.00 dB, but sharply drops at SNR $\geq$ 2.00 dB . Based on this observation, one can make a general statement on the BER performance of a TC as a function of its code rate at low SNR, ~ (0.25 – 2.50 dB), as following:

i. The optimum TC code rate is R = 1/3.
ii. Reducing the code rate degrades rather than improves the BER.
iii. The smaller the code rate, the larger the range of SNR in which the TC with smaller code rate underperforms the one with optimum rate, i.e. R = 1/3.

The fact that TC1 yields the best performance at SNR ~ [0.25 – 2.20 dB] is not a surprise for a simple and straight forward reason. At the receiving end, the codewords of these three TCs are first segmented to form several rate 1/2 RSC codewords which are in turn decoded by identical rate 1/2 RSC decoders. In a power limited system, the transmitted energy per coded bit for rate 1/3, 1/4 and 1/5 are E_B/3, E_B/4 and E_B/5 respectively, where E_B denotes the energy per information bit. At low SNR, the number of bits received in

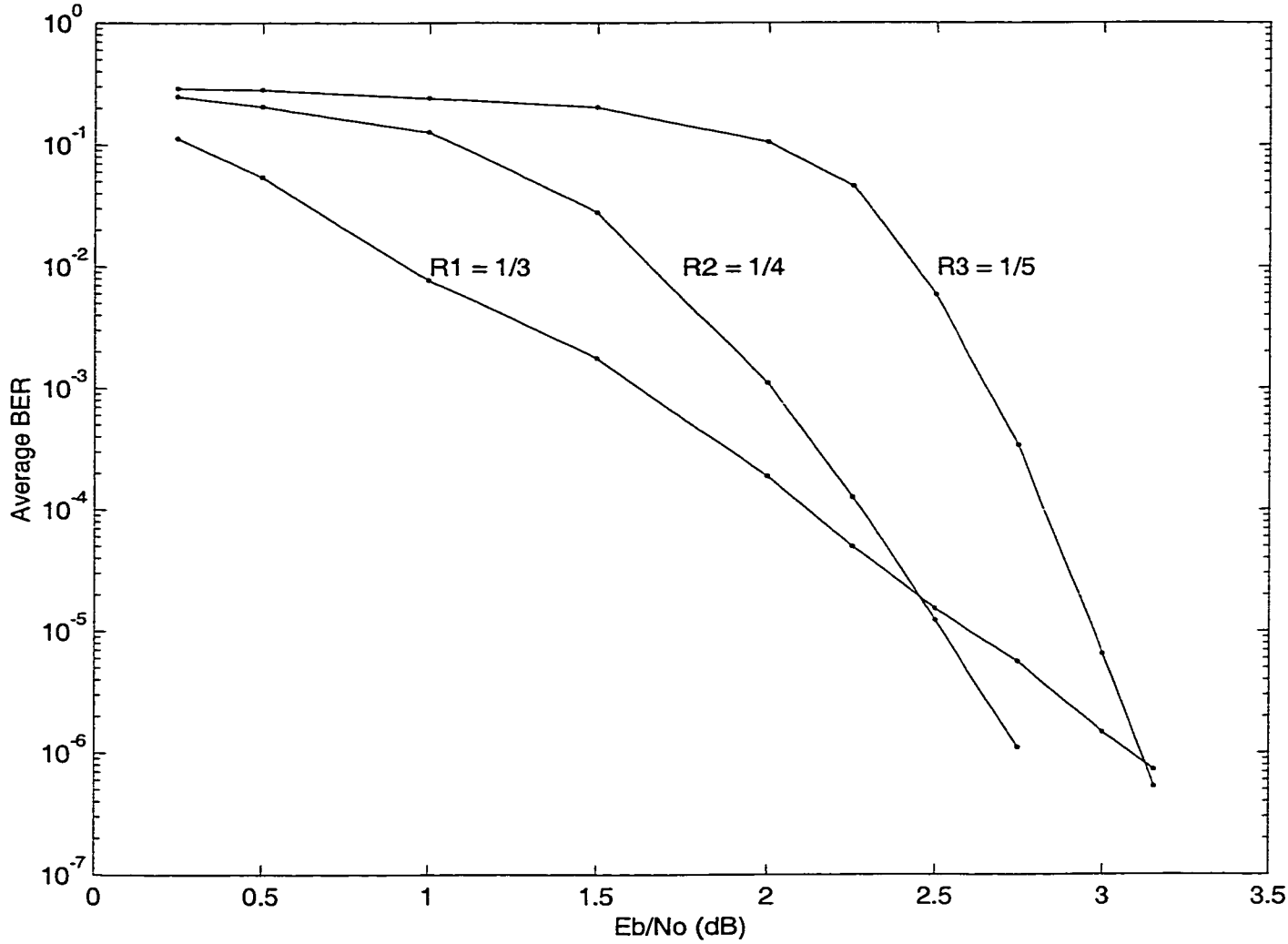


Figure 2.13: BER performances for rate $R_1 = 1/3$, $R_2 = 1/4$ and $R_3 = 1/5$ Turbo Codes with random interleavers at 5th decoding Iteration ($N = 900$; $G = [1, 1+D^2 / 1+D+D^2]$).

errors per rate 1/2 RSC codeword is much beyond the error correction capability of the rate 1/2 RSC decoder. The situation is much worse for rate 1/4 and 1/5, hence resulting in larger decoding errors for rate 1/4 and 1/5 than rate 1/3. At moderate SNR, the number of bit errors per RSC codeword quickly decreases toward the error correction capability of the rate 1/2 RSC decoder for three TCs, hence resulting in better performance for the two higher rate TCs. The superior performances of rate 1/4 and 1/5 at moderate SNR are mainly attributed to the diversity effect of TC. Furthermore, the superior performances of rate 1/4 and 1/5 are amplified by larger amount of decoding efforts for rate 1/4 and 1/5 TCs than for rate 1/3 TC . For instance, three decoding iterations for rate 1/5 TC would require as much as efforts as five decoding iterations for rate 1/3 TC.

Based on the three BER performance curves of figure 2.13, an important conclusion can be made with regard to the impact or the effect of puncturing on the BER performances of rate $1/q$ TC. Hagenauer has shown through simulation that puncturing the rate 1/3 TC to higher rate R ($1/3 < R < 1$) results in a performance degradation across the entire range of SNR [22]. Simulation results of figure 2.13 however suggest that puncturing the rate $1/q$ TC ($q > 3$) does not result in a performance degradation across the entire range of SNR. In fact, it degrades the BER performance at high SNR but improves the BER performance at low SNR. For instance, puncturing the rate 1/4 TC to punctured rate 1/3 TC with a puncturing matrix [1110] is equivalent to removing the third parity sequence, $\boldsymbol{x}_3$, from the unpunctured rate 1/4 TC codewords. The resulting punctured rate 1/3 TC, which is simply the unpunctured rate 1/3, outperforms the unpunctured rate 1/4 at low SNR as shown in figure 2.13. The same observation can be extended to the general rate $1/q$ TC where $q > 3$.

2.5. Conclusion

This chapter provides a comprehensive study of low rates Turbo Codes in the region of low SNR. It is organized in two parts. The first one begins with a generic discussion of the encoding and decoding characteristic of the current Turbo Codes, followed by a detailed analysis of essential properties or parameters such as weight distribution, effective free distance, interleaver length and RSC constraint length which affect the BER performance. The second part mainly involves simulating and discussing the BER performances of several Turbo Codes (TC) with different parameters. The following summarizes the findings of this chapter:

(*i*) The TC BER reduces with the number of iterations. However, it appears that most of the BER reduction is achieved within the first five or six iterations. Furthermore, it seems that there is an error floor for each value of Eb/No. Once such error floor has been reached, additional decoding iteration would yield negligible or no BER improvement.

(*ii*) For a given value of Eb/No and number of decoding iterations, the TC BER can be decreased by increasing the constraint length of the associated RSC encoder and/or the associated interleaver length. It seems, however, that at low SNR, increasing the interleaver length yields better BER improvement than that obtained by increasing the constraint length of the RSC encoder.

(*iii*) Iterative decoding is a very powerful tool which allows or makes it possible to iteratively decrease the BER. However, it appears to be powerless in the absence of interleaving. Indeed, it was shown through simulation that iterative decoding completely loses its strength when the interleavers were removed from the rate 1/3 Turbo Codes encoder and decoder, despite the fact that the weight distribution of the corresponding rate 1/3 Turbo Codes was maintained relatively unchanged.

(*iv*) Unlike the BER performances of other existing codes, the BER performance of Turbo Codes with iterative decoding is not dominated by the minimum or free distance. Nor it is dominated by the small number of codewords with minimum or free distance. It is the effect of interleaving which is the dominant factor responsible for the outstanding BER performance.

(*v*) In the low region of SNR, ~ [0 – 2.50 dB], simulation results indicated that the optimum unpunctured TC code rate, i.e. the one which yields the lowest BER, is 1/3. For the general unpunctured rate $1/q$ TC where $q > 3$, the larger the q value, the larger the Eb/No value below which the unpunctured rate $1/q$ underperformed the unpunctured rate 1/3.

(*vi*) The unpunctured rate 1/3 TC can be punctured to produce higher code rate. This is achieved at the expense of the BER performance degradation across the entire range of SNR [22]. Similarly, the unpunctured rate $1/q$ TC where $q > 3$ can also be punctured to increase the code rate. However, unlike puncturing the rate 1/3, puncturing the rate $1/q$ does not yield performance degradation across the entire range of SNR. In fact, puncturing the rate $1/q$ results in a BER improvement in the region of low SNR.

Chapter 3

Variants of Turbo Codes

3.1. Introduction

The parity bits of the rate $1/q$ Turbo Codes can be punctured to increase the overall code rate. Hagenauer, Offer and Papke conducted a study on the effect of puncturing on the BER performance of the rate 1/3 Turbo Codes whereby the parity bits were punctured to increase the overall code rate from 1/3 to 4/5 [22]. They showed that puncturing the parity bits of the rate 1/3 Turbo Codes degrades the BER performances across the entire range of SNR. As such, puncturing is not considered as a mean for improving the BER performance of the unpunctured rate 1/3 TC.

In the previous chapter, it was shown through simulations that in the low region of SNR, ~ (0.25 – 2.50dB), the current unpunctured rate 1/3 Turbo Codes (TC) produce optimum BER performance compared to those obtained with lower code rates, for example the unpunctured rate 1/4 and 1/5. Furthermore, it was briefly argued that in principle, it would be possible to enhance the performance of the current unpunctured rate 1/3 TC in the low region of SNR, ~ (0.25 - 2.50dB). This chapter investigates such possibility by introducing and studying some TC variants which primarily consist of enhancing the reliability of the parity sequence of the unpunctured rate $1/q$ Turbo Codes where $q \geq 3$. In particular, it will present the design, simulation results and BER performance comparisons between the TC variants of rate 1/3, 1/4 and 1/5 and the conventional unpunctured rate 1/3, 1/4 and 1/5 TC. This study mainly focuses on the rate 1/3, 1/4 and 1/5 TC. However, the extension to the general rate $1/q$ TC should be straightforward.

3.2 Parity Enhancement

3.2.1 Overview

For a rate $1/q$ TC, the information sequence $\boldsymbol{u}$ is independently encoded $(q\text{-}1)$ times by rate 1/2 RSC encoders to produce a codeword containing the original data sequence $\boldsymbol{u}$ and $(q - 1)$ parity sequences, $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \ldots, \boldsymbol{x}_{q-1}$, multiplexed as illustrated in figure 3.1. The sequence $\boldsymbol{u}$ is error protected by the $(q\text{-}1)$ parity sequences in the sense that some bit errors in $\boldsymbol{u}$ caused by the channel noise can be corrected through iterative decoding of the codeword pairs $(\boldsymbol{u}, \boldsymbol{x}_1), (\boldsymbol{u}'_2, \boldsymbol{x}_2), (\boldsymbol{u}'_3, \boldsymbol{x}_3), \ldots, (\boldsymbol{u}'_{q-1}, \boldsymbol{x}_{q-1})$, where $\boldsymbol{u}'_2 \neq \boldsymbol{u}'_3 \neq \ldots \neq \boldsymbol{u}'_{q-1}$ denote different interleaved versions of $\boldsymbol{u}$. The parity sequences themselves are however not error protected, and as such they are likely to be corrupted by the channel noise as much as the sequence $\boldsymbol{u}$ is.

u_1	$x_{1,1}$	....	$x_{q-1,1}$	....	....	....	u_N	$x_{1,N}$	....	$x_{q-1,N}$

Figure 3.1: The bit stream of a rate $1/q$ TC codeword.

The fact that the $(q\text{-}1)$ parity sequences are not error protected, can adversely affects the decoding performance, the error floor in particular, since the very same corrupted parity sequences $(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \ldots, \boldsymbol{x}_{q-1})$ are repeatedly used to decode $\boldsymbol{u}$. At the beginning, the decoder is able to correct some of the errors and produce an improved estimate of $\boldsymbol{u}$. But as the decoding goes on and the BER reaches a certain level, referred to as an error floor, further decoding will not yield any improvement.

The reason why the BER stops to decrease after a certain number of decoding iterations can be explained as following. Initially, the received $\boldsymbol{u}$, $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, $\boldsymbol{x}_3$, ..., $\boldsymbol{x}_{q-1}$ which are equally affected by the noisy communication channel, have roughly the same number of bits in error, assuming independent and identically distributed noise samples. If the number of bits in error is relatively small, the decoder is able to correct some errors in the received $\boldsymbol{u}$ by decoding the codeword pairs $(\boldsymbol{u}, \boldsymbol{x}_1), (\boldsymbol{u}'_2, \boldsymbol{x}_2), (\boldsymbol{u}'_3, \boldsymbol{x}_3), ..., (\boldsymbol{u}'_{q-1}, \boldsymbol{x}_{q-1})$. The decoded or estimated sequence $\boldsymbol{u}$ which has normally fewer errors than the received $\boldsymbol{u}$, can provide extrinsic information about the transmitted $\boldsymbol{u}$ to further reduce the BER at the next decoding iteration. While at a code rate below the channel capacity, the BER of the estimated sequence $\boldsymbol{u}$ is getting smaller after every additional decoding iteration until it reaches the error floor, the reliability values of the received parity sequences remain unchanged. As the number of errors in the decoded $\boldsymbol{u}$ is much smaller than those in the received parity sequences, the decoding performance for the codewords $(\boldsymbol{u}, \boldsymbol{x}_1)$, $(\boldsymbol{u}'_2, \boldsymbol{x}_2), (\boldsymbol{u}'_3, \boldsymbol{x}_3), ..., (\boldsymbol{u}'_{q-1}, \boldsymbol{x}_{q-1})$ are strongly dominated by the number of errors in the received parity sequences $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, ..., \boldsymbol{x}_{q-1}$ rather than the number of errors in estimated sequence $\boldsymbol{u}$, thus explaining the absence of the decoding improvement after a number of decoding iterations.

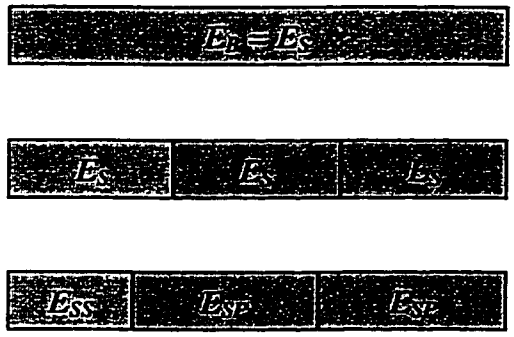
Based on the above principle or reasoning, it is believed that the BER performance of the unpunctured rate $1/q$ TC can be improved by increasing the immunity of the $(q - 1)$ parity sequences $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_{q-1}$ against the channel noise while keeping the noise immunity of $\boldsymbol{u}$ intact. Two possible but distinct approaches can be used to enhance the noise immunity of the parity sequences, i.e. increasing their transmitted power or adding new parity bits to the existing ones. The following studies the applications of these two approaches to the parity sequences of unpunctured rate 1/3, 1/4 and 1/5 Turbo Codes, and their impacts on the BER performance of such Turbo Codes. Again, the extension of these applications to the rate $1/q$ TC is straightforward.

3.2.2 Parity Enhancement with Unequal Energy Partitioning

This section investigates and studies the effect of raising the reliability values of the parity bits of the unpunctured rate 1/3 Turbo Codes on the BER performance. The results of the study can be extended or applied to all Turbo Codes regardless of their code rates. Here, unpunctured rate 1/3 Turbo Codes are specifically selected for the study because of their optimum performances in the low region of SNR compared to the other unpunctured rates.

A simple way to increase the reliabilities or noise immunity of the parity sequences is to raise the energy transmitted of the corresponding parity bits. Unfortunately, this approach also increases the overall average transmitted energy and hence may not be practical for power limited communications such as satellite transmission which is already operating at the maximum output power. As such, one needs to consider a different approach which raises the transmitted energy of the individual parity bits without actually increasing the overall average transmitted energy. To achieve this, a technique called unequal energy partitioning is proposed.

Consider a conventional rate $1/q$ TC encoder which encodes a single information bit into q coded bits. To maintain the same information rate, a transmitter must transmit q coded bits per every information bit interval. If E_B denotes the total transmitted energy available per information bit interval, the transmitted energy per coded bit, E_S, is simply $(1/q)E_B$. In conventional rate $1/q$ TC, all coded bits are treated as of equal importance in the sense that the available energy E_B is equally divided among the three coded bits: equivalently, each coded bit is transmitted with the same amount of energy $(1/q)E_B$. However, the view taken here is that the parity bits should be of higher importance than the systematic bits because the same received parity sequences are repeatedly decoded to reduce errors in the decoded systematic sequence. It would therefore be desirable that the received parity sequences contain as few errors as possible. Such view leads to the idea or concept of unequal energy partitioning for rate 1/3 TC as illustrated in figure 3.2. The same concept can also be applied to the generic rate $1/q$ TC.



a) uncoded transmission

b) conventional coded transmission

c) proposed unequal energy partitioning

Figure 3.2: Comparison between various transmission schemes.

Unequal energy partitioning technique can be viewed as an unequal distribution of energy among coded bits according to their order of importance. Obviously, the more important coded bits are transmitted with high energy while the less important ones are transmitted with low energy. For the rate $1/q$ TC, the $(q - 1)$ parity bit streams are considered as of equal importance and each corresponding parity bit is transmitted with energy E_{SP} ($E_{SP} < E_B$). The systematic sequence is, on the other hand, considered as less important than the $(q - 1)$ parity sequences. As such, each corresponding systematic bit is transmitted with energy E_{SS}, where $E_{SS} < E_{SP}$. The following relationships are obtained for the proposed energy partitioning

$$E_B = E_{SS} + (q - 1) E_{SP} \qquad (3.1)$$

The goals of unequal energy partitioning is twofold - to change the error distribution within the individual TC codewords, and to keep the average number of errors per each codeword approximately the same as if no energy partitioning is used, i.e. $E_{SS} = E_{SP}$. Conceptually, changing the values of E_{SS} and E_{SP} affects the distribution of errors within the received codewords, i.e. shifting some errors from one segment to the other segment of codeword. In our case, the number of errors in the parity sequences is kept relatively low compared to that in the systematic sequence as a result of raising E_{SP} and lowering E_{SS}. Obviously, the challenge is how to change the error distribution without significantly

increasing the average number of errors per codeword. If the adjustment of E_{SS} and E_{SP} values are not done properly, one may end up having less errors in the parity segments but at the same time significantly more errors per codeword. It is quite obvious that the larger the number of errors per codeword, the larger the probability of decoding error. As such, it is essential to maintain an average number of errors per codeword roughly constant, otherwise the unequal energy partitioning technique will degrade rather than improve the BER performance as will be shown through simulations later.

So without violating eq. (3.1), the question is how much should one deviate E_{SS} and E_{SP} from $(1/q)E_B$, or equivalently what would be the optimum values for E_{SS} and E_{SP} ? The answer depends on several factors including the range of SNR, the interleaver length and the error correction capability of the constituent codes. The theoretical evaluation of optimum E_{SS} and E_{SP} can be very challenging. Instead of computing the exact theoretical values, several pairs of values for (E_{SS}, E_{SP}) are estimated and simulated to determine the pair of values which yield better BER performance than the conventional transmission. Since a SNR change of less than 0.1dB has negligible impact on the BER performance, one only needs to estimate and simulate a few energy pairs of (E_{SS}, E_{SP}) in order to find a good one.

As mentioned earlier, the optimum values for (E_{SS}, E_{SP}) depend on several factors which include the SNR range, the constituent codes and the interleaver length. In the following, the BER performances of the conventional unpunctured rate 1/3 TC are compared to those obtained with energy partitioning for a given interleaver length and various energy pairs of (E_{SS},E_{SP}). The conventional unpunctured rate 1/3 TC without energy partitioning, i.e. $E_{SS} = E_{SP} = (1/3)E_B$, is used as a benchmark for comparison. Finally, the simulations make use of the following set of values:

- Constituent codes: $G = [1,\ 1 + D^2 / 1 + D + D^2\ ]$.
- Random interleaver of length: $N = 500$.
- Number of iterations = 10.
- SNR = (0.25 – 2.50 dB).
- Normalized $(E_{SS}, E_{SP}) = \{(1/4, 3/8), (1/5, 2/5), (1/6, 5/12)\}$.

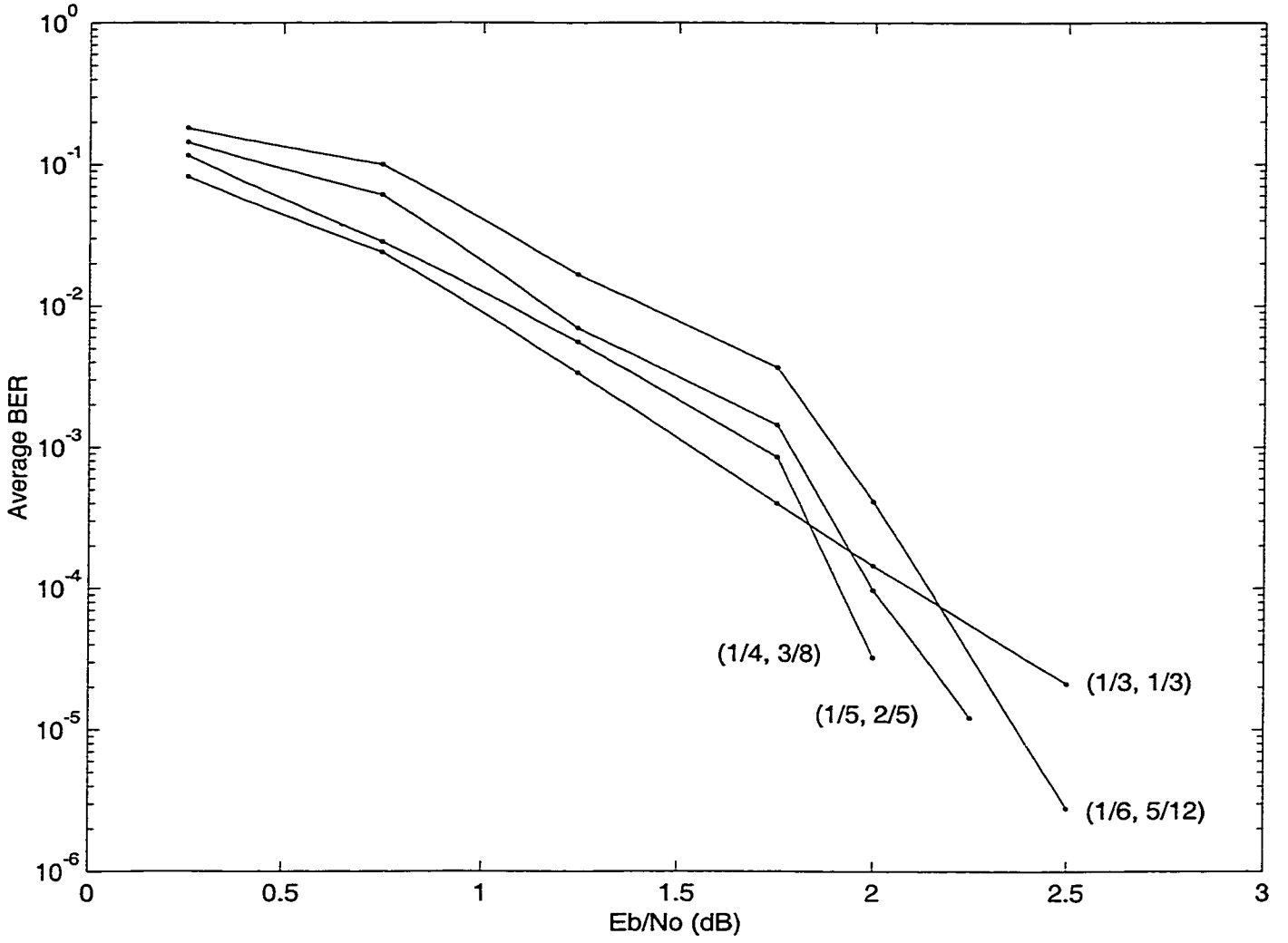


Figure 3.3: BER performances for original rate 1/3 TC and rate 1/3 TC with unequal energy partitioning at 10th decoding iteration (random interleaver $N = 500$ and generator $G = [1+D^2/1+D+D^2]$).

Figure 3.3 plots and compares the performance of the original rate 1/3 TC to those of the rate 1/3 TC with unequal energy partitioning for $N = 500$ at tenth decoding iteration. This figure clearly shows that the rate 1/3 TC with unequal energy partitioning can outperform the original rate 1/3 TC. At low SNR, i.e. roughly 1.75 dB and below, the rate 1/3 TC with unequal energy partitioning seems to underperform the original rate 1/3 TC. The trend is, however, reversed at larger SNR. A possible explanation is that at SNR < 1.75dB, reducing E_{SS} from (1/3) to (1/4)E_B or lower would result in a substantial increase of the average number of errors per received codeword, hence worsening the decoding performance. As the SNR increases, unequal energy partitioning would become more effective in the sense that it would be able to change the error distribution in favor of the parity bits with minor increase of average errors per received codeword. It is interesting to note that unequal energy partitioning with the normalized pair (E_{SS}, E_{SP}) = (1/4, 3/8) yields the best BER performance among the three normalized unequal energy partitioning pairs. Furthermore, there seems to be a correlation between each E_{SS} value of the normalized unequal energy partitioning pairs (E_{SS}, E_{SP}), and the SNR value above which the TC with corresponding unequal energy partitioning outperforms the original TC. Finally, it was observed during the simulation that the benefit of the unequal energy partitioning becomes increasingly visible as the number of decoding iteration increases.

Figure 3.4 depicts the BER performance of rate 1/3 TC with different values of SNR and normalized E_{SS} for $N = 500$ at fifth decoding iteration. These curves provide a useful mean to determine the optimum (E_{SS}, E_{SP}) for this particular interleaver length N and an operating SNR. For instance at SNR = 2.0 dB, the normalized E_{SS} = 1/4 yields the best BER and the corresponding optimum pair of (E_{SS}, E_{SP}) is (E_B/4, 3E_B/8). For clarity, a normalized value $E_{SS} = 1$ corresponds to uncoded transmission, i.e. no parity bit is transmitted or more precisely, the bit energy E_B is entirely allocated to the systematic bit ($E_{SS} = 1$), leaving no energy for the parity bits ($E_{SP} = 0$). Finally, it is worth mentioning that the proposed unequal energy partitioning scheme represents an efficient technique for improving the BER performance of the conventional rate 1/3 TC without actually increasing the system average output power, system bandwidth and decoder complexity.

The only disadvantage is that it requires a transmitter with two different transmitting powers.

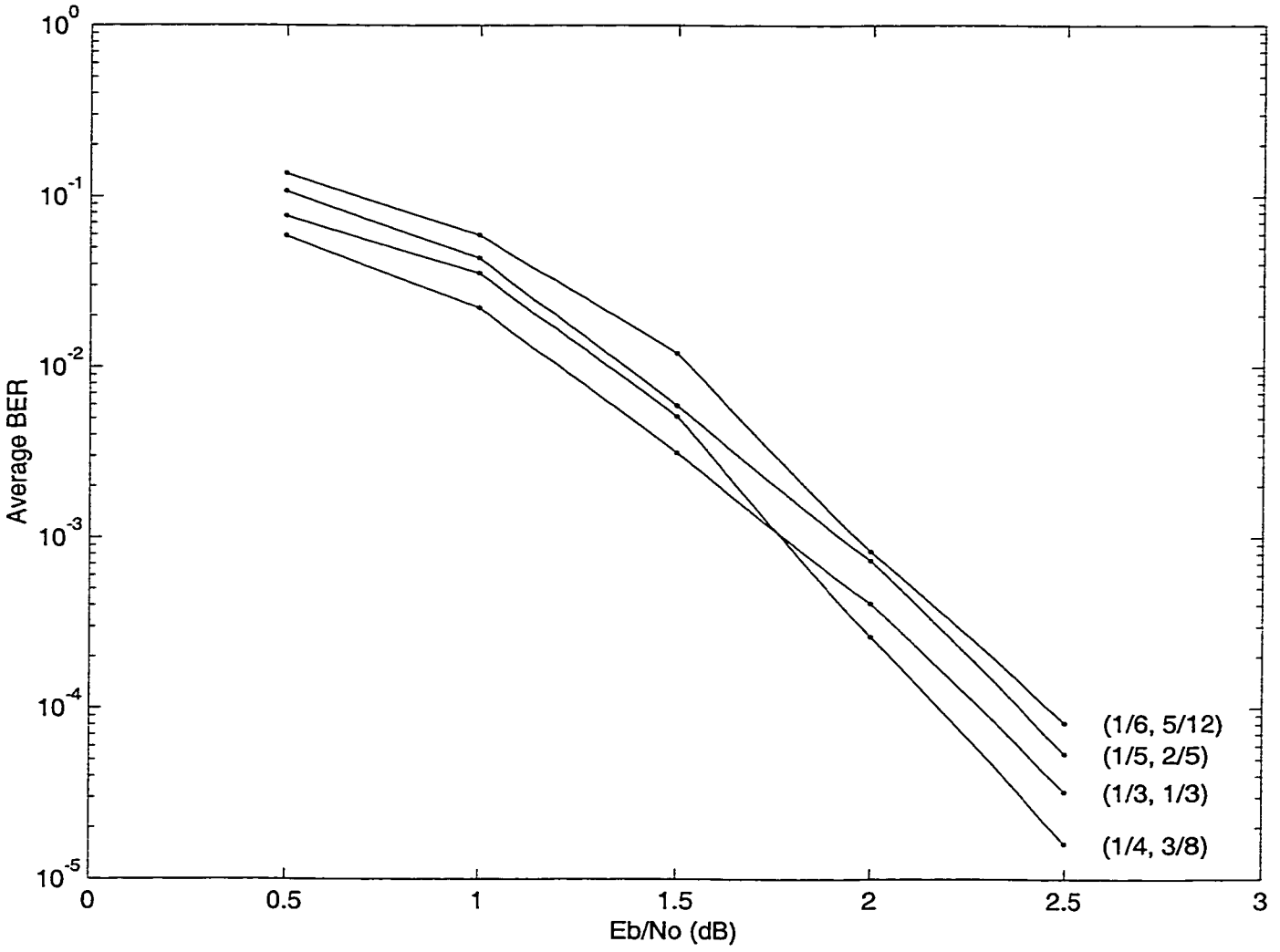


Figure 3.4: BER performances versus normalized E_{SS} for rate 1/3 TC with energy partitioning at 5th decoding iteration (N = 500; G= [1+D^2/1+D+D^2]; E_{SS} = {1/8, 1/6, 1/5, 1/4, 1/3, 1/2, 2/3, 3/4, 4/5, 1}).

3.3 Parity-over-Parity

3.3.1 Full Parity-over-Parity

Section 3.2.2 suggests using unequal energy partitioning as a simple and efficient mean for increasing the reliability of the received parity bits. In this section, an alternative approach to the previous unequal energy partitioning scheme is proposed for study. This alternative scheme is referred to as full parity-over-parity. As implied by its name, this scheme attempts to improve the reliabilities of the parity bits using the parity-over-parity encoding.

Under this scheme, the two parity sequences of the conventional unpunctured rate 1/3 TC are independently encoded by two rate 1/2 RSC encoders (ENC 3 and ENC 4) as depicted in figure 3.5, and the overall code rate is reduced from 1/3 to 1/5. The five sequences u, x_1, x_2, x_3 and x_4 are multiplexed to form a single bit stream for transmission. The BER performance of the rate 1/5 parity-over-parity Turbo Codes are simulated and compared to that of the conventional unpunctured rate 1/5 Turbo Codes. The reason for comparing this modified rate 1/5 to the conventional rate 1/5 Turbo Codes is mainly because the reliability values of the received systematic bits are the same for both codes, while the reliability values of the estimated parity bits of the modified rate 1/5 TC are larger than those of the conventional rate 1/5 TC as a result of the extra parity encoding.

For simplicity, all constituent encoders (ENC1, ..., ENC4) of figure 3.5 are chosen to be identical rate 1/2 RSC encoders. It is interesting to observe that the modified rate 1/5 TC with parity-over-parity encoder of figure 3.5 possesses a structure similar to that of an encoder for hybrid concatenated codes which encompasses both serial and parallel concatenations. For instance, the three sequences (u, x_1, x_3) alone can be viewed as a serial concatenated coded (SCC) codeword where (u, x_1) and (u, x_1, x_3) correspond to the inner and outer codes respectively. Likewise, (u', x_2, x_4) alone can be seen as another SCC codeword with (u', x_2) and (u', x_2, x_4) representing the associated inner and outer

codes respectively. The two triplets (u, x_1, x_3) and (u', x_2 , x_4) are multiplexed and punctured to form a single parallel codeword (u, x_1, x_2, x_3, x_4).

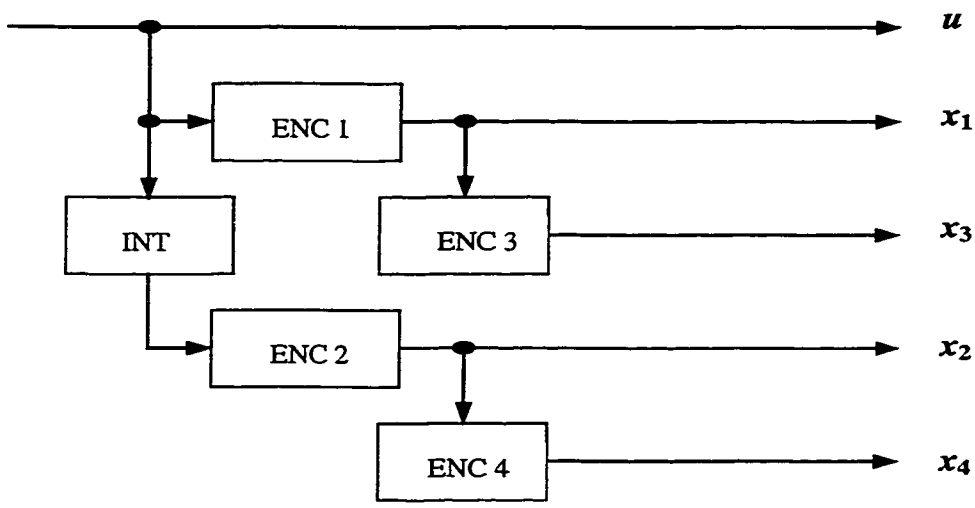


Figure 3.5: A modified rate 1/5 TC with full parity-over-parity encoder.

At the receiver, the parallel codeword (u, x_1, x_2, x_3, x_4) is de-multiplexed and interleaved to produce the two SCC codewords (u, x_1, x_3) and (u', x_2, x_4), which are in turn decoded to produce $(u, \tilde{x}_1)$ and $(u', \tilde{x}_2)$, where u' denotes the interleaved version of u and $\tilde{x}_2$ is the estimate of x_2. Finally, $(u, \tilde{x}_1)$ and $(u', \tilde{x}_2)$ are alternately and iteratively decoded similar to the decoding of the conventional unpunctured rate 1/3 TC. All the encoding and decoding works to obtain $(\tilde{x}_1, \tilde{x}_2)$ are carried out such that decoded parity sequences are likely more reliable than the parity sequences of the conventional rate 1/5 TC. As such, the modified rate 1/5 could yield better performance than the conventional rate 1/5 at low SNR. At moderate to large SNR region, it is however anticipated that the conventional rate 1/5 will outperform the modified rate 1/5, just as it outperforms the conventional unpunctured rate 1/3.

Figure 3.7 depicts the simulation results of the conventional rate 1/5 TC, and that of the modified rate 1/5 TC at fifth iteration. The two TCs are constructed with identical random interleavers of length $N = 900$ and rate 1/2 RSC with generator polynomial $G = [1, 1+D/1+D+D^2]$. Figure 3.7 shows that the modified rate 1/5 TC slightly outperforms the conventional rate 1/5 TC at SNR$\leq$ 1.75 dB. However, the latter starts to yield better performance at SNR > 1.75 dB. It is also observed from the graph that the performance difference between the two TC rapidly widen. At BER roughly 5.0×10^{-5}, the conventional rate 1/5 TC is almost 1.5dB better than the modified rate 1/5 TC after five decoding iterations. The positive or desirable aspect of the modified rate 1/5 TC is its decoding computations, which is approximately 2/3 of the conventional rate 1/5 TC's.

Some important remarks or explanations deserves mentioning about the simulation results obtained in figure 3.6. First, it is noted that the received systematic and parity sequences of the modified rate 1/5 and those of the conventional rate 1/5 are equally reliable since they are all transmitted with equal energy through the same noisy channel. However, the estimated parity sequences $\tilde{\boldsymbol{x}}_1$ and $\tilde{\boldsymbol{x}}_2$ of the modified rate 1/5, which are used to iteratively decode the received systematic sequence $\boldsymbol{u}$ are likely to contain fewer errors than the received parity sequences of the conventional rate 1/5, hence should favor the performance of the modified rate 1/5.

At moderate to high SNR, $\tilde{\boldsymbol{x}}_1$ and $\tilde{\boldsymbol{x}}_2$ are also more reliable than the received parity sequences of the conventional rate 1/5, yet the modified rate 1/5 performs poorly compare to the conventional rate 1/5. The reason of this inferior performance is possibly due to the fact that at moderate to high SNR, the reliability gain in the estimated parity sequences of the modified rate 1/5 is not enough to compensate the lost or reduction of the diversity provided by the parity sequences. Recall that in the modified rate 1/5, the received systematic sequence is iteratively decoded with only two (estimated) parity sequences, whereas in the conventional one, the same received systematic sequence is iteratively decoded with four parity sequences. At low SNR, the systematic and the parity sequences are strongly corrupted by noise, hence decoding the systematic with two or

four parity sequences yield little difference. However, at moderate to high SNR, each decoding with an additional parity sequence produces a significant improvement. As a summary, the main point to retain in this analysis is that increasing the reliability of the parity sequences at the expense or reduction of its diversity degrade the overall BER performance of the rate 1/5 TC at moderate to high SNR. The same result can also be extended to rate $1/q$ TC as will be confirmed with the rate 1/4 TC in the next section.

3.3.2 Joint Parity-Over-Parity

The objective of this section is twofold: first to demonstrate that the effect of parity-over-parity encoding/decoding on the overall BER performance is similar for both rate 1/4 and 1/5 TCs, hence allowing one to extend the result to the generic rate $1/q$ TC with a higher degree of confidence; and second and more importantly to study the compromise or the effect of varying the parity diversity versus the increase of the estimated received parity. To accomplish this, we propose a modified rate 1/4 TC in which the two parity sequences of the conventional rate 1/3 TC are jointly encoded by a single rate 2/3 RSC encoder (ENC3) to produce a new parity sequence as illustrated in figure 3.6. The BER performance of this so-called modified rate 1/4 TC will be simulated and compared to that of the conventional rate 1/4 TC.

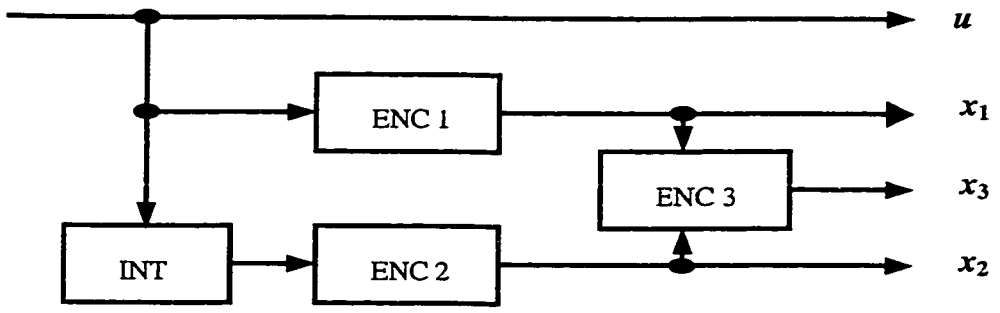


Figure 3.6: A modified rate 1/4 TC with joint parity-over-parity encoder.

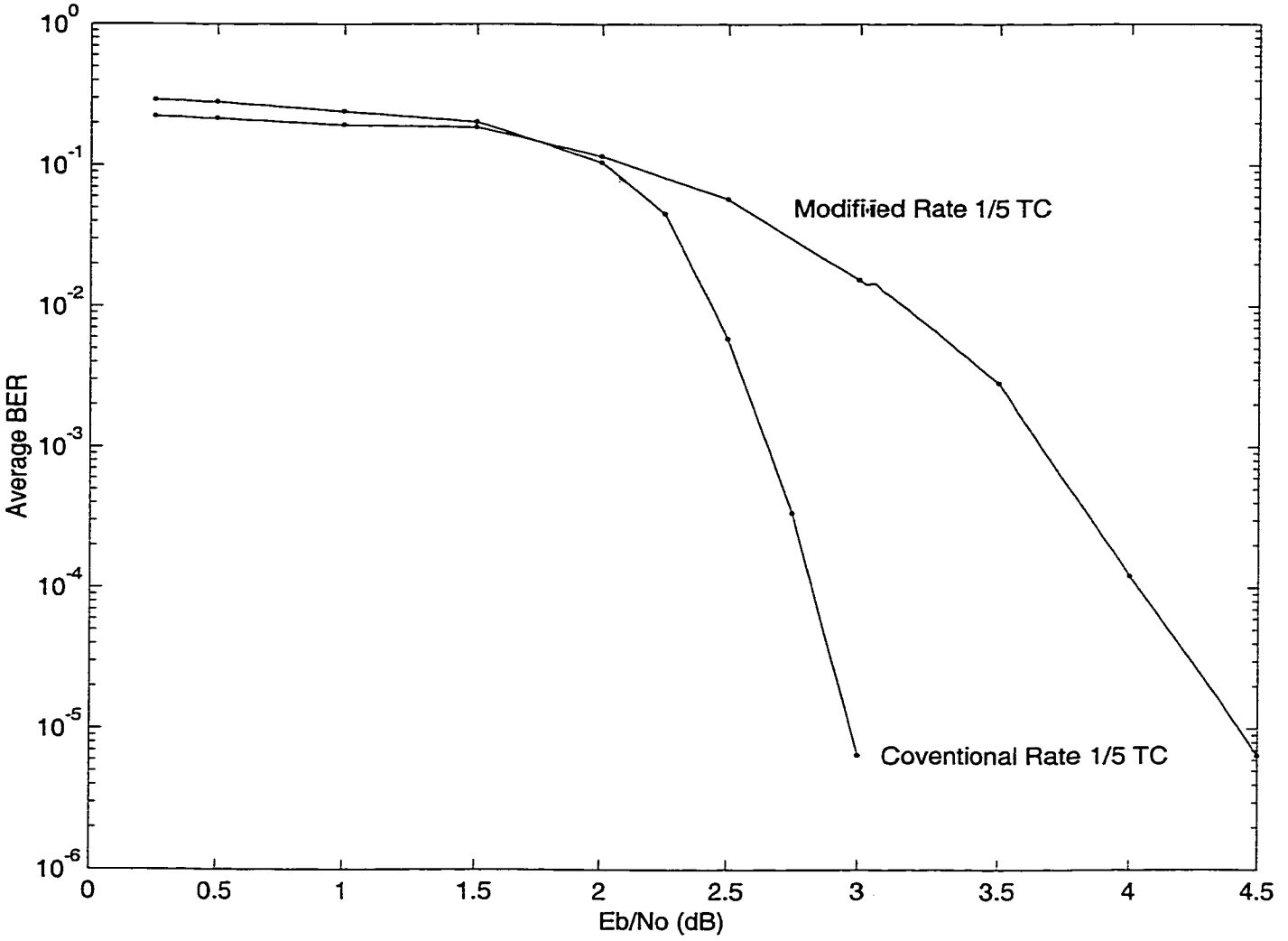


Figure 3.7: BER performances for the conventional rate 1/5 TC and the modified rate 1/5 TC with full parity-over-parity at 5th decoding iteration (random interleaver $N = 900$; $G = [1+D^2/1+D+D^2]$).

The encoder of figure 3.6 encodes a single information block $\boldsymbol{u}$ into a rate 1/4 codeword ($\boldsymbol{u x_1 x_2 x_3}$). For clarity, ENC1 and ENC2 are two identical rate 1/2 RSC encoders which only output parity sequences. Similarly, ENC3 is a rate 2/3 RSC encoder which outputs a single parity-over-parity sequence. At the receiving end, the decoder performs soft decision decoding on ($\boldsymbol{x_1 x_2 x_3}$) to produce $\tilde{\boldsymbol{x}}_1$ and $\tilde{\boldsymbol{x}}_2$ which correspond to the estimate of $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ respectively. The two estimated parity sequences, $\tilde{\boldsymbol{x}}_1$ and $\tilde{\boldsymbol{x}}_2$, are subsequently multiplexed with $\boldsymbol{u}$ to produce two rate 1/2 RSC codewords ($\boldsymbol{u}\,\tilde{\boldsymbol{x}}_1$) and ($\boldsymbol{u}'\,\tilde{\boldsymbol{x}}_2$), which are alternatively and iteratively decoded as in the conventional rate 1/3 TC. The challenge is to perform soft decision decoding on the rate 2/3 RSC codeword ($\boldsymbol{x_1 x_2 x_3}$) to minimize the errors in the parity sequences separately rather than jointly, i.e. minimizing the errors in the individual $\tilde{\boldsymbol{x}}_1$ and $\tilde{\boldsymbol{x}}_2$ rather than in the pair ($\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2$). To achieve this, we introduced a modified Soft Output Viterbi Algorithm (SOVA) as derived below.

Suppose that an information bit u is transmitted over a noisy channel and received as y. As shown in appendix A, the soft decoded bit, $L(\tilde{u})$, produced by a SISO decoder, is given by

$$L(\tilde{u}) = L(u) + y \cdot L_C + L_E(\tilde{u}) \tag{3.2}$$

$$L(u) = \ln\left[\frac{P(u=1)}{P(u=-1)}\right] \tag{3.3}$$

$$L_C = 4 \cdot a \cdot (E_S / N_0) \tag{3.4}$$

where $a = 1$ for AWGN channel, (E_S / N_O) is the per bit transmitted energy to noise ratio, and L_C and $L_E(\tilde{u})$ are the channel reliability and the extrinsic information respectively. The above equation is used for deriving the extrinsic information for the input bits of a rate 2/3 convolutional code. Consider two input streams ($\boldsymbol{x_1}$, $\boldsymbol{x_2}$) which are "almost"

statistically independent and which are encoded by a rate 2/3 RSC encoder to produce three sequences (x_1, x_2, x_3). The three sequences are first multiplexed into a single sequence X, then transmitted over an AWGN channel and received as Y. At the receiving end, the VA decoder searches for the state sequence $S^{(\rho)}$, or equivalently the desired input sequences ($x_1^{(\rho)}$, $x_2^{(\rho)}$), which maximizes the a posteriory probability over all possible paths ρ, i.e.

$$\max_{\rho} P(S^{(\rho)} | Y)$$

Since Y is independent of ρ, this is equivalent to $\max_{\rho} p(Y|S^{(\rho)})\, P(S^{(\rho)})$. For clarity, we denote $Y = (Y_1, Y_2, \ldots, Y_k, \ldots, Y_N)$ and $S^{(\rho)} = (S_1^{(\rho)}, \ldots, S_N^{(\rho)})$ where the index N represents the number of branches, $Y_k = (Y_1, Y_2, \ldots, Y_k)$ and $S_k^{(\rho)} = (S_1^{(\rho)}, S_2^{(\rho)}, \ldots, S_k^{(\rho)})$. At time $t = k$, we have $Y_k = (y_{1,k}, y_{2,k}, y_{3,k})$ and

$$\begin{aligned} P(S_k) &= P(S_{k-1}) \cdot P(x_{1,k}, x_{2,k}) \\ &= P(S_{k-1}) \cdot P(x_{1,k}) \cdot P(x_{2,k}) \end{aligned} \tag{3.5}$$

since $x_{1,k}$ and $x_{2,k}$ are almost statistically independent. Furthermore, assuming that the channel is memoryless, it follows

$$p(Y_k | S_k^{(\rho)}) = \prod_{l=0}^{l=k} p(Y_l | S_{l-1}^{(\rho)}, S_l^{(\rho)}) \tag{3.6}$$

and

$$\begin{aligned} p(Y_k | S_{k-1}^{(\rho)}, S_k^{(\rho)}) &= p(Y_k | X_k^{(\rho)}) \\ &= \prod_{m=1}^{3} p(y_{m,k} | x_{m,k}^{(\rho)}) \end{aligned} \tag{3.7}$$

It follows that:

$$\max_{\rho} p(Y_k, S_k^{(\rho)}) = \max_{\rho} p(Y_k \mid S_k^{(\rho)}) \cdot P(S_k^{(\rho)})$$

$$= \max_{\rho} \left\{ P(S_{k-1}^{(\rho)}) \prod_{l=0}^{k-1} p(Y_l \mid S_l^{(\rho)}, S_{l-1}^{(\rho)}) \right.$$

$$\left. P(x_{k,1}^{(\rho)}) \cdot P(x_{k,2}^{(\rho)}) \cdot p(Y_k \mid X_k^{(\rho)}) \right\} \qquad (3.8)$$

The maximization of eq. (3.8) does not change if we take the logarithm, multiply by two and three positive constants C_1, C_2 and C_3 which are independent of ρ. Denoting the logarithm of the first line, right hand side of eq. (3.8) by the metric value $M_{k-1}^{(\rho)}/2$ and substituting the logarithm of the product by the sum of logarithm yields

$$\max_{\rho} \left\{ M_{k-1}^{(\rho)} \right\} = \max_{\rho} \left\{ M_{k-1}^{(\rho)} + [2 \log P(x_{k,1}^{(\rho)}) - C_1] + [2 \log P(x_{k,2}^{(\rho)}) - C_2] \right.$$

$$\left. + \sum_{m=1}^{3} [2 \log p(y_{m,k}^{(\rho)} \mid x_{m,k}^{(\rho)}) - C_3] \right\} \qquad (3.9)$$

For simplicity, all constants are chosen as following

$$C_1 = \log P(x_{1,k} = -1) + \log P(x_{1,k} = 1) \qquad (3.10)$$

$$C_2 = \log P(x_{2,k} = -1) + \log P(x_{2,k} = 1) \qquad (3.11)$$

$$C_3 = \log p(y_{m,k} \mid x_{m,k} = -1) + \log p(y_{m,k} \mid x_{m,k} = 1) \qquad (3.12)$$

Therefore,

$$[\, 2 \log P(x_{1,k}^{(\rho)}) - C_1 \,] = \log \frac{P(x_{1,k}^{(\rho)}) \cdot P(x_{1,k}^{(\rho)})}{P(x_{1,k} = 1) \cdot P(x_{1,k} = -1)}$$

$$= \begin{cases} L(x_{1,k}) & \text{if } x_{1,k}^{(\rho)} = 1 \\ -L(x_{1,k}) & \text{if } x_{1,k}^{(\rho)} = -1 \end{cases}$$

$$= x_{1,k}^{(\rho)} . L(x_{1,k}) \tag{3.13}$$

Similarly, it can be shown

$$[\, 2 \log P(x_{2,k}^{(\rho)}) - C_2 \,] = x_{2,k}^{(\rho)} \cdot L(x_{2,k}) \tag{3.14}$$

$$\sum_{m=1}^{3} [2 \log p(y_{m,k}^{(\rho)} \mid x_{m,k}^{(\rho)}) - C_3 \,] = \sum_{m=1}^{3} x_{k,m}^{(\rho)} \cdot L_C \cdot y_{k,m}^{(\rho)} \tag{3.15}$$

The metric of the path ρ, $M_k^{(\rho)}$, can be recursively expressed as

$$M_k^{(\rho)} = M_{k-1}^{(\rho)} + x_{1,k}^{(\rho)} \cdot L(x_{2,k}) + x_{2,k}^{(\rho)} . L(x_{2,k}) + \sum_{m=1}^{3} x_{m,k}^{(\rho)} \cdot L_C \cdot y_{m,k}^{(\rho)} \tag{3.16}$$

It is worth noting that the probability of the path ρ at time k is related to the metric $M_k^{(\rho)}$ by

$$P(\text{path } \rho) = P(\boldsymbol{Y}_k, \boldsymbol{S}_k^{(\rho)}) = 10^{M_k^{(\rho)}/2} \tag{3.17}$$

Figure 3.8 illustrates a generic trellis for rate 2/3 convolutional codes. There are four possible trellis branches entering or leaving each trellis node. The VA proceeds with the computations of metrics for all possible paths using eq. (3.16). At time $t = k$, the VA has selected the ML-path with index $\rho_{0,1}$, i.e. the one with largest metric, and has discarded the other three paths with indexes $\rho_{0,2}$, $\rho_{0,3}$ and $\rho_{0,4}$. The probability that the selected path decision is correct at time $t = k$ is given by

$$P\,(\text{correct}) = \left[\frac{P\,(\text{path}\,\rho_{0,1})}{P\,(\text{path}\,\rho_{0,1}) + P\,(\text{path}\,\rho_{0,2}) + P\,(\text{path}\,\rho_{0,3}) + P\,(\text{path}\,\rho_{0,4})}\right]$$

$$= \left[\frac{10^{M_k^{(\rho_{0,1})}/2}}{10^{M_k^{(\rho_{0,1})}/2} + 10^{M_k^{(\rho_{0,2})}/2} + 10^{M_k^{(\rho_{0,3})}/2} + 10^{M_k^{(\rho_{0,4})}/2}}\right] \tag{3.18}$$

Using the notations

$$P(\text{correct} = 1) = P(\text{correct})$$

$$P(\text{correct} = -1) = P(\text{wrong})$$

$$= 1 - P(\text{correct})$$

$$\Delta_k^0 = \frac{1}{2}\left| M_k^{(\rho_{0,1})} - \max\left\{M_k^{(\rho_{0,2})},\ M_k^{(\rho_{0,3})},\ M_k^{(\rho_{0,4})}\right\}\right| \tag{3.19}$$

the log-likelihood ratio or soft value of the selected path decision is expressed as

$$\log\left[\frac{P(\text{correct})}{1 - P(\text{correct})}\right] = \log\left[\frac{10^{M_k^{(\rho_{0,1})}/2}}{10^{M_k^{(\rho_{0,2})}/2} + 10^{M_k^{(\rho_{0,3})}/2} + 10^{M_k^{(\rho_{0,4})}/2}}\right]$$

$$= (1/2)\,M_k^{(\rho_{0,1})} - \log\left[10^{M_k^{(\rho_{0,2})}/2} + 10^{M_k^{(\rho_{0,3})}/2} + 10^{M_k^{(\rho_{0,4})}/2}\right] \tag{3.20}$$

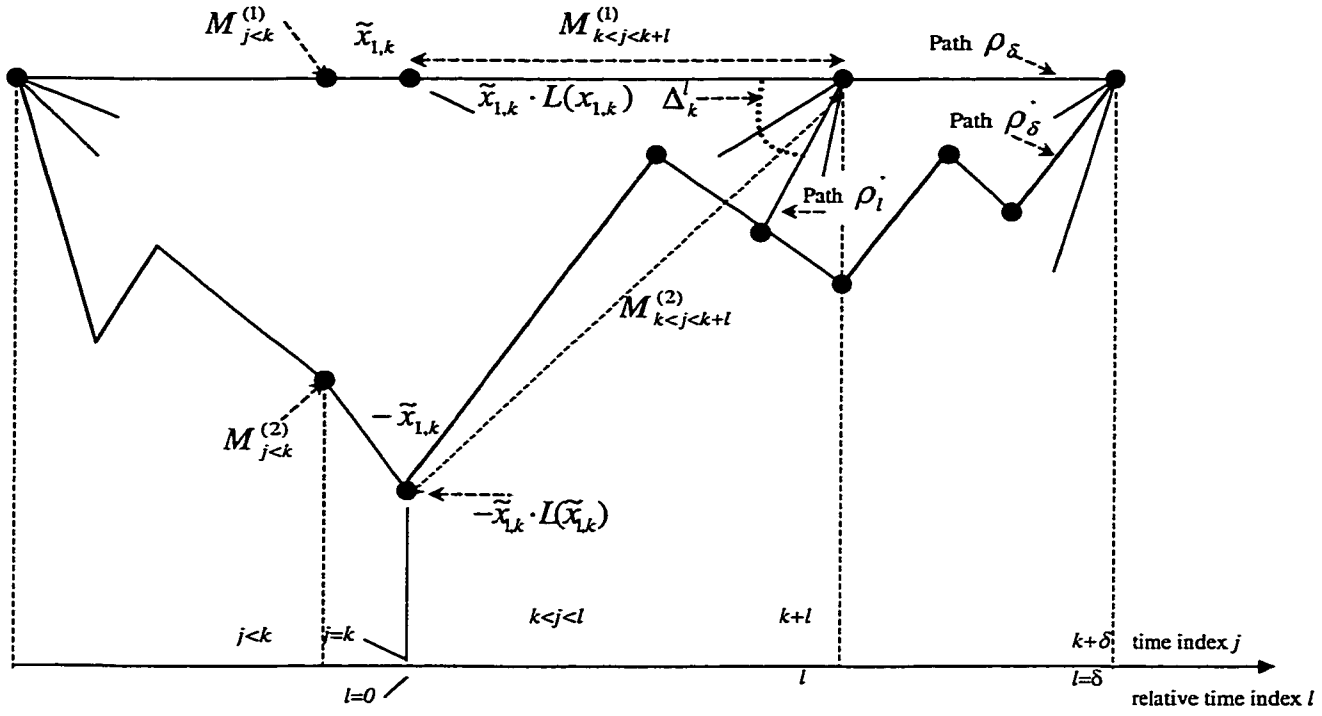


Figure 3.8: Generic trellis of rate 2/3 convolutional codes with constraint length K = 3 (For simplicity, the input $x_{2,k}$ is omitted from the figure.).

Using the approximation $(10^x + 10^y + 10^z) \approx \max(10^x, 10^y, 10^z)$, eq. (3.20) can be approximated as

$$\log\left[\frac{P(\text{correct})}{1-P(\text{correct})}\right] \approx (1/2)\left(M_k^{(\rho_{0,1})} - M_k^{(\rho_{max})}\right) \approx \Delta_k^0 \tag{3.21}$$

where $M_k^{(\rho_{max})} = \max\left\{M_k^{(\rho_{0,2})}, M_k^{(\rho_{0,3})}, M_k^{(\rho_{0,4})}\right\}$. Along the ML-path $\rho_{0,1}$, $3(\delta+1)$ non-surviving paths with indexes $l = (0,1,\ldots, 3\delta+2)$ have been discarded. The difference of their metric is $\Delta_k^l \geq 0$. In our case, the input pairs $(x_{1,k}, x_{2,k})$ of the rate 2/3 convolutional

codes are statistically independent due to the random interleaving action. As such their soft outputs can be computed independently from each other as followed. Suppose that if the bit $x^l_{1,k-\delta}$ on the discarded path is equal to $x_{1,k-\delta}$, then one has made no bit error by selecting the discarded path. The reliability of this bit decision would thus be infinity, ∞. But, if the two bits differ, the log-likelihood ratio of a bit error $e^l_{k-\delta}$ is equal to Δ^l_k, i.e.

$$\begin{aligned} L(e^l_{k-\delta}) &= \log\left[\frac{P(e^l_{k-\delta}=1)}{P(e^l_{k-\delta}=-1)}\right] \\ &= \begin{cases} \infty & x^l_{1,k-\delta} = x_{1,k-\delta} \\ \Delta^l_k & x^l_{1,k-\delta} \neq x_{1,k-\delta} \end{cases} \end{aligned} \tag{3.22}$$

Taking into account all $3(\delta+1)$ discarded paths, the sum of all $L(e^l_{k-\delta})$'s for the bit $x_{1,k-\delta}$ is approximated by

$$\begin{aligned} \left[\sum_{l=0}^{3\delta+2}\langle+\rangle\ L(e^l_{k-\delta})\right] &= \left[\sum_{l=0}^{3\delta+2}\langle+\rangle\ \Delta^l_k\right] \\ &\approx \min_{l=0,\ldots,3\delta+2} \Delta^l_k \end{aligned} \tag{3.23}$$

where $\sum\langle+\rangle$ is defined as logarithmic summation (see eq. A.27 of appendix A). It follows that the LLR of the bit decision, or the soft-output of the VA (SOVA), is the hard decision $x_{1,k-\delta}$ multiplies with the LLR of the total errors, i.e.

$$L(x_{1,k-\delta}) \approx x_{1,k-\delta} \cdot \min_{l=0,\ldots,3\delta+2} \Delta^l_k \tag{3.24}$$

Note that in eq. (3.24) the minimum should only be taken over the indexes l where the bits $x_{1,k-\delta}$ and $x^l_{1,k-\delta}$ differ. Continuing with the same analysis, it can be shown that

$$L(x_{2,k-\delta}) \approx x_{2,k-\delta} \cdot \min_{l=0,\ldots,3\delta+2} \Delta_k^l \tag{3.25}$$

From eq. (3.21), Δ_k^l is given by

$$\Delta_k^l \approx (1/2)\left(M_{k+l}^{(\rho_{0,1})} - M_{k+l}^{(\rho_{max})}\right)$$

Replacing the metric terms by eq. (3.16), Δ_k^l becomes

$$\begin{aligned}\Delta_k^l \approx\ & (1/2)\cdot\left[\left(M_{j<k}^{\rho_{0,1}} - M_{j<k}^{\rho_{max}}\right) + \left(M_{k<j<k+l}^{\rho_{0,1}} - M_{k<j<k+l}^{\rho_{max}}\right)\right] \\ & + (1/2)\cdot\left[L_C \cdot y_{1,k} \cdot [\tilde{x}_{1,k} - (-\tilde{x}_{1,k})] + \sum_{m=2}^{3} L_C \cdot y_{m,k} \cdot (x_{m,k}^{(\rho_{0,1})} - x_{m,k}^{(\rho_{max})})\right] \\ & + (1/2)\cdot\left[L(x_{1,k}) \cdot [\tilde{x}_{1,k} - (-\tilde{x}_{1,k})] + L(x_{2,k}) \cdot [\tilde{x}_{2,k} - (-\tilde{x}_{2,k})]\right]\end{aligned} \tag{3.26}$$

The minimum of $\{\Delta_k^l, l = 1, 2, \ldots, 3\delta+2\}$ would also have the same structure as eq. (3.26). Substituting eq. (3.26) in eq. (3.24) and (3.25) yields

$$L_{SOVA}(\tilde{x}_{1,k}) \approx L_C \cdot y_{1,k} + L(x_{1,k}) + \text{other terms} \tag{3.27}$$

$$L_{SOVA}(\tilde{x}_{2,k}) \approx L_C \cdot y_{2,k} + L(x_{2,k}) + \text{other terms} \tag{3.28}$$

The "other terms" represent the extrinsic information. Finally, the above two soft-output Viterbi Algorithm equations for rate 2/3 convolutional codes have similar structure as that in eq. (3.2). As such, they can be rewritten in term of their respective extrinsic information as

$$L_{SOVA}(\tilde{x}_{1,k}) \approx L_C \cdot y_{k,1} + L(x_{1,k}) + L_E(\tilde{x}_{1,k}) \tag{3.29}$$

$$L_{SOVA}(\tilde{x}_{2,k}) \approx L_C \cdot y_{k,2} + L(x_{2,k}) + L_E(\tilde{x}_{2,k}) \tag{3.30}$$

With the proposed modified SOVA just derived, the BER performance of the modified rate 1/4 TC with joint parity-over-parity generated by the encoder shown in figure 3.7 is simulated. For this particular simulation, following parameters were used: random interleaver of length $N = 900$, rate 1/2 RSC encoder with $G = [1, 1+D^2/1+D+D^2]$ for ENC 1 and 2, and rate 2/3 RSC encoder with G' given by

$$G' = \begin{bmatrix} 1 & 0 & \dfrac{1+D^2}{1+D+D^2} \\ 0 & 1 & \dfrac{D}{1+D+D^2} \end{bmatrix} \tag{3.31}$$

for ENC3.

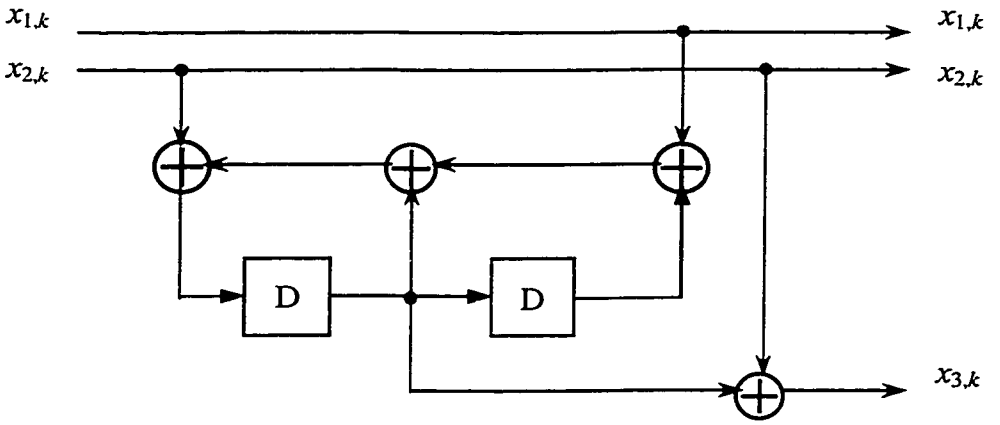


Figure 3.9: Rate 2/3 RSC encoder with G' (K=3).

Figure 3.10 depicts the simulation results of the conventional rate 1/4 TC and that of the modified rate 1/4 TC at the fifth decoding iteration. As anticipated, figure 3.10 displays similar characteristics to those of the rate 1/5 TC (refer to figure 3.6) in the sense that the modified rate 1/4 TC slightly outperforms the conventional rate 1/4 TC at very low SNR but significantly underperforms the conventional one at moderate to high SNR. Two interesting observations worth mentioning about figure 3.10 compared to figure 3.6. By going from rate 1/5 to rate 1/4, the value of SNR below which the modified TC outperforms the conventional TC reduces from roughly 1.75 dB to 0.75 dB 1/5 TC. The performance difference between the modified and the conventional TCs also narrows down from roughly 1.5 to 1.00 dB at the fifth decoding iteration and BER ~ 5.00×10^{-5}. The positive or desirable aspect of the modified rate 1/4 TC compared to the conventional rate 1/4 TC is its decoding computations, which is approximately 4/5 of the conventional rate 1/4 TC's for five decoding iterations.

The possible reasons behind the performance difference between the modified and conventional rate 1/4 TCs can be explained as in the case of rate 1/5. First, the received systematic and parity sequences of the modified rate 1/4 and those of the conventional rate 1/4 are equally reliable since they are all transmitted with equal energy through the same noisy channel. However, the estimated parity sequences $\tilde{\boldsymbol{x}}_1$ and $\tilde{\boldsymbol{x}}_2$ of the modified rate 1/4, which are used to iteratively decode the received systematic sequence $\boldsymbol{u}$ are likely more reliable than the received parity sequences of the conventional rate 1/4, hence favoring the performance of the modified rate 1/4 at SNR < 0.50 dB. At larger SNR, $\tilde{\boldsymbol{x}}_1$ and $\tilde{\boldsymbol{x}}_2$ are also more reliable than the received parity sequences of the conventional rate 1/4, yet the modified rate 1/4 performs poorly compare to the conventional rate 1/4 because the reliability gain in the estimated parity sequences of the modified rate 1/4 is

not enough to compensate the lost or reduction of the diversity provided by the parity sequences. Recall that in the modified rate 1/4, the received systematic sequence is iteratively decoded with only two (estimated) parity sequences, whereas in the conventional one, the same received systematic sequence is iteratively decoded with three parity sequences.

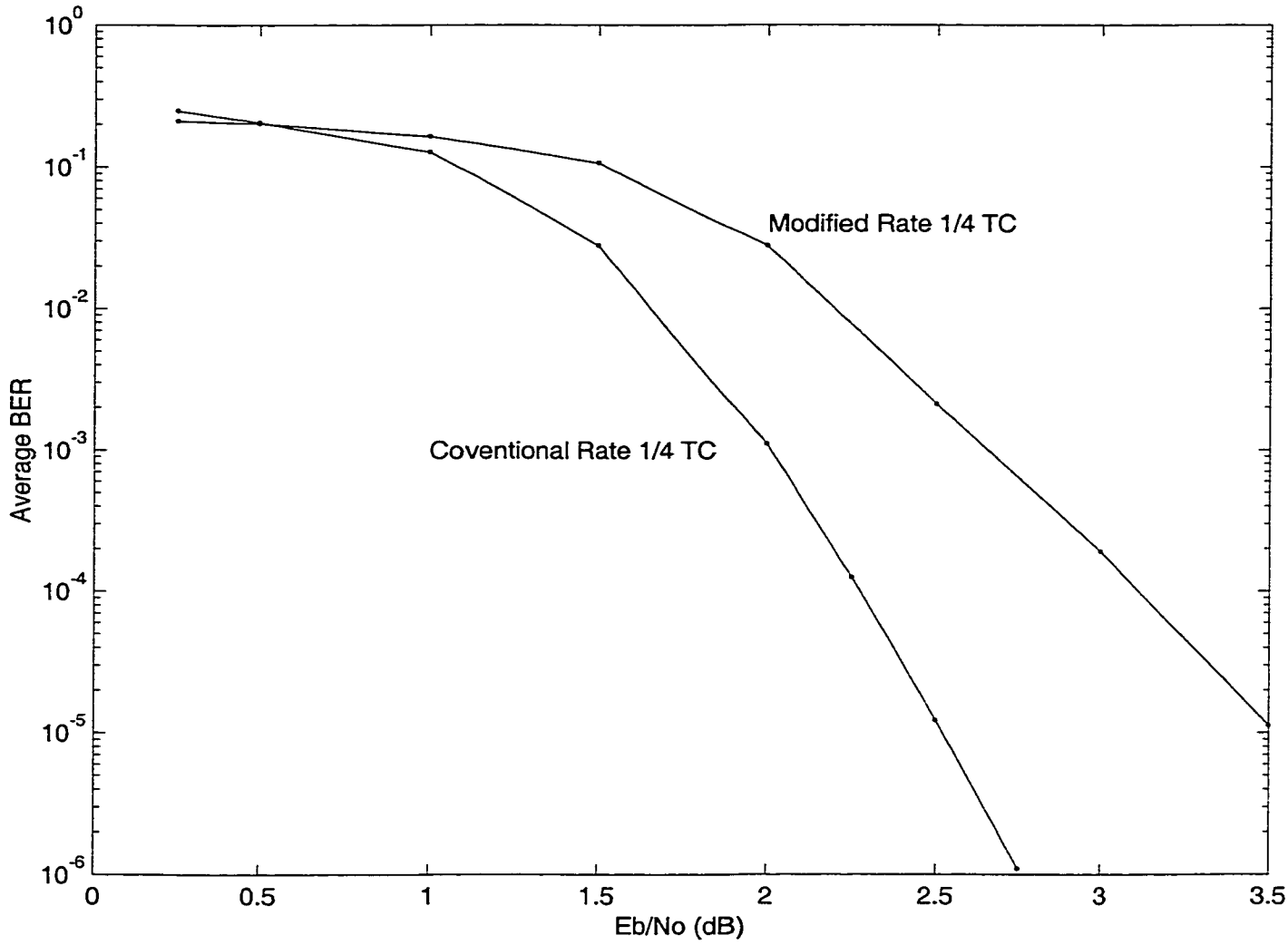


Figure 3.10: BER of the conventional rate 1/4 and the modified rate 1/4 TC at the fifth decoding iteration using random interleaver (N=900).

3.4 Suggested Research Topic

3.4.1 Motivation

So far, the TC study in this thesis has found out that for a given choice of constituent codes and interleaver, the unpunctured rate 1/3 TC yields the best BER performance at SNR < 2.50dB. Furthermore, it was shown that the performance of such TC can be increased through the enhancement of the corresponding parity sequences. For the same reliability values of the systematic sequence and the same parity diversity, the larger the reliability values of the corresponding parity sequences, the lower the decoding BER. Two different approaches for improving or enhancing the reliability values of the parity sequences of the conventional TC were proposed and their effects on the associated decoding BER's were also studied.

Simulation results indicated that increasing the transmitted power is the most efficient and simplest way for improving the reliability values of the parity sequences and the overall decoding BER. This approach has the flexibility to increase the reliability values of the individual parity bits without reducing the overall code rate or the parity diversity. The weakness of this approach is that it requires some increase of the average transmitted power in order to yield noticeable BER improvement. An alternative approach to enhance the reliability of the parity sequences is to add redundancy to the parity bits, i.e. parity-over-parity.

There are two undesirable effects associated with the parity-over-parity scheme, namely the reduction of the overall code rate or/and parity diversity. For example, the introduction of the extra parity-over-parity bits to the conventional rate 1/3 TC changes the overall code rate from 1/3 to 1/5, hence reducing the available transmitted power per coded bit which makes the systematic sequence of the rate 1/5 parity-over-parity less reliable than that of the rate 1/3 TC. Comparing the rate 1/5 TC with parity-over-parity to the conventional rate 1/5 TC, it is clear that their systematic sequences are equally

reliable. However, the parity diversity of the former is half as much as that of the latter. For the rate 1/5 TC with parity-over-parity, only two parity sequences are available for iterative decoding of the systematic sequence, while the conventional rate 1/5 TC has four parity sequence available for iterative decoding of the systematic sequence. Simulations showed that the parity-over-parity approach tends to degrade rather than improve the BER performance. Such performance degradation is mainly due to fact that the reliability gain in the parity sequences is not enough to compensate the reliability lost in the systematic sequence or the lost of the parity diversity.

It appears from the simulation results in this chapter that in order to noticeably improve the BER performance of the conventional TC, there must be an increase in the reliability of the received parity sequences without reducing either the reliability of the received systematic sequence and the parity diversity. Based on these findings, a new class of concatenated codes with enhanced parity-over-parity is suggested as a research topic. Such codes would have not only the same code rate, hence same transmitted power per coded bit, as the conventional unpunctured TC, but also the same parity diversity. The characteristics of these codes are conceptually outlined and discussed in this section. For convenience, the discussion begins with the rate 1/3, and extends to the general rate $1/q$ where appropriate.

3.4.2 Encoding/Decoding Characteristics

Figure 3.11 and 3.12 depict the encoding relationships among the systematic and the two parity sequences of the conventional rate 1/3 TC, and those of the so-called rate 1/3 concatenated codes with enhanced parity-over-parity. As can be seen from figure 3.11, the systematic bits of the conventional rate 1/3 TC are independently protected against the channel noise by the parity bits, while the parity bits themselves have no error protection. Parity-over-parity bits can be inserted to provide noise protection to the parity bits. However in doing so, there is a reduction in the available power per coded bits. So far, the study of this chapter has demonstrated that the insertion of the parity-over-parity

bits, generated from RSC encoders, has not achieved enough gain in the reliability of the parity bits to compensate the reduction of the transmitted power per coded bit. It is worth noting that the parity-over-parity encoding technique studied in this chapter specifically makes use of only RSC encoders to simplify the overall decoder structure. One can use encoders other than RSC encoders to encode the parity bits of the conventional rate 1/3 TC. For instance, one may encode them with a Hyper Code encoder. It should be noted though that the Hyper Codes with iterative coding do not have good coding gain at low SNR (≤2.2dB) [40]. As such, it is anticipated that coding the parity bits of the conventional rate 1/3 TC with Hyper Codes encoder will not improve the BER performance of the conventional rate 1/3 TC at low SNR.

Figure 3.12 conceptually illustrates the principle encoding characteristics of the so-called rate 1/3 concatenated codes with enhanced parity-over-parity. These codes would have the same code rate (R = 1/3) as the conventional rate 1/3 TC, hence there is no reduction of the transmitted power per coded bits. An advantage of these codes over the conventional rate 1/3 Turbo Codes is that each sequence, systematic or parity, is of equal importance and is error protected by the other two sequences. This is not the case for the conventional rate 1/3 Turbo Codes where the two parity sequences are treated as uncorrelated or independent. Because the three sequences of the rate 1/3 concatenated codes with enhanced parity-over-parity are treated as dependent, an observation or reception of one sequence can provide some information on the other two and vice versa. As such, it is possible to decode each sequence alternatively and iteratively with the other two. This feature provides a mean to iteratively correct some bit errors in the received parity sequences, and obtain more reliable estimated parity sequences which in turn can be used to yield a better estimation for the systematic sequence.

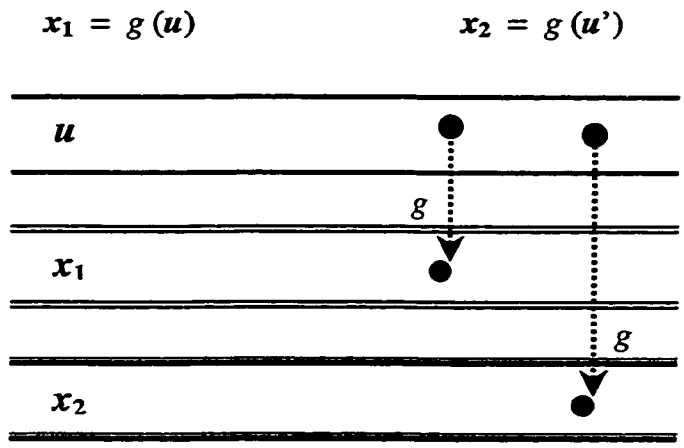


Figure 3.11: The encoding relationships among the systematic and the two parity sequences of the conventional rate 1/3 TC.

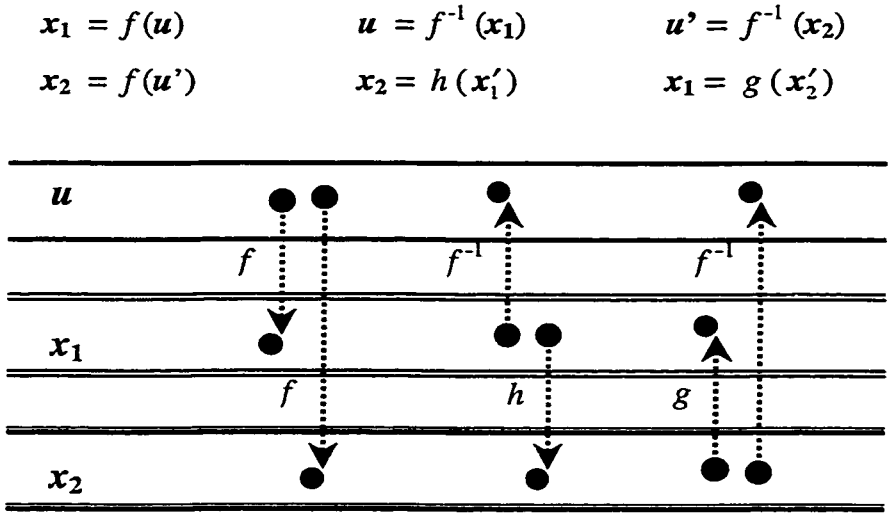


Figure 3.12: The encoding relationships among the systematic and the two parity sequences of the rate 1/3 Concatenated Codes with enhanced parity-over-parity, where f^{-1} and g^{-1} are the inverse of f and g.

The rate 1/3 concatenated codes with enhanced parity-over-parity can be extended to a general rate $1/q$, where $q \geq 3$. The conventional rate $1/q$ TC and the rate $1/q$ enhanced parity-over-parity would have the same parity diversity. Their systematic sequences would be error protected by $(q\text{-}1)$ parity sequences. The main difference between these two codes is that there is an effort to exploit the correlation between the $(q - 1)$ parity sequences, while in the conventional rate $1/q$ TC, the $(q - 1)$ parity sequences are simply treated as uncorrelated.

The decoding of the rate $1/q$ concatenated codes with enhanced parity-over-parity is similar to the decoding of the conventional rate $1/q$ Turbo Codes. It is however carried out in two sequential steps. The first step consists of decoding each parity sequence alternatively and iteratively with the systematic and other parity sequences. The second step involves decoding the systematic sequence alternatively and iteratively with the $(q\text{-}1)$ estimated or decoded parity sequences obtained during the first step. Because of the iterative decoding strength, the $(q\text{-}1)$ estimated or decoded parity sequences would be much more reliable than the received parity sequences, hence resulting a low BER for the estimated systematic sequence. It should be mentioned that every sequence of the concatenated codes with enhanced parity-over-parity, not just the systematic one, is iteratively decoded. As such, the decoding of the rate $1/q$ concatenated codes with enhanced parity-over-parity will require at least q times more computations than decoding the conventional rate $1/q$ TC.

3.4.3 The Challenge

The essential characteristics of the concatenated codes with enhanced parity-over-parity have conceptually been identified. The challenge is to search for or develop a code which possesses such characteristics. There would probably be a number of different developmental techniques for obtaining the so-called concatenated codes with enhanced parity-over-parity. The following describes a technique we have attempted and its inherent difficulty.

We first consider that the conventional rate $1/q$ TC represents a special case or simplified version of the so-called rate $1/q$ concatenated codes with enhanced parity-over-parity. In fact, the former can be viewed as a subset of the latter since the latter possesses all the characteristics of the former. With this in mind, our approach to the development of the rate $1/q$ concatenated codes with enhanced parity-over-parity is to exploit the existing characteristics of the conventional TC, and to search for additional functions which establish correlations among the parity sequences as described below.

For simplicity, consider the conventional rate 1/3 TC and the rate 1/3 concatenated codes of figure 3.11 and 3.12 respectively. It is observed that each codeword of the conventional rate 1/3 TC contains two elementary rate 1/2 RSC codewords namely $(\boldsymbol{u}\boldsymbol{x}_1)$ and $(\boldsymbol{u}'\boldsymbol{x}_2)$, where $\boldsymbol{u}'$ is the interleaved version of $\boldsymbol{u}$. On the other hand, each codeword of the so-called concatenated codes with enhanced parity-over-parity is made up of six different elementary rate 1/2 systematic codewords, four of which are rate 1/2 RSC codewords. Two of these four rate 1/2 RSC codewords belong to the conventional rate 1/3 TC. The six elementary codewords of the enhanced parity-over-parity concatenated codes are $\{(\boldsymbol{u}\boldsymbol{x}_1), (\boldsymbol{u}'\boldsymbol{x}_2), (\boldsymbol{x}_1\boldsymbol{u}), (\boldsymbol{x}_1'\boldsymbol{x}_2), (\boldsymbol{x}_2\boldsymbol{u}')$ and $(\boldsymbol{x}_2'\boldsymbol{x}_1)\}$.

Four of the six elementary codewords, namely $\{(\boldsymbol{u}\boldsymbol{x}_1), (\boldsymbol{x}_1\boldsymbol{u}), (\boldsymbol{u}'\boldsymbol{x}_2)$ and $(\boldsymbol{x}_2\boldsymbol{u}')\}$, are obtainable directly from the conventional rate 1/3 TC because of the existence of the inverse generator function or matrix for the RSC encoder. For example, if $(\boldsymbol{u}\boldsymbol{x}_1)$ is a rate 1/2 RSC codeword generated from a generator matrix $f = [1, 1 + D^2/1 + D + D^2]$, then $(\boldsymbol{x}_1\boldsymbol{u})$ is also a rate 1/2 RSC codeword produced from the inverse generator matrix f^{-1}, where $f^{-1} = [1, 1 + D + D^2/1 + D^2]$. Consequently, the challenge in the development of the rate 1/3 enhanced parity-over-parity concatenated codes is reduced to finding the generator functions or matrices for the remaining two elementary rate 1/2 systematic codewords, namely $(\boldsymbol{x}_1'\boldsymbol{x}_2)$ and $(\boldsymbol{x}_2'\boldsymbol{x}_1)$.

So far we have not yet been able to overcome the above challenge, i.e. obtaining single-level encoding expressions or functions for both $(x_1' x_2)$ and $(x_2' x_1)$, which would allow us to derive the required decoding algorithms. The meaning of single-level function will be relevant later. The following illustrates the degree of complexity or difficulty for obtaining the single-level encoding functions for $(x_1' x_2)$ and $(x_2' x_1)$. We would like to express x_1 as a single-level function of x_2 or vice versa, i.e.

$$x_1 = g(x_2') \tag{3.32}$$

$$x_2 = h(x_1') \tag{3.33}$$

where x_1' and x_2' denote the interleaved versions of x_1 and x_2 respectively. It is worth mentioning that h is not an inverse of g, i.e. $h \neq g^{-1}$. From the conventional rate 1/3 TC (see figure 3.11), we have the following relationship

$$x_1 = f(u) \tag{3.34}$$

$$u' = f^{-1}(x_2) \tag{3.35}$$

where f is a known parity generator function of a given rate 1/2 RSC, and is f^{-1} the inverse of f. Let $I\{x\}$ and $D\{x\}$ denote the interleaving and de-interleaving of x respectively, i.e. $I\{x\} = x'$ and $D\{x'\} = x$. Substituting eq. (3.35) in eq. (3.34) yields:

$$x_1 = f(D\{f^{-1}(D\{x_2'\})\}) \tag{3.36}$$

Eqs. (3.32) and (3.36) are equivalent in the sense that x_1 is expressed in term of x_2 only. Furthermore, they state that the observation x_2 of can provide information about x_1, a characteristic that was not exploited in the conventional TC. From the encoding standpoint, eq. (3.36) is easily realizable. Figure 3.14 illustrates a direct realization of eq. (3.36). However, from the decoding standpoint, both eq. (3.36) and the associated encoder of figure 3.13 are not practical. It should be noted that the main objective here is

to decode the rate 1/2 systematic codeword $(x_2' x_1)$ alone independent of any other sequence, but there is no known soft decoding algorithm which allows us to decode the codeword $(x_2' x_1)$ based on eq. (3.36) or the encoder of figure 3.13. An alternative solution is to reduce the multi-level function of eq. (3.36) to a single-level function of eq. (3.32), where soft decoding could be carried out in a single-step using existing or modified algorithm. Figure 3.14 illustrates a single stage encoder which is a direct realization of eq. (3.2). Both encoders of figures 3.13 and 3.14 are equivalent in the sense that they generate the same rate 1/2 systematic codeword $(x_2' x_1)$. However, the soft decoding operations of $(x_2' x_1)$ based on the encoder figure 3.14 would be much simpler than those based on figure 3.13.

The challenge is then reduced to finding the parity generator function, g( ), of eq. (3.32). This is quite a complex problem that we have so far been unable to resolve, given the recursive nature of $f(\ )$ and the large size of $D\{\ \}$.

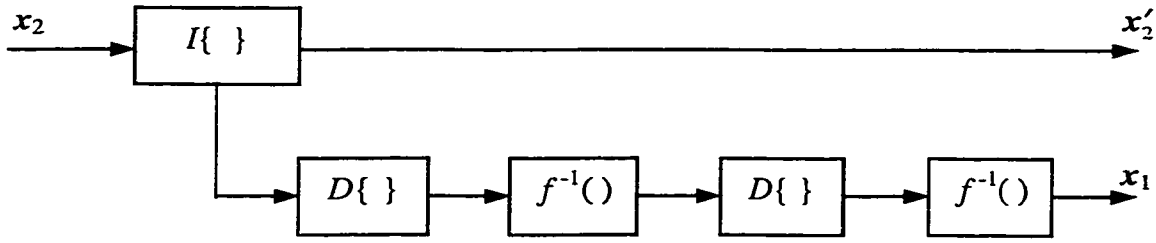


Figure 3.13: Multi-stage rate 1/2 systematic encoder.

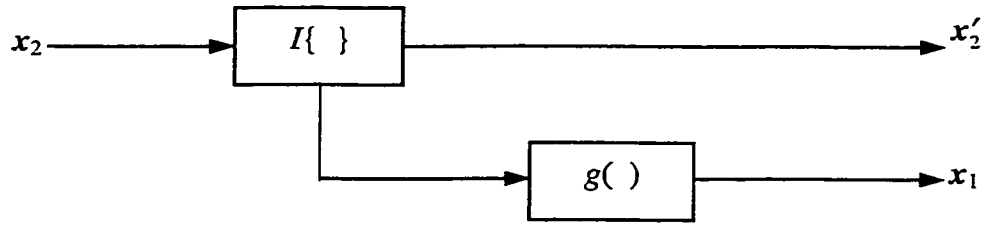


Figure 3.14: Single-stage rate 1/2 systematic encoder.

3.5. Conclusion

This chapter attempted to demonstrate that the BER performance of the current Turbo Codes can be improved by increasing the noise immunity of the parity bits alone. It introduced several modifications to the current Turbo Codes for reducing the number of error bits in the received parity sequences, and studied the effect of such modifications on the overall BER performance. Specifically, three distinct modifications to the current Turbo Codes were studied.

The first modification, referred to as unequal energy partitioning, consisted of raising the transmitted energy for each parity bit while simultaneously reducing the transmitted energy for the corresponding systematic bit in order to keep the average transmitted energy unchanged. Significant increase of the transmitted energy for the individual parity bits could result in large reduction of the transmitted energy for the corresponding systematic bits, and lead to the overall BER performance degradation. Simulation results showed that unequal energy partitioning could improve the BER performance at E_B/N_O >1.75 dB, and normalized $(E_{SS}, E_{SP}) = (1/4, 3/8)E_S$ appears to be the optimal pair.

The second modification attempted to protect the parity bits against the channel noise by applying a parity-over-parity encoding scheme whereby each parity sequence is encoded independently. The BER performance of the modified rate 1/5 Turbo Codes with parity-over parity encoding were simulated and compared with that of the conventional rate 1/5 Turbo Codes. Simulation results showed that the two codes had comparable BER performances at $E_B/N_O \leq 1.75$ dB. However, the BER performance of modified rate 1/5 was significantly inferior to that of the conventional rate 1/5 at E_B/N_O >1.75 dB.

The third modification is similar to the second one in the sense that it provided error protection to the parity bits through parity-over-parity encoding. The difference is that under this scheme, two parity sequences are jointly rather than independently encoded. A

modified SOVA algorithm was developed to decode the two jointly encoded parity sequences. The BER performance of the modified rate 1/4 Turbo Codes with joint parity-over parity encoding were simulated and compared with that of the conventional rate 1/4 Turbo Codes. Simulation results showed that the two codes had comparable BER performances at $E_B/N_O \leq 0.75$ dB. However, the modified rate 1/4 Turbo Codes clearly underperformed the conventional rate 1/4 Turbo Codes at $E_B/N_O > 0.75$ dB.

The study of this chapter clearly shows that increasing the reliabilities of the parity bits results in better BER performance. However, it also shows that the reliabilities of the parity bits are not the only variable affecting the BER performance. There are other important variables, namely the reliabilities of the systematic bits and the parity diversity. These three variables are not independent and changing the value of one variable can affect the values of others. As such, increasing the reliabilities of the received parity sequences alone can improve the overall BER performance if the overall reliability gain in the parity bits is more than enough to compensate the lost of the reliability lost in the systematic bits or the lost of the parity diversity.

Finally, the study in this chapter suggests that parity-over-parity encoding is not an effective way to increase the reliabilities of the Turbo Codes parity bits because it reduces either the reliabilities of the corresponding systematic bits or the parity diversity, causing serious degradation of the BER performance. These findings were used to lay down some ground works for the possible development of Turbo Codes with enhanced parity-over-parity as described in section 3.4.

Chapter 4

CONCLUSION

4.1 Thesis Summary

This thesis conducted an extensive study on the conventional low rate Turbo Codes (TC) and some of its potential variants in the region of low signal to noise ratio. The objectives or goals of the thesis are threefold. The first goal is to study the BER performances of the conventional low rate TC as a function of some independent parameters, and to determine the optimum TC code rate which yields minimum BER in the region of low SNR. The second goal is to highlight a possible area of improvement to the current TC, and to examine several modifications referred to as TC variants. Finally, the last goal is to make use of the findings from the study of the conventional low rate TC and TC variants to formulate the characteristics of new concatenated codes which could be viewed as a superset of the current TC, and which could achieve lower BER.

The thesis was mainly organized in two parts. The first part began with a comprehensive analysis of the current TC as introduced by Berrou, Glavieux and Thitimajshima. The analysis included the evaluation of the TC BER probability expression, and the discussion of independent and influential parameters affecting the BER probability, specifically the interleaver length, the polynomial generator of the constituent codes and the TC code rate. An important parameter – namely the number of decoding iterations – was not included in the BER probability equation. However, its impact or behavior on the BER performance at low SNR, ~ [0.25 – 2.50 dB], was studied through numerous simulations in conjunction with other parameters. The following observations were made during the simulations:

① *In general, the average BER decreases as the number of decoding iterations increases. However, it seems that most of the BER reduction is achieved within the first five or six iterations. Furthermore, there appears to be an error floor for each value of SNR. After the average BER has reached the error floor, additional decoding iteration would yield negligible, or no, BER improvement;*

② *The higher the SNR, the larger the BER reduction can be achieved by each additional decoding iteration before it reaches the error floor;*

③ *At SNR $\approx$ 0.50 dB or less, the average BER of the rate 1/3 TC does not seem to decrease as a function of the number of decoding iterations. For general rate 1/q TC, the value of SNR below which the average BER does not depend significantly on the number of decoding iterations increases as a function of q;*

④ *Iterative decoding is effective if and only if it is accompanied by interleaving. In that sense, the most important role of the interleaving in Turbo Codes is to decorrelate the extrinsic information output by one decoder and the observed inputs of the other decoders. The less important role of interleaving is to increase the corresponding TC effective free distance;*

⑤ *Unlike the BER performances of other existing codes, the BER performance of Turbo Codes with iterative decoding is not dominated by the minimum or free distance. Nor it is dominated by the small number of codewords with minimum or free distance. It is the effect of interleaving which is the dominant factor responsible for the outstanding TC BER performance.*

⑥ *In general, the larger the interleaver length the more effective the iterative decoding, i.e. the smaller the associated BER. The BER improvement is however achieved at the increase of the decoding delay;*

⑦ *For a given interleaver of length N, the average BER decreases as the constraint length K of the constituent codes increases. For a random interleaver of length N = 900, increasing K yields negligible or no BER improvement at SNR ≈ 1.00 dB or less;*

⑧ *At SNR ~ [0.50 - 2.50 dB], optimizing the interleaver length appears to yield better BER improvement than optimizing the constraint length of the constituent codes. At SNR < 0.50 dB, the choices of these two parameters seem to have negligible effect on the BER;*

⑨ *At SNR ~ [0.25 - 2.50 dB], TC with code rate 1/3 yields the best performance. For the generic rate 1/q TC where q > 3, the larger the q value, the larger the SNR value below which the rate 1/q underperforms the rate 1/3 TC. Also, at low SNR, the reliability of systematic bits is more important than that of the parity bits. At moderate to high SNR, the trend is reversed; and*

⑩ *Puncturing has different effect on the TC BER performances. For the rate 1/3 TC, puncturing degrades the performance across the entire range of SNR. For the generic rate 1/q where q > 3, puncturing improves the performance at low SNR but has reversed effect at moderate to large SNR.*

The second part of the thesis highlighted a possible area of improvement to the current TC and predicted that the BER performance of the current TC, specifically the rate 1/3 TC, could be improved by increasing the reliabilities of the corresponding parity sequences alone. With that view in mind, two different approaches for improving the reliabilities of the parity sequences were proposed and investigated. Three variations or modifications to the current TC were introduced, and their performances were simulated and compared to the current TC's. The four TC variants are:

① *Modified TC with unequal energy partitioning;*
② *Modified TC with full parity-over-parity; and*
③ *Modified TC with joint parity-over-parity.*

The first TC variant attempted to increase the noise immunity of the corresponding parity bits by raising the corresponding transmitted energy without increasing the overall average transmitted energy. Under this variant, systematic bits are considered as less important than the parity bits . As such, they are transmitted with less energy than the parity bits. Simulation showed that for a random interleaver of length N = 500, the unequal energy partitioning start to outperform the original TC at SNR roughly greater than 1.75 dB. Furthermore, it is observed that the benefit of the unequal energy partitioning becomes increasingly visible as the number of decoding iteration increases.

The last two TC variants, referred to as parity-over-parity, attempted to enhance the noise immunity of the parity bits by adding additional or extra parity bits to the exiting parity bits. Specifically, the third TC variant independently encoded the two parity components of the conventional rate 1/3 TC to produce a modified rate 1/5 TC. Simulation showed that the BER performance of the modified rate 1/5 TC is slightly better than that of the conventional rate 1/5 TC at SNR < 1.75 dB, but worse than the conventional rate 1/5 TC at SNR ≥ 1.75 dB.

The third TC variant attempted to enhance the noise immunity of the parity bits by jointly encoding the two parity components of the conventional rate 1/3 TC with a rate 2/3 RSC encoder to produce a modified rate 1/4 TC. Again, simulation using a modified SOVA showed that the BER performance of the modified rate 1/4 TC is slightly better than that of the conventional rate 1/4 TC at SNR < 0.50 dB, but worse than the conventional rate 1/4 TC at larger value of SNR. The fact that the last two TC variants underperform the conventional TC of the same code rates does not mean that the same principle applied in the last two TC variant contradicts the one applied in the first TC variants, or that the principle is flaw. The poor performances of the modified rate 1/4 and 1/5 TC' s can be attributed to the lost of diversity in the modified rate 1/4 and 1/5 TC' s. It is interesting to observe that the modified rate 1/4 and 1/5 TC' s also underperform the conventional rate 1/3 TC although they all have the same diversity. This is because their systematic bits are more vulnerable to the channel noise than those of the conventional rate 1/3 TC. An important conclusion drawn from the last two TC variant is the following: "Improving the noise immunity of the parity bits yield better a BER performance for the resulted Turbo Coded if and only if the reliability gain of the parity bits is larger than the lost of the parity diversity or/and the lost of the noise immunity of the systematic bits".

It is worth mentioning that only RSC encoders were used to encode the parity bits of the modified rate 1/4 and 1/5 TC' s. More powerful encoder such as Hyper codes encoder can also be used. However, it is expected that the results would be the same in the low region of SNR, i.e. ~ [0.25 - 2.50 dB], since the Hyper codes have high BER at this range of SNR.

4.2 Thesis Contributions

This thesis attempts to make contributions to the field of error control coding by conducting detailed study on the behaviors of the low rate Turbo Codes whose constituent codes are rate 1/2 RSC, simulating and reporting different impacts of critical parameters on the BER performance. Furthermore, it highlights a possible area of improvement to the original TC and investigates the BER performances of some TC variants. Finally, the thesis attempt to establish a framework for the search or future development of a new class of concatenated codes which can be viewed as a superset of the current TC. Specifically, the thesis contributions include :

- Simulating the optimum TC code rate at low SNR;
- Investigating the impact of puncturing on the low rate TC;
- Pointing out an area of improvement to the current TC;
- Studying the effect of unequal energy partitioning on the BER performance;
- Studying the effect of parity-over-parity encoding/decoding on the BER performance;
- Evaluating mathematical expressions for implementing soft decision decoding (SOVA) on the rate 2/3 recursive systematic convolutional codes; and
- Defining the essential characteristics for a new class of concatenated codes, and attempting to develop such codes by exploiting the properties of the current TC.

4.3 Suggestion for further research

The simulation results reported in chapter 3 clearly indicate that improving the reliability values of the parity sequences alone could lead to better BER performance. However, the improvement of the parity reliability should not be achieved at the expense of parity diversity or/and code rate. One way to accomplish this is to make use of the correlations among the parity sequences, which are not considered under the current Turbo Codes. This view leads to the postulation of a new class of concatenated codes whose encoding structures make it possible to decode not only the systematic sequence but also the parity sequences. The characteristics of this new class of concatenated codes were pictorially highlighted.

Note that Turbo Code is a special case of this new class of concatenated codes in the sense that each parity sequence can be expressed as a function of the other. As such, one way to improve the BER performance over the entire SNR range is to develop a more efficient decoding algorithm which allows to exploit the correlation among the parity sequences. The complexity or challenge of this task was discussed in section 3.4. An alternative is to search for a more sophisticated encoding technique which produces codes with characteristics shown in figure 3.12, but have less complex decoding algorithm than the one discussed in section 3.4. The search or development of such encoding technique is left as a suggestion for further research.

Appendix A

Concatenated Codes and Iterative Decoding

This annex provides a comprehensive review of concatenated codes with main emphasis on serial concatenated codes, and an introduction to iterative decoding. The annex is essentially organized in four small sections. The first section describes different groups of concatenated codes in the generic framework. The second section provides a coverage on serial concatenated codes mainly from the encoding point of view. Generic parallel concatenated codes are briefly exposed in the third section, while the detail of encoding, decoding and performance of specific parallel concatenated codes such as Turbo Codes are covered in chapter 1. Finally, the last section discusses the essential aspects of iterative decoding.

A.1 GENERIC CONCATENATED CODES

The main objective in the field of coding theory primarily involves searching for codes with efficient structure or properties, which can achieve a performance close to the limit defined by Shannon, while allowing for encoding and decoding with equipment of moderate complexity. It was well known in the early days of Shannon theory that the larger the codes the better the performance. Unfortunately, the complexity of existing decoding algorithms can rapidly increase with the code size up to a level where decoding becomes physically unrealizable. To overcome the decoding difficulty, many research works have been undertaken with the aim of constructing powerful codes which have performance equivalent to that of a large code, and a decomposable structure which allows to replace the complex maximum-likelihood (ML) decoding into simpler partial decoding steps. This leads to the development of concatenated codes.

Concatenated coding schemes were first introduced by Forney for achieving large coding gain. The technique essentially consists of combining two or more simple component codes to form a powerful codes. The resulted concatenated codes can achieve an error correction capability equivalent to that of a much larger code, and a decomposable structure which allows for relatively low decoding complexity.

Prior to 1993, most of applications including NASA deep space communication made use of serial concatenation scheme. That trend changes in 1993 when Berrou, Glavieux and Thitimajshima introduced a revolutionary concatenated code called Turbo Codes. This new code can be viewed as a parallel concatenation scheme which exploits the desirable characteristics of the concatenated codes and the strength of the iterative decoding to achieve high performance close to the Shannon limit. Constructed with only two simple component codes and an interleaver, the resulted Turbo Codes have shown to outperform all previous existing codes. With this iterative decoding technique, simulations for additive white Gaussian noise (AWGN) channel have reported that Turbo Codes can approach the Shannon limit within 0.7 dB or better at the bit error rate of 10^{-5}. Today, Turbo Codes concepts and principles have been extended to other concatenated schemes such as serial and hybrid concatenations. The latter can be thought of as a combination of serial and parallel concatenations.

In general, concatenated codes are constructed from two or more simple constituent or component codes (CC). There is no restriction on the composition or selection of CC component codes in the sense that any CC, be it linear block codes or convolutional codes, can be concatenated. However, studies have suggested some CCs are more suitable than others in certain applications. Depending on the classifications of the CCs used and the way they are concatenated, one obtains different families or structures of concatenated codes as illustrated in figure A.1. Figure A.2 depicts the three possible concatenation schemes - serial, parallel and hybrid concatenations.

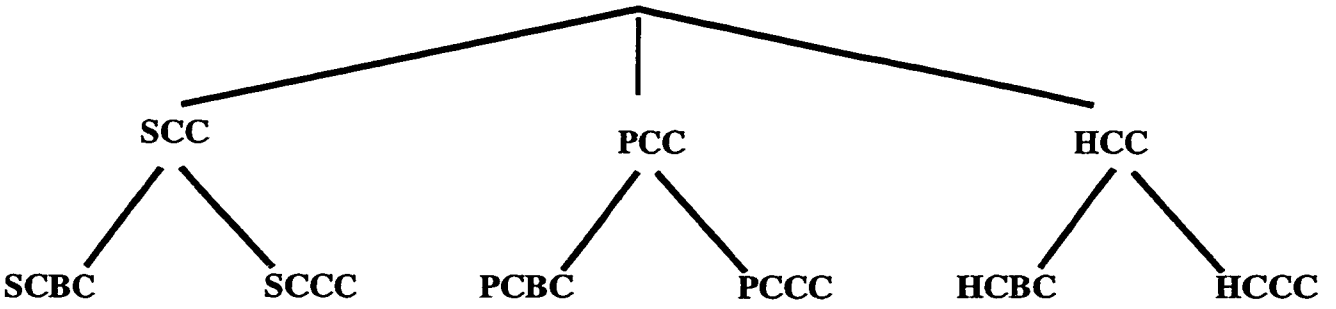


Figure A.1: Family tree for concatenated codes

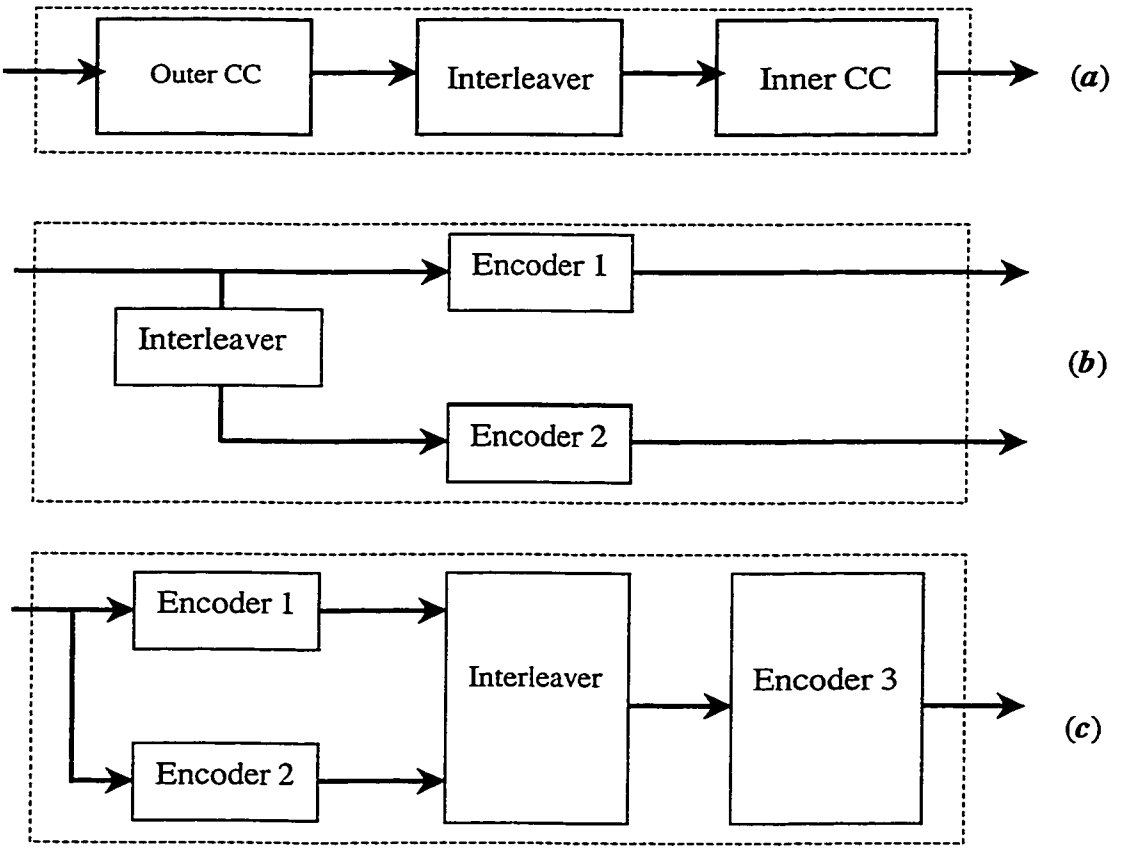


Figure A.2: Three concatenating schemes – (*a*) Serial; (*b*) Parallel; and (*c*) Hybrid.

A.2 SERIAL CONCATENATED CODES

A serial concatenated code $\mathbf{C}_{SCC}$ is basically a cascade of two constituent codes - an outer code $\mathbf{C}_O$ and an inner code $\mathbf{C}_I$. The outer code $\mathbf{C}_O$ is referred to the constituent code which gets encoded first but decoded last, whereas the inner code $\mathbf{C}_I$ is the one that gets encoded last but decoded first. The constituent codes (CC) can be either linear block codes or convolutional codes. If the CC are block codes, the resulting serial concatenated code is referred to as a serial concatenated block code (SCBC), otherwise it would be referred to as serial concatenated convolutional code (SCCC).

A.2.1 Encoding and Decoding

A typical serial concatenated code (SCC) encoder is depicted in figure A.2.*a*. The outer and the inner code encoders are usually linked by an interleaver whose primary role is to spread out the error bursts and distribute them evenly within the codewords. A data block of length k is first encoded by an (n_O, k) outer coded $\mathbf{C}_O$ to form an intermediate length n_O codeword which is in turn encoded by an (n_I, n_O) inner code $\mathbf{C}_I$ to form a SCC codeword of length n_I and overall code rate $R = k/n_I$.

As far as the encoding is concerned, the SCC performance is strongly influenced by the choice of the constituent codes. Some CCs yield good performance for the SCC at low signal to noise ratio (SNR), while others produce better performance at large SNR. In deep space communications, a (255, 223) Reed-Solomon code is used as an outer code while the inner code is the rate 1/2 and constraint length K = 7 convolutional codes. The search for constituent codes which optimize the performance of the resulted SCC over a range of SNR remains an active area of research within the coding community.

The decoding of the SCC is performed in two stages, starting with the decoding of the inner code C_I and followed by that of the outer code C_O. Any error burst resulted from the first stage decoding is de-interleaved prior to the second stage decoding in order to improve the performance. Although it is theoretically possible to perform single stage decoding with the SCC, it is more practical to use two stages decoding as the latter has complexity much lower than that of the alternative single stage decoding. Traditionally, the SCC is decoded using hard decision decoders. With the advent of Turbo Codes, SCC decoder is now implemented with elementary soft decision decoders which make it possible to perform iterative decoding and iteratively decrease the bit error rate.

A.2.2 Performance Analysis

This section focuses on the analysis the SCC performance; In particular, the analytical bounds for the BER probability for the SCBC are derived and the obtained results are extended to the SCCC when appropriate. Such bounds are useful for establishing some design guidelines or criterion for selecting good constituent codes.

Consider a rate (k/n) $\mathbf{C}_{SCBC}$, containing an outer code $\mathbf{C}_O$ and inner code $\mathbf{C}_I$ linked by an interleaver of length N. The outer code $\mathbf{C}_O$ and the inner code $\mathbf{C}_I$ are (N, k) and (n, N) linear block codes, respectively. Since $\mathbf{C}_O$ and $\mathbf{C}_I$ are linear, $\mathbf{C}_{SCBC}$ is also linear and thus subject to the uniform error property in the sense that its BER probability, P_B, can be computed with the assumption that all-zero codeword was transmitted [24]. In order to derive an upper bound of the BER probability for C_{SCBC} , the concept of uniform interleaver was introduced by Benedetto and Montorsi in [12]. A uniform interleaver of length N is simply a probabilistic device which maps a given input sequence of length N and weight w into all of its $\binom{N}{w}$ distinct permutations with equal probability $p = 1/\binom{N}{w}$ as shown in figure A.3.

It should be noted that the uniform interleaver of length N was introduced as an abstract concept to allow the computation of the SCBC average performance which corresponds to the mean performance of the SCBCs constructed from same constituent codes but all possible interleavers of length N. It was proved in [12] that for each value of SNR, there exists at least a practical interleaver of length N which performs as well as or better than the uniform interleaver.

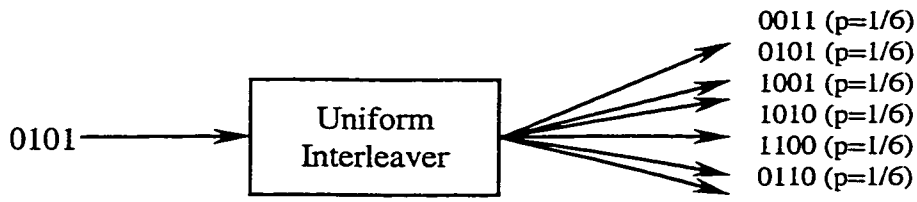


Figure A.3: Action of a uniform interleaver of length 4 on the sequence 0101.

A.2.2.1. Weight Enumerating Functions

Weight enumerating functions are often used to describe the weight distribution of a given code, i.e. the number of codewords of weight 0, 1, 2, ...N where N corresponds to the codeword length. Furthermore, they can be used to derive the performance bounds for the concatenated codes. For a given SCBC of length N, the corresponding weight enumerating function (WEF) can be expressed as

$$A^{C_{SCBC}}(D) \equiv \sum_{d=0}^{N} A_d^{C_{SCBC}} D^d \qquad \text{(A.1)}$$

where $A_d^{C_{SCBC}}$ denotes the number of SCBC codewords of Hamming weight d and D is a dummy variable. The WEF provides no information with regards to the information blocks associated with codewords. By extending it to include the weights of the input blocks, we obtain an input-output weight enumerating function (IOWEF) which can be expressed as

$$A^{C_{SCBC}}(W,D) \equiv \sum_{w,d} A_{w,d}^{C_{SCBC}} W^w D^d \qquad \text{(A.2)}$$

where W and D are dummy variables, and $A_{w,d}^{C_{SCBC}}$ denotes the number of SCBC codewords of Hamming weight d produced by input blocks of Hamming weight w. By Hamming weight, we mean the number of binary ones. Unless otherwise stated, all weights are referred to Hamming weight. The IOWEF, $A^{C_{SCBC}}(W,D)$, can be used to derive the corresponding conditional weight enumerating function (CWEF), $A^{C_{SCBC}}(w,D)$, which represents the weight distribution of $\mathbf{C}_{\mathrm{SCBC}}$ codewords of weight d generated by the input words of weight w. The SCBC CWEF can be expressed, in term of the IOWEF, as

$$A^{C_{SCBC}}(w,D) \equiv \left(\frac{1}{d!}\right)\left.\frac{\partial^d A^{C_{SCBC}}(W,D)}{\partial D^d}\right|_{D=0} \qquad \text{(A.3)}$$

Both of the above SCBC CWEF and IOWEF can be computed from the CWEFs of the corresponding inner and outer code, and the length of the uniform interleaver as shown in the following. Let $A^{C_O}(w,L)$ and $A^{C_I}(l,D)$ denote the CWEFs of the outer code $\mathbf{C}_{\mathrm{O}}$ and inner code $\mathbf{C}_{\mathrm{I}}$ respectively. Recall $\mathbf{C}_{\mathrm{O}}$ and $\mathbf{C}_{\mathrm{I}}$ are (n', k) and (n, n') linear block codes respectively. For each input block of length k and weight w, the outer encoder delivers an

outer codeword length n' and weight l, where $0 \le l \le n'$ Through the action of the length n' uniform interleaver which links the outer encoder to the inner encoder, the inner encoder produces $\binom{n'}{l}$ distinct codewords of length n and same weight d, from each outer codeword length n' and weight l. Therefore, it follows that the CWEF and IOWEF for the corresponding $\mathbf{C}_{SCBC}$ are given by

$$A^{C_{SCBC}}(w,D) = \sum_{l=o}^{n'} \frac{A_{w,l}^{C_o} \times A^{C_I}(l,D)}{\binom{n'}{l}} \tag{A.4}$$

$$A^{C_{SCBC}}(W,D) = \sum_{l=o}^{n'} \frac{A^{C_o}(W,l) \times A^{C_I}(l,D)}{\binom{n'}{l}} \tag{A.5}$$

where $A^{C_o}(W,l)$ denotes the conditional weight distribution of the input words which generate C_O codewords of weight l, and $A_{w,l}^{C_o}$ corresponds to the number of the outer codewords of weight l associated with input blocks of weight w. For instance, consider the (7,3) SCBC constructed using an interleaver of length n' = 4, and a (4,3) parity check code and a (7,4) Hamming code as outer and inner codes respectively. The IOWEF of the outer and inner codes are given by [15]

$$A^{C_o}(W,L) = 1 + W(3L^2) + W^2(3L^2) + W^3(L^4)$$
$$A^{C_I}(L,D) = 1 + L(3D^3 + D^4) + L^2(3D^3 + 3D^4) + L^3(D^3 + 3D^4) + L^4(D^7)$$

Therefore,

$$A^{C_o}(W,0) = 1 \qquad A^{C_I}(0,D) = 1$$
$$A^{C_o}(W,1) = 0 \qquad A^{C_I}(1,D) = 3D^3 + D^4$$
$$A^{C_o}(W,2) = 3W + 3W^2 \qquad A^{C_I}(2,D) = 3D^3 + 3D^4$$
$$A^{C_o}(W,3) = 0 \qquad A^{C_I}(3,D) = D^3 + 3D^4$$
$$A^{C_o}(W,4) = W^3 \qquad A^{C_I}(4,D) = D^7$$

thus there are eight outer codewords - one with weight 0, six with weight 2 and one with weight 4. The single codeword of weight 0 is generated from the input word of weight 0, the six codewords of weight 2 are produced from six input words, three of which have weight 1 and the other three have weight 2 and so on. Similarly, there are sixteen inner codewords - one codeword of weight 0, seven codewords of weight 3, seven codewords of weight 4 and one codeword of weight 7. Of the seven inner codewords of weight 4, one is generated from the outer codeword of weight 1, three are produced from three outer codewords of weight 2, whereas the remaining three inner codewords of weight 4 are generated from three outer codewords of weight 3. Finally, the corresponding SCBC IOWEF can be computed using equ. A.5 as

$$A^{C_{SCBC}}(W,D) = \sum_{l=0}^{4} \frac{A^{C_O}(W,l) \times A^{C_I}(l,D)}{\binom{4}{l}}$$

$$= \frac{1\times 1}{1} + \frac{0\times(3D^3+D^4)}{4} + \frac{W^3\times D^7}{1} + \frac{(3W+3W^2)(3D^3+3D^4)}{6} + \frac{0\times(D^3+3D^4)}{4}$$

$$= 1 + W(1.5D^3 + 1.5D^4) + W^2(1.5D^3 + 1.5D^4) + W^3D^7$$

A.2.2.2. Bit Error Probability

With the knowledge of the CWEF, one can now compute an upper bound for the bit error probability. For a given additive white Gaussian noise (AWGN) channel and maximum likelihood (ML) soft decision decoding (SDD), the upper bound to the bit error probability of the C_{SCBC} can be expressed in term of the previous CWEF as following [15]

$$P_B \le \sum_{w=1}^{k} \left(\frac{w}{k}\right) A^{C_{SCBC}}(w,D)\Big|_{D=e^{-RE_B/N_O}} \qquad (A.6)$$

where $R = k/n$ and E_B/N_0 denote the C_{SCBC} code rate and the per bit signal-to-noise ratio respectively. Although the above analytical bound is derived for SCBC, it is also valid for SCCC, provided that the input word length is much larger than the memory m of the corresponding inner and outer convolutional codes. Suppose a data block of length $(N\text{-}m)$ is encoded by a rate R_C convolutional encoder of memory m to produce a coded sequence of length N/R_C which starts and ends at the zero states. It is well known that for N much larger than m, the ML performance of this convolutional code is almost identical to the equivalent $(N/R_C, N\text{-}m)$ linear block code whose codewords are the convolutional coded sequences of length N/R_C which start and end at the zero states. This principle is equally valid for serial and parallel concatenated code [11][12][15]. As such, the upper bound of bit error probability (equ.A.6) derived for the SCBC also holds for the SCCC. The following discusses the design and analysis of a rate 1/3 SCCC.

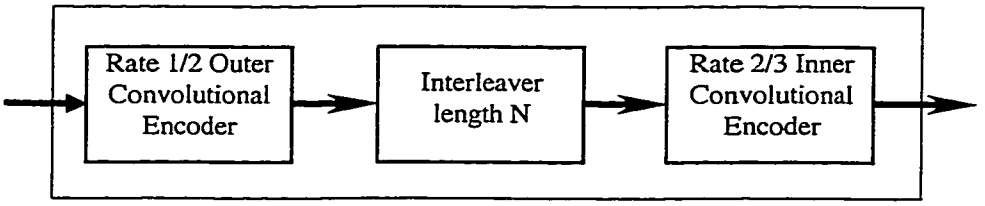


Figure A.4: A typical rate 1/3 SCCC encoder

In practice, SCCC is often preferred to SCBC because of two main reasons [15]. First, the maximum a posteriori (MAP) algorithms are much less complex for convolutional constituent codes than for block constituent codes, and secondly the interleaver gain can be made larger for convolutional than for block codes. A typical rate 1/3 SCCC encoder is illustrated in figure A.4. For the interleaver length, N, much larger than the memory of the outer and inner convolutional encoders, the ML performance of the SCCC is almost identical to that of the equivalent ($3N$, N- m_O - m_I) SCBC, where m_O and m_I correspond to the number of bits required to clear the memory cells of the outer and inner convolutional codes respectively. The upper bound of the bit error probability for this rate 1/3 SCCC can be expressed using the CWEF of the equivalent SCBC as [15]:

$$P_B \le \sum_{w=w_m^o}^{NR_C^o} \left(\frac{w}{NR_C^o} \right) \times A^{C_{SCBC}}(w, D)\Big|_{D=e^{-RE_B/N_O}}$$

$$= \sum_{d=d_m}^{N/R_C^I} \sum_{w=w_O}^{NR_C^O} \left(\frac{w}{NR_C^O} \right) \times A_{w,d}^{C_{SCBC}} \times e^{-dRE_B/N_O} \qquad \text{(A.7)}$$

where d_m is the minimum weight of the SCCC codewords; w_O corresponds to the minimum weight of an input word which generates an error event of the outer code. An error event of a convolutional code can be thought of as a segment which diverges from the zero state at time t and reemerges back to the zero state at time $t+j$, $j>0$. Finally, $A_{w,d}^{C_{SCBC}}$ denotes the CWEF of the equivalent SCBC and can be computed from the corresponding outer and inner CWEFs, i.e. $A_{w,l}^{C_o}$ and $A_{l,d}^{C_I}$.

To compute the CWEF of the outer and inner codes, consider a typical rate $R_C = p/q$ convolutional code $\mathbf{C}_C$ with memory m, and its equivalent $(N/R_C, N\text{-}pm)$ block code $\mathbf{C}_B$ whose codewords are all $\mathbf{C}_C$ codewords of length N/R_C which start and end at zero states. Obviously, the $\mathbf{C}_B$ codewords are concatenations of j error events of the convolutional code $\mathbf{C}_C$ (see figure A.5). Let $A^{C_C}(w, D, j)$, be the weight enumerating function of the sequences of the convolutional code $\mathbf{C}_C$ that concatenate j error events with a total input weight w, it follows that

$$A^{C_C}(w, D, j) = \sum_d A^{C_C}_{w,d,j} D^d \qquad \text{(A.8)}$$

where $A^{C_C}_{w,d,j}$ corresponds to the number of codewords of weight d, input weight w and number of concatenated error events j. For N much larger than the memory length of the convolutional encoder, the coefficient $A^{C_B}_{w,d}$ of the equivalent block code $\mathbf{C}_B$ can be approximated as [15]

$$A^{C_B}_{w,d} \sim \sum_{j=1}^{n_M} \binom{N/p}{j} \times A_{w,d,j} \qquad \text{(A.9)}$$

where n_M is the maximum number of the error events concatenated within a $\mathbf{C}_B$ codeword of weight d which is produced from an input of weight w. Note that n_M varies with d and w and is specific for each encoder. The following describes the computation of the CWEF, $A^{C_{SCCC}}_{w,d}$, and the upper bound bit error probability of a generic serial concatenated block codes, $\mathbf{C}_{SCBC}$, using the approximation obtained in eq. (A.9).

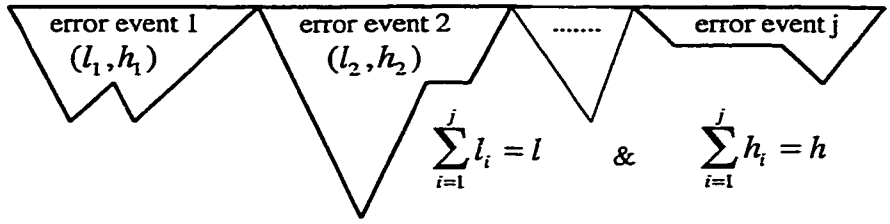


Figure A.5: A $\mathbf{C}_B$ codeword of weight h generated by an input sequence of weight l. The codeword is a concatenation of j error events.

Consider a serial concatenated convolutional code, $\mathbf{C}_{SCCC}$, constructed with a rate $R_O = p_O/q_O$ outer and $R_I = p_I/q_I$ inner convolutional codes of memory m_O and m_I respectively, and an interleaver of length N. Obviously, the equivalent block codes for the outer and inner convolutional codes are $(N/R_O, N\text{-}p_O m_O)$ and $(N/R_O R_I, [N/R_O] - p_I m_I)$. Carrying the same analysis as described for the previous rate $R_C = p/q$ convolution code $\mathbf{C}_C$, we obtain

$$A_{w,k}^{C_O} \sim \sum_{n^O=1}^{n_M^O} \binom{N/p_O}{n^O} \times A_{w,k,n^O}^{C_O} \tag{A.10}$$

$$A_{w,k}^{C_I} \sim \sum_{n^I=1}^{n_M^I} \binom{N/R_O p_I}{n^I} \times A_{w,k,n^I}^{I} \tag{A.11}$$

$$A_{w,d}^{C_{SCCC}} \sim \sum_{k=d_f^O}^{N} \sum_{n^O=1}^{n_M^O} \sum_{n^I=1}^{n_M^I} \frac{\binom{N/p_O}{n^O} \times \binom{N/R_O p_I}{n^I}}{\binom{N}{k}} \times A_{w,k,n^O}^{C_O} \times A_{k,d,n^I}^{C_I} \tag{A.12}$$

where d_f^o is the minimum free distance of the outer convolutional code $\mathbf{C}_O$, i.e. the minimum weight of error events for the outer code. Using binomial approximation $\binom{N}{n} \sim (N^n / n!)$, eq. A.12 can be rewritten as

$$A_{w,d}^{C_{SCCC}} \sim \sum_{k=d_f^o}^{N} \sum_{n^O=1}^{n_M^o} \sum_{n^I=1}^{n_M^I} N^{n^O+n^I-k} \times A_{w,k,n^O}^{C_O} \times A_{k,d,n^I}^{C_I} \frac{k!}{n^O! n^I! (p_O)^{n^O} (p_I)^{n^I} (R_O)^{n^I}} \quad \text{(A.13)}$$

Substituting eq. A.13 in A.7 yields

$$P_B \leq \sum_{d=d_m}^{N/R_I} e^{-dRE_B/N_0} \left[\sum_{w=w_O}^{NR_O} \sum_{k=d_f^o}^{N} \sum_{n^O=1}^{n_M^o} \sum_{n^I=1}^{n_M^I} (N^{n^O+n^I-k-1}) \times \right.$$

$$\left. \left(A_{w,k,n^O}^{C_O}\right) \times \left(A_{k,d,n^I}^{C_I}\right) \times \left(\frac{k!\, i}{n^O! n^I! (p_O)^{n^O} (p_I)^{n^I} (R_O)^{n^I+1}} \right) \right] \quad \text{(A.14)}$$

Eq. A.14 can be used to derive important design considerations. First it is observed that the upper bound to the bit error probability is obtained by adding terms of the first summation with respect to the index d which corresponds to the weight of SCC codewords. Each term of the first summation is associated with a series of coefficients, i.e. the expression in the square bracket ([]), which depend on several parameters including the interleaver length N. For large N and each term of the first summation, the dominating coefficient is the one with the largest exponent of N. It can be shown that the largest exponent of N is $(-d_f^o/2)$ for even values of d_f^o and $[-(d_f^o+1)/2]$ for odd values of d_f^o. For all terms of the first summation, their associated dominating coefficients decrease with N, hence always leading to an interleaver gain. Furthermore, for large value of E_B/N_0, the first summation is dominated by the first term, i.e. $d = d_m$. Based on these observations, Benedetto suggested some important SCCC design considerations for minimizing the upper bound bit error probability, as quoted below [15]:

a. for the values of N and E_B/N_0 where the SCCC performance is dominated by its free distance $d_f^{C_{sccc}} = d_m$, increasing the interleaver length would result in better performance;

b. to increase the interleaver gain, an outer code with large odd d_f^o should be selected;

c. to improve the performance with E_B/N_0, a combination of the outer and inner codes which yields large d_m should be selected;

d. the inner encoder must be recursive convolutional encoder with large effective free distance (the notion of effective free distance will become relevant later);

e. the outer encoder should be non-recursive convolutional encoder whose free distance codewords are generated from minimum number of input sequences.

f. for low values of E_B/N_0, the above choice of outer and inner convolutional encoders should be reversed.

A.3 PARALLEL CONCATENATED CODES

A parallel concatenated code (PCC) C_{PCC} is basically constructed from several constituent codes linked together by interleavers. A typical rate $1/n$ systematic PCC encoder contains $(n-1)$ constituent systematic encoders (ENC) interconnected as shown in figure A.6. To reduce the decoder complexity, these ENC encoders are chosen to be identical. Each ENC encoder independently and simultaneously encodes the input sequence to produce two output sequences - one systematic sequence and one parity sequence. The systematic sequences from individual ENC encoders are identical, but the parity sequences are not for most of the times and are uncorrelated. As such, only one systematic sequence needs to be transmitted to increase the overall code rate. A good PCC encoder minimizes the probability that its constituent encoders simultaneously output parity sequences with low weight for a given input sequence. This is normally accomplished by either permuting or interleaving the input sequence prior to encoding by the ENC encoders .

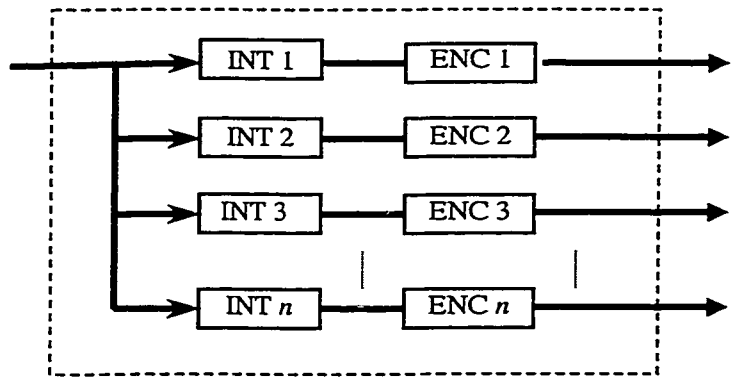


Figure A.6: A typical rate $1/n$ PCC encoder.

Like the serial concatenated codes, there are two sub-groups of parallel concatenated codes, namely the parallel concatenated block code (PCBC) and the parallel concatenated convolutional (PCCC). Some literature refers to PCCC as Turbo Codes. Recent works have indicated that the PCBC can outperform the PCCC at high code rates. For instance at code rate ($0.95 < R < 1$), the PCBC can operate at less than 0.45 dB from the Shannon's limit [19]. The PCCC, on the other hand, performs quite well at code rate $R \sim 0.5$, but exhibits significant performance degradation at code rate greater than 0.7 [19]. There has been some suggestions with regard to establishing a "threshold rate" for a given BER. If an application requires that the code rate be greater than the threshold rate, than the PCBC should be used; else select the PCCC. Some characteristics of PCBC are briefly pointed out here, while the details of PCCC are studied in depth in chapter 2. The extension of the PCCC to the PCBC is straight forward.

A parallel concatenated code (PCBC) $\mathbf{C}_{PCBC}$ has slightly larger code rate than the SCBC $\mathbf{C}_{SCBC}$, however the latter has higher coding gain. Suppose two linear block codes (n_1, k_1, d_1) and (n_2, k_2, d_2) are used to constructed a $\mathbf{C}_{PCBC}$ and a $\mathbf{C}_{SCBC}$. The (n, k, d) triplet denotes codeword length, the input block length and the minimum Hamming distance of the code. The code rates and minimum Hamming distances for the $\mathbf{C}_{PCBC}$ and $\mathbf{C}_{SCBC}$ are given by

$$R_{PCBC} = [(R_1R_2)/(R_1 + R_2 - R_1R_2)] \quad \text{(A.15)}$$

$$R_{SCBC} = R_1R_2 \quad \text{(A.16)}$$

$$d_{PCBC} = (d_1 + d_2 - 1) \quad \text{(A.17)}$$

$$d_{SCBC} = (d_1\, d_2) \quad \text{(A.18)}$$

where R_{PCBC} and d_{PCBC}, and R_{SCBC} and d_{SCBC} denote the code rate and Hamming distance of the PCBC and SCBC respectively. R_1 and R_2 are the code rates of the (n_1, k_1, d_1) and (n_2, k_2, d_2) linear block codes. The asymptotic coding gain of the PCBC and SCBC are given by G_{PCBC} and G_{PCBC} respectively as

$$G_{PCBC} = 10 \log (R_{PCBC} \cdot d_{\mathrm{PCBC}}) \quad \text{(A.19)}$$

$$G_{SCBC} = 10 \log (R_{SCBC} \cdot d_{\mathrm{SCBC}}) \quad \text{(A.20)}$$

Similar to $\mathbf{C}_{\mathrm{SCBC}}$, the $\mathbf{C}_{\mathrm{PCBC}}$ performance increases as a function of the interleaver length but decreases as the code rate goes higher. There are two options of interleaving: random and deterministic (or nonrandom). Past simulations have suggested that there is no significant performance difference between these interleavers. This is generally not the case for the PCCC where random interleaver can achieve better performance than the nonrandom one [19].

A.4 ITERATIVE DECODING

Since the early days of Shannon theory, it has been well known that increasing the codeword length N of linear block codes would yield better performance. Unfortunately, increasing N would also cause the complexity of maximum-likelihood decoding (ML) algorithms to increases up to a level where decoding becomes physically unrealizable. An alternative solution is to iteratively perform sub-optimal decoding on fragments of the codeword as opposed to performing the optimum ML decoding on the whole codeword. Such decoding technique referred to as iterative decoding.

A.4.1 Principles of Iterative Decoding

The concept of iterative decoding, in the context of information theory, began apparition with the Turbo Codes (TC) development. Essentially, it involves the use of several soft decision decoders with a feedback loop to repeatedly or iteratively decode concatenated codewords several times until a predefined stopping criteria is met. Such stopping criteria can be either the number of iterations which is limited by the maximum allowable decoding delay, or the cross entropy which can be thought of as a measure of the BER improvement between two consecutive iterations.

At the end of each iteration, the iterative decoding delivers a sequence of "soft" estimated bits along with a sequence of extrinsic information which represents an incremental or an additional information about the transmitted data bits. The extrinsic information obtained from the current iteration is then used with the same received sequence to improve the decoding decision of the next iteration. Once the pre-defined stopping criteria has been met, the iterative decoding produces a hard decision output (0 or 1) for each decoded bit based on its reliability value of the last iteration.

A typical iterative decoder for a SCC built from two constituent codes is shown in figure A.7. It is made up of two serially cascaded elementary soft-input/soft-output (SISO) decoders, an interleaver, a de-interleaver, a feedback loop and a conventional hard decision decoder. The principle idea behind the SISO decoders and the feedback loop is to exploit the so-called extrinsic information which is available from the second stage or inner SISO decoder, and to go back to the first stage or outer SISO decoder and start the decoding operations over. By using the extrinsic information from the second stage SISO decoder to update the a priory information of the relevant bits to be decoded, the first stage SISO decoder is able to improve the decoding decisions which in turn yields a better performance for the second stage decoding.

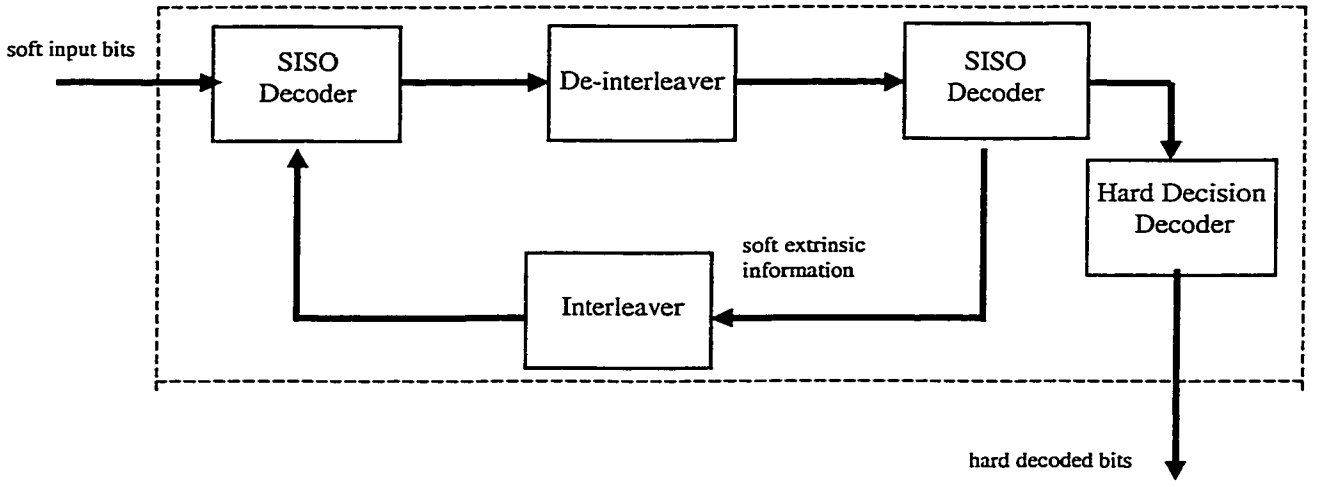


Figure A.7: An example of an SCC iterative decoder.

A.4.2 Probability and Log-likelihood Algebra

The mathematical tools needed for the binary iterative decoding are presented below. Let u be a binary random variable in a GF(2) with the two elements {-1,+1} corresponding to the information bits {0,1}, i.e. -1 is the null element under the modulo-2 addition, $\oplus$. The log-likelihood ratio (LLR) of u, denoted by $L(u)$, is defined as:

$$L(u) = \ln\left[\frac{P(u=+1)}{P(u=-1)}\right] \tag{A.21}$$

where $P(u = +1)$ is the probability that $u = +1$. $L(u)$ is often referred to as the soft or L-value of u: its sign, sign$[L(u)]$, represents the hard decision whereas its magnitude, $|L(u)|$, indicates the reliability of this decision. Using Bayes' theorem, it can be shown that a conditional log-likelihood ratio, $L(u|v)$, is given by [22]:

$$\begin{aligned} L(u|v) &= \ln\left[\frac{P(u=+1)}{P(u=-1)}\right] + \ln\left[\frac{P(v\,|\,u=+1)}{P(v\,|\,u=-1)}\right] \\ &= L(u) + L(v|u) \end{aligned} \tag{A.22}$$

Using the relationships

$$\begin{aligned} P(u_1 \oplus u_2 = +1) = {} & [P(u_1 = +1)\,]\,[P(u_2 = +1)] \\ & + [1 - \{P(u_1 = +1)\}]\,[1 - \{P(u_2 = +1)\}] \end{aligned} \tag{A.23}$$

$$P(u=+1) = \frac{e^{L(u)}}{1+e^{L(u)}} \tag{A.24}$$

$$\begin{aligned} P(u=-1) &= 1 - P(u=+1) \\ &= \frac{1}{1+e^{L(u)}} \end{aligned}$$

The LLR of a summation can be approximated as [9]:

$$L(u_1 \oplus u_2) = \ln\left[\frac{e^{L(u_1)} + e^{L(u_2)}}{1 + e^{L(u_1)} \cdot e^{L(u_2)}}\right]$$

$$\approx (-1) \times \text{sign}\,[L(u_1)] \times \text{sign}\,[L(u_2)] \times \min[|L(u_1)|, |L(u_2)|] \quad \text{(A.25)}$$

Defining the summation of log-likelihood, <+>, as

$$L(u_1) <+> L(u_2) \equiv L(u_1 \oplus u_2) \quad \text{(A.26)}$$

With

$$L(u) <+> +\infty = -L(u)$$
$$L(u) <+> +0 = 0$$

Using eq. (A.25) and eq. (A.26) can be extended to [9] [22]

$$\sum_{i=1}^{N} <+> L(u_i) = L\left(\sum_{i=1}^{N} \oplus u_i\right)$$

$$\approx (-1)\left(\prod_{i=1}^{N} sign[L(u_i)]\right) \times \min_{i=1\ldots N}\left(|L(u_i)|\right) \quad \text{(A.27)}$$

A.4.3 Channel Reliability

Consider a binary symmetric channel (BSC) with a cross-over probability P_C. The reliability value of such channel, denoted by L_C, is defined as the LLR of the cross-over probability, i.e. $L_C = \ln[(1-P_C)/P_C]$. Let the binary random variables u, x and y represent the information, the transmitted and the received bits of a systematic code respectively. If one encodes the binary value of u which has a soft value of $L(u)$, one obtains x whose soft value is $L(x)$. Using equation (A.22) to compute the soft value of the received bit y conditioned to the transmission of the bit x, one obtains [3][2][9][14][22]:

$$L(x|y) = \ln\left[\frac{P(x=+1)}{P(x=-1)}\right] + \ln\left[\frac{P(y\,|\,x=+1)}{P(y\,|\,x=-1)}\right]$$

$$= L(x) + \ln \frac{\exp[-(E_S/N_0)(y-a)^2]}{\exp[-(E_S/N_0)(y+a)^2]}$$

$$= L(x) + y \cdot L_C \qquad \text{(A.28)}$$

$$L_C = 4 \cdot a \cdot (E_S/N_0) = 2 \cdot a \cdot (1/\sigma^2)$$

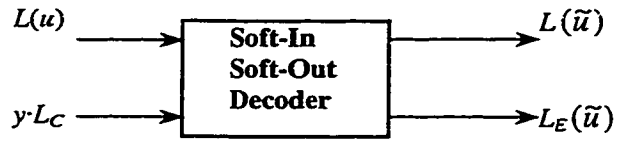
Where E_S ($=RE_B$) and σ^2 denotes the symbol energy and noise variance respectively. For a Gaussian channel, a =1 while for a fading channel a is the fading amplitude. For a BSC, L_C is the LLR of the cross-over probability as indicated previously. Of particular interest, equ. (A.28) can be extended to a statistically independent transmission such as a dual diversity or repetitive transmission, i.e.

$$L(x|y_1,y_2) = y_1 \cdot L_{C1} + y_2 \cdot L_{C2} \qquad \text{(A.29)}$$

A.4.4. Soft Decision Decoding and Extrinsic Information

Consider an (N,K) systematic code where $\boldsymbol{u} = (u_1, u_2, u_3, \ldots, u_K)$ and $\boldsymbol{x} = (u_1, u_2, u_3, \ldots, u_K, x_{K+1}, x_{K+2}, \ldots, x_N)$ represent the information and the coded sequences respectively. Suppose $\boldsymbol{x}$ is transmitted and received as $\boldsymbol{y} = (y_1, y_2, y_3, \ldots, y_N)$ which is decoded using a soft-in/soft-out decoder shown in figure A.8. This decoder can be viewed as a symbol-by-symbol maximum a posteriori (MAP) decoder which outputs an estimate of the soft a posteriori value of each information bit, i.e. $L(\tilde{u})$,

$$L(\tilde{u}) \equiv L(u|\mathbf{y}) = \ln\left[\frac{P(u=+1\,|\,\mathbf{y})}{P(u=-1\,|\,\mathbf{y})}\right] \qquad \text{(A.30)}$$



$L(u)$: a priori values for all information bits.
$y \cdot L_C$: channel output values for all coded bits.
$L(\tilde{u})$: a posteriori values for all information bits.
$L_E(\tilde{u})$: extrinsic values for all information bits.

Figure A.8: A typical soft-in/soft-out (SISO) decoder.

The above decoder admits the soft a priori values $L(u)$ for all information bits and the soft channel values $(y \cdot L_C)$ for all coded bits as inputs, and then delivers the soft outputs $L(\tilde{u})$ for all information bits along with the associated extrinsic information $L_E(\tilde{u})$. The extrinsic information $L_E(\tilde{u}_j)$ for the information bit u_j can be thought of as additional information about that information bit gained from decoding all the other bits in the codeword $\boldsymbol{x}$. It is not influenced by the $L(u_j)$ and $(y_j \cdot L_C)$, and is used to update the soft a priori values $L(u_j)$ for the next iteration. The soft outputs produced by the above decoder is given by

$$L(\tilde{u}) = L(u) + (y \cdot L_C) + L_E(\tilde{u}) \qquad \text{(A.31)}$$

Equ. (A.31) clearly suggests that there are three independent estimates for the LLR of the information bits. They are the soft channel values $(y \cdot L_C)$, the extrinsic values $L_E(\tilde{u})$ and the soft a priory values $L(u)$. At the beginning of the decoding, i.e. at the first iteration, one does not know the a priory probability of the information bits. Hence, one assumes equally likely information bits (i.e. $P(0) = P(1) = 1/2$), and initializes $L(u) = 0$. At the second iteration, one sets $L(u) = L(\tilde{u})$ obtained at the first iteration. At the $(j+1)^{th}$ iteration, one sets $L(u) = L(\tilde{u})$ obtained from the j^{th} iteration, and so on.

A.4.5 Iterative Decoding Applications

To put things into perspective, consider a two-dimensional product code. The information sequence $\boldsymbol{u} = (u_1, u_2, u_3,\ldots,u_K)$ is arranged in a $(k_1 \times k_2 = K)$ matrix and is encoded twice, row-wise and column-wise. The resulted coded sequence consists of three distinct components namely: the systematic $\boldsymbol{u}$, the horizontal parity $\boldsymbol{x}_h$ and the vertical parity $\boldsymbol{x}_v$ as shown in figure A.9. Suppose the coded sequence $\boldsymbol{x} = (\boldsymbol{u}, \boldsymbol{x}_h, \boldsymbol{x}_v)$ is transmitted and received as a sequence $\boldsymbol{y}$. The iterative decoding of the sequence $\boldsymbol{y}$ can be carried out as following [22]:

i) Set the initial soft a priori information $L(u) = 0$, i.e. $P(0) = P(1) = 1/2$.

ii) Compute the horizontal extrinsic information using equ. (A.31), i.e.

$$L_{Eh}(\tilde{u}) = L(\tilde{u}) - L(u) - (y \cdot L_C)$$

iii) Set $L(u) = L_{Eh}(\tilde{u})$

iv) Compute the vertical extrinsic information, i.e.

$$L_{Ev}(\tilde{u}) = L(\tilde{u}) - L(u) - (y \cdot L_C)$$

v) Set $L(u) = L_{Ev}(\tilde{u})$

vi) If the stopping criteria is met, go to vii; else go to ii.

vii) Compute soft output decision, i.e.

$$L(\tilde{u}) = (y \cdot L_C) + L_{Eh}(\tilde{u}) + L_{Ev}(\tilde{u})$$

Note that systematic code is used as an example to discuss the iterative decoding because it is the preferred choice in the TC construction. This very same principle can also be extended to the non-systematic codes.

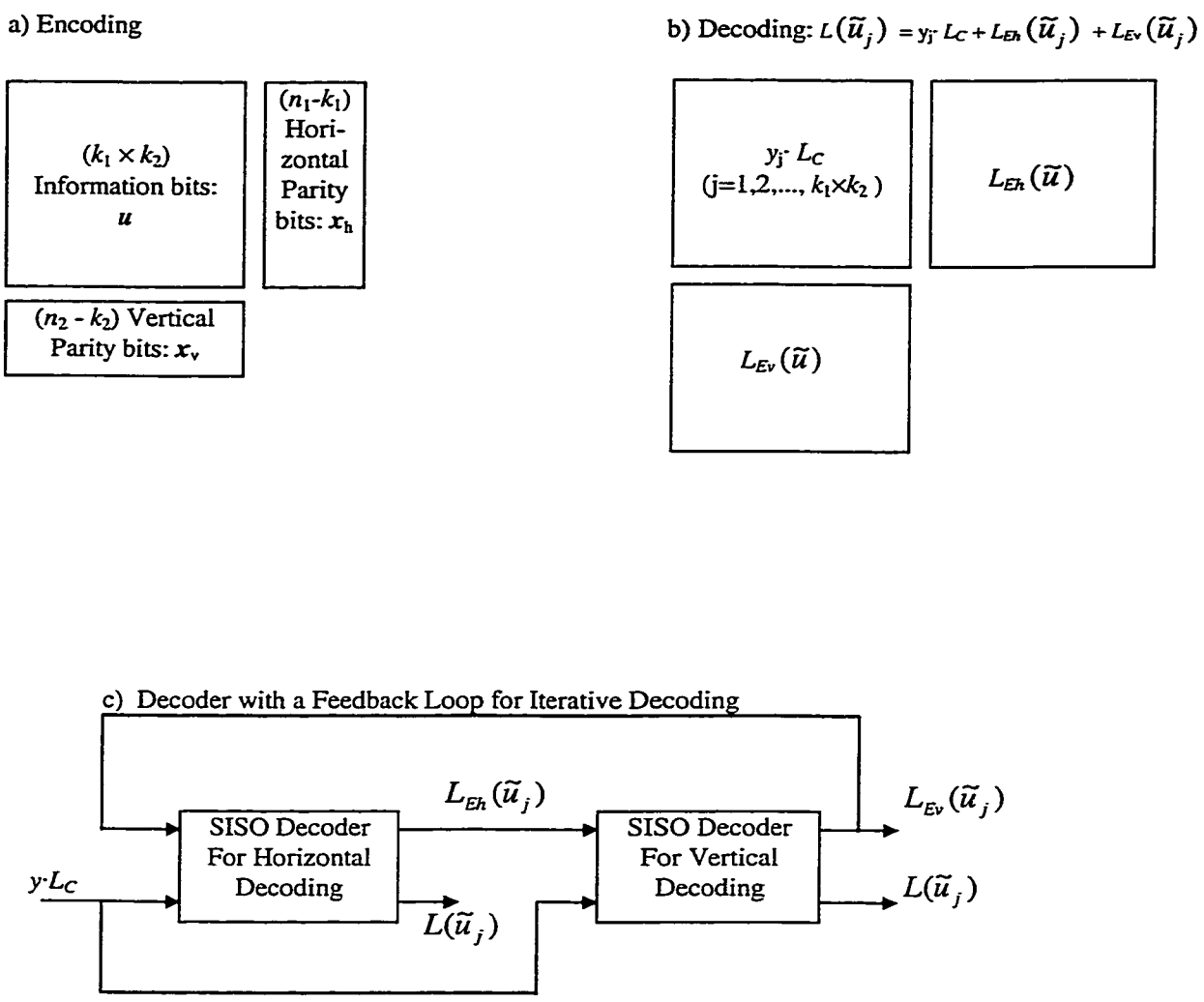


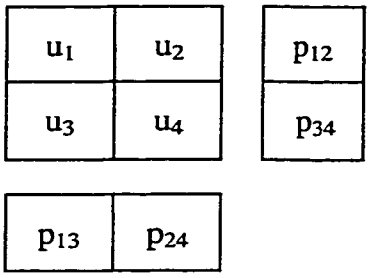
Figure A.9: Example of two dimensional product code with iterative decoding.

A.4.6 An Example of Iterative Decoding

As a practical example of the iterative decoding, consider a two-dimensional product code constructed from two single parity check (SPC) systematic code (3,2,2). Each product code codeword $\boldsymbol{x}$ consists of four information bits followed by four parity check bits, i.e.

$$\begin{aligned}\boldsymbol{x} &= (u_1, u_2, u_3, u_4, p_{12}, p_{34}, p_{13}, p_{24}) \\ &= (x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8)\end{aligned}$$

where



$$u_1 \oplus u_2 = p_{12}$$
$$u_3 \oplus u_4 = p_{34}$$
$$u_1 \oplus u_3 = p_{13}$$
$$u_2 \oplus u_4 = p_{24}$$

$$\Rightarrow u_1 = u_2 \oplus p_{12} \quad \text{and} \quad u_1 = u_3 \oplus p_{13}$$
$$\Rightarrow u_2 = u_1 \oplus p_{12} \quad \text{and} \quad u_2 = u_4 \oplus p_{24}$$
$$\Rightarrow u_3 = u_1 \oplus p_{13} \quad \text{and} \quad u_3 = u_4 \oplus p_{34}$$
$$\Rightarrow u_4 = u_2 \oplus p_{24} \quad \text{and} \quad u_4 = u_3 \oplus p_{34}$$

Assume the codeword $\boldsymbol{x}$ = (10101100) is transmitted over an AWGN channel with zero mean and unit variance, i.e. $L_C = 2$, using a bipolar signal sequence, i.e. (+1,-1,+1,-1,+1,+1,-1,-1). Suppose the transmitted sequence is corrupted and received as $\boldsymbol{y} = (y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8)$ = (1.00, 0.10, 0.90, 0.15, 0.30, 0.60, -0.30, -0.90). The expressions and numerical values of the extrinsic information are computed below

$$\begin{aligned}L_{Eh}(\tilde{u}_1) &= [(y_2 \cdot L_C) + L(u_2)] <+> (y_5 \cdot L_C) \\ &= [0.20 + 0] <+> 0.60 \\ &= -0.20\end{aligned}$$

$$L_{Eh}(\tilde{u}_2) = [(y_1 \cdot L_C) + L(u_1)] <+> (y_5 \cdot L_C)$$
$$= [2.0 + 0\ ] <+> 0.60$$
$$= - 0.60$$
$$L_{Eh}(\tilde{u}_3) = [(y_4 \cdot L_C) + L(u_4)] <+> (y_6 \cdot L_C)$$
$$= [0.30 + 0\ ] <+> 0.3$$
$$= - 0.30$$

$$L_{Eh}(\tilde{u}_4) = [(y_3 \cdot L_C) + L(u_3)] <+> (y_6 \cdot L_C)$$
$$= [\ 1.80 + 0] <+> 1.20$$
$$= - 1.20$$
$$L_{Ev}(\tilde{u}_1) = [(y_3 \cdot L_C) + L(u_3)] <+> (y_7 \cdot L_C)$$
$$= [1.80 - 0.30] <+> (-0.60)$$
$$= 0.60$$
$$L_{Ev}(\tilde{u}_2) = [(y_4 \cdot L_C) + L(u_4)] <+> (y_8 \cdot L_C)$$
$$= [\ 0.30 - 1.20\ ] <+> (-1.80)$$
$$= -0.90$$
$$L_{Ev}(\tilde{u}_3) = [(y_1 \cdot L_C) + L(u_1)] <+> (y_7 \cdot L_C)$$
$$= [2.00 - 0.20] <+> (-0.60)$$
$$= 0.60$$
$$L_{Ev}(\tilde{u}_4) = [(y_2 \cdot L_C) + L(u_2)] <+> (y_8 \cdot L_C)$$
$$= [0.20 - 0.60] <+> (-1.80)$$
$$= - 0.40$$

$$L(\tilde{u}_1) = (y_1 \cdot L_C) + L_{Eh}(\tilde{u}_1) + L_{Ev}(\tilde{u}_1) = 2.40$$
$$L(\tilde{u}_2) = (y_2 \cdot L_C) + L_{Eh}(\tilde{u}_2) + L_{Ev}(\tilde{u}_2) = -1.30$$
$$L(\tilde{u}_3) = (y_3 \cdot L_C) + L_{Eh}(\tilde{u}_3) + L_{Ev}(\tilde{u}_3) = 2.10$$
$$L(\tilde{u}_4) = (y_4 \cdot L_C) + L_{Eh}(\tilde{u}_4) + L_{Ev}(\tilde{u}_4) = -1.30$$

The above results are rewritten in tabular form

$y_j \cdot L_C$			after first horizontal decoding (L_{Eh})		after first vertical decoding (L_{Ev})	
2.00	0.20	0.60	−0.20	−0.60	0.60	−0.90
1.80	0.30	1.20	−0.30	−1.20	−0.60	−0.40
−0.60	−1.80					

after one iteration

2.40	−1.30
2.10	−1.30

Performing the hard decision decoding (HDD) on the soft values of the first iteration, the received sequence would be decoded as (1010), which corresponds indeed to the transmitted sequence. It has just been shown how a particular iterative decoding scheme is used to iteratively decode a product codes. This very same scheme is also used to decode the current Turbo Codes.

Several interesting remarks can be made to the above iterative decoding scheme. First, it is noted that only the extrinsic values associated with the received systematic bits are updated during the iterative process. Secondly, it is observed that the extrinsic values of the parity bits are not considered during the decoding process, thus no improvement to the parity components of the received sequence. The same soft values of the horizontal and vertical parity bits are used repeatedly to update the soft extrinsic values of the received systematic bits. This is acceptable if all errors are confined to systematic bits and all parity bits are received correctly. If one of these parity bits is in error, it can adversely affect every iteration and consequently the overall decoding performance.

The reason for not being able to compute the extrinsic values of the received parity bits and use it to update the soft values of the corresponding received parity bits is due to the inherent encoding structure of these specific product codes. As seen in chapter 2, the same scenario takes place in the current Turbo Codes. In the previous example, the very same information sequence is independently coded twice. The information sequence $(u_1u_2u_3u_4)$ is first coded by an encoder, say ENC1, to produce a parity sequence $(p_{12}p_{34})$, and then coded again by another encoder, say ENC2, to produce a parity sequence $(p_{13}p_{24})$. As a result, the parity bits from ENC1 are weakly correlated to the parity bits from ENC2, and because ENC1 and ENC2 work independently, the reception of $(p_{12}p_{34})$ only provides additional information on $(u_1u_2u_3u_4)$ but not $(p_{13}p_{24})$. Similarly, the reception of a parity bits $(p_{13}p_{24})$ provides extra information on only $(u_1u_2u_3u_4)$ but not $(p_{12}p_{34})$. Hence it is not possible to compute the extrinsic information for any parity bit.

6. REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo Codes," in Proc. IEEE Int. Conf. Commun., 1993, pp. 1064-1070.

[2] C. Berrou and A. Glavieux, "Near Optimum Error Correcting Coding and Decoding: Turbo Codes," IEEE Trans. Commun., vol. 44, no. 10, pp. 1261-1271, Oct. 1996.

[3] J. Hagenauer, "The Turbo Principle: Tutorial Introduction and State of the Art," in Proceedings of the Int. Symp. On Turbo Codes and Related Topics, Brest, France, pp.1-11, Sep. 1997.

[4] L.R. Bahl, J. Cocke, F. Jelineck, and J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," IEEE Trans. Inform. Theory, vol. IT-20, pp. 284-287, Mar. 1974.

[5] G. Battail, M.C. Decouvelaere, and P. Godlewski, "Replication Decoding," IEEE Trans. Inform. Theory, vol. IT-25, pp. 332-345, May. 1979.

[6] G.D. Forney, Jr., "Concatenated Codes," Cambridge, MA: MIT. Press, 1966.

[7] G. Battail, "On Random Like Codes," in Information Theory and Applications II, J-Y Chouinard, Paul Fortier and T. Aaron Gulliver, Eds., LNCS 1133 Springer-Verlag, 1995, pp.76-93.

[8] T. Kasami et al., "On Bit-Err0r Probability of a Concatenated Coding Scheme," IEEE Trans. Commun., vol. 45, no. 5, pp. 536-543, May 1997.

[9] B. Sklar, "A Primer On Turbo Code Concepts," IEEE Commun. Mag., pp.92-102, Dec. 1997.

[10] D. Divsalar and F. Pollara, "On The Design of Turbo Codes," TDA Progress Report vol. 42-123, Jet Propulsion Laboratory, Pasadena, California, pp.99-121, November 15, 1995.

[11] S. Benedetto and G. Montorsi, "Design of Parallel Concatenated Convolutional Codes," IEEE Trans. Commun., vol. 44, no. 5, pp. 591-600, May 1996.

[12] S. Benedetto and G. Montorsi, "Unveiling Turbo Codes: Some Results on Parallel Concatenated Schemes," IEEE Trans. Inform. Theory, vol. 42, no. 2, pp. 409-428, Mar. 1996.

[13] K.R. Narayanan and G.L. Stuber, "Selective Serial Concatenation of Turbo Codes," IEEE Commun. Lett., vol. 1, no. 5, pp. 136-139, Sep. 1997.

[14] D. Divsalar and F. Pollara, "Hybrid Concatenated Codes and Iterative Decoding" TDA Progress Report vol. 42-130, Jet Propulsion Laboratory, Pasadena, California, pp.1-23, August 15, 1997.

[15] S. Benedetto et al., "Serial Concatenation of Interleaved Codes: Performance Analysis, Design and Iterative Decoding," TDA Progress Report vol. 42-126, Jet Propulsion Laboratory, Pasadena, California, pp.1-26, August 15, 1996.

[16] H. Nickl, J. Hagenauer, and F. Burkert, "Approaching Shannon's Capacity Limit by 0.27dB Using Simple Hamming Codes," IEEE Commun. Lett., vol. 1, no. 5, pp. 130-132, Sep. 1997.

[17] S. Benedetto et al., "Analysis, Design, and Iterative Decoding of Double Serially Concatenated Codes with Interleavers," IEEE J. Select. Areas Commun. , vol. 16, no. 2, pp.231-244, Feb. 1998.

[18] P. Adde et al., "Block Turbo Decoder Design," in Proceedings of the Int. Symp. On Turbo Codes and Related Topics, Brest, France, pp.166-169, Sep. 1997.

[19] R. Pyndiah, "Iterative Decoding of Product Codes: Block Turbo Codes," in Proceedings of the Int. Symp. On Turbo Codes and Related Topics, Brest, France, pp.71-79, Sep. 1997.

[20] D.F. Freeman and A. M. Michelson, "A Two-Dimensional Product Code with Robust Soft-Decision Decoding," IEEE Trans. Commun., vol. 44, no. 10, pp. 1222-1226, Oct. 1996.

[21] R. M. Pyndiah, "Near-Optimum Decoding of Product Codes: Block Turbo Codes," IEEE Trans. Commun., vol. 46, no. 8, pp. 1003-1010, Aug 1998.

[22] J. Hagenauer, E. Offer, and L. Papke, "Iterative Decoding of Binary Block and Convolutional Codes," IEEE Trans. Inform. Theory, vol. 42, no. 2, pp. 429-445, Mar. 1996.

[23] R. Lucas, M. Bossert and M. Breitbach, "On Iterative Soft-Decision Decoding of Linear Binary Block Codes and Product Codes," IEEE J. Select. Areas Commun. , vol. 16, no. 2, pp.276-296, Feb. 1998.

[24] A. B. Kiely et al., "Trellis Decoding Complexity of Linear Block Codes," IEEE Trans. Inform. Theory, vol. 42, no. 6, pp. 1687-1697, Nov. 1996.

[25] S. Benedetto et al., "A Soft-Input Soft-Output Maximum Aposteriori (MAP) Module to Decode Parallel and Serial Concatenated Codes," TDA Progress Report vol. 42-127, Jet Propulsion Laboratory, Pasadena, California, pp.1-20, November 15, 1996.

[26] X-A. Wang and S. B. Wicker, "A Soft-Output Decoding Algorithm for Concatenated Systems," IEEE Trans. Inform. Theory, vol. 42, no. 2, pp. 543-553, Mar. 1996.

[27] A. Ambroze, G. Wade and M. Tomlinson, "Iterative MAP Decoding For Serial Concatenated Convolutional Codes," IEE Proc.- Commun. vol. 145, no. 2, pp. 53-59, Apr. 1998.

[28] S. Benedetto et al., "Soft-Output Decoding Algorithms in Iterative Decoding of Turbo Codes," TDA Progress Report vol. 42-124, Jet Propulsion Laboratory, Pasadena, California, pp.63-87, February 15, 1996.

[29] V. Franz and J. B. Anderson, "Concatenated Decoding with a Reduced-Search BCJR Algorithm," IEEE J. Select. Areas Commun. , vol. 16, no. 2, pp.186-195, Feb. 1998.

[30] P. Robertson, "An Overview of Bandwidth Efficient Turbo Coding Scheme," in Proceedings of the Int. Symp. On Turbo Codes and Related Topics, Brest, France, pp.103-110, Sep. 1997.

[31] D. Divsalar and R.J. McEliece, "The Effective Free Distance of Turbo Codes," Jet Propulsion Laboratory and Department of Electrical Engineering California Institute of Technology, Pasadena, California 91125, USA, pp.1-6.

[32] S. Dolinar and D. Divsalar, "Weight Distributions for Turbo Codes Using Random and Nonrandom Permutations," TDA Progress Report vol. 42-122, Jet Propulsion Laboratory, Pasadena, California, pp.56-65, August 15, 1995.

[33] D. Divsalar and F. Pollara, "Transfer Function Bounds on the Performance of Turbo Codes," TDA Progress Report vol. 42-122, Jet Propulsion Laboratory, Pasadena, California, pp.44-54, August 15, 1995.

[34] J. K. Wolf, "A Survey of Coding Theory: 1967-1972," IEEE Trans. Inform. Theory, vol. 19, no. 4, pp. 381-389, Jul. 1973.

[35] E.K. Hall and S. Wilson, " Design and Analysis of Turbo Codes on Rayleigh Fading Channels," IEEE J. Select. Areas Commun. , vol. 16, no. 2, pp.160-174, Feb. 1998.

[36] S. Benedetto, R. Garello and G. Montorsi, "A Search for Good Convolutional Codes to Be Used in the Construction of Turbo Codes," IEEE Trans. Commun., vol. 46, no. 9, pp. 1101-1105, Sep. 1998.

[37] R.J. McEliece, "The General Theory of Convolutional Codes," TDA Progress Report vol. 42-113, Jet Propulsion Laboratory, Pasadena, California, pp.89-98, May 15, 1993.

[38] M-G. Kim, "On Systematic Punctured Convolutional Codes," IEEE Trans. Commun., vol. 45, no. 2, pp. 133-139, Feb. 1997.

[39] P. Komulainen and K. Pehkonen, "Performance Evaluation of Superothogonal Turbo Codes in AWGN and Flat Rayleigh Fading Channels," IEEE J. Select. Areas Commun. , vol. 16, no. 2, pp.196-205, Feb. 1998.

[40] A. Hunt, and S. Crozier, " Hyper-Codes: High-Performance Low-Complexity Error Correcting Codes," in Proceedings of the Biennial Symp. on Communications, Kingston, Ontario, Canada, pp.263-267, May. 1998.

[41] J. Hagenauer, "Source-Controlled Channel Decoding," IEEE Trans. Commun., vol. 43, no. 9, pp. 2449-2457, Sep. 1995.

[42] S.B. Wicker, " Error Control Systems for Digital Communication and Storage," Englewood Cliffs, NJ: Prentice Hall, 1995.

[43] J.G. Proakis, " Digital Communications," 2nd ed., McGraw-Hill, Inc., 1989.

[44] B. Sklar, " Digital Communications: Fundamentals and Applications", Englewood Cliffs, NJ: Prentice Hall, 1988.

[45] C. Heegard and S. B. Wicker, "Turbo Coding," Kluwer Academic Publishers, 1999.

[46] C.E. Shannon, " A Mathematical Theory of Communication," Bell Syst. Tech. J., vol27, pp. 379-423, July 1948.

[47] T.M. Duman and M. Salehi, "On Optimal Power Allocation for Turbo Codes", ISIT 1997, Ulm Germany, pp.104, June 29 – July 4.

Internet Sites:

[48] "An Introduction To Turbo Codes", http://www.ee.vt.edu/valenti/references.html

[49] "JPL TMO Progress Reports", http://tda.jpl.nasa.gov/progress_report/index.html